FACULTY OF APPLIED SCIENCES
UNIVERSITY
OF WEST BOHEMIA

DEPARTMENT OF
COMPUTER SCIENCE
AND ENGINEERING

Doctoral Thesis

# Cross-lingual Sentiment Analysis

Pavel Přibáň

PILSEN, CZECH REPUBLIC

2024

University of West Bohemia
Faculty of Applied Sciences

# Cross-lingual Sentiment Analysis

Ing. Pavel Přibáň

Doctoral Thesis

Submitted in partial fulfillment of the requirements for a degree of Doctor of Philosophy in Computer Science and Engineering

**Supervisor:** Doc. Ing. Josef Steinberger, Ph.D.
Department of Computer Science and Engineering

Pilsen, 2024

Západočeská univerzita v Plzni
Fakulta aplikovaných věd

# Mezijazyčná analýza sentimentu

Ing. Pavel Přibáň

Disertační práce

K získání akademického titulu doktor v oboru Informatika a výpočetní technika

**Školitel:** Doc. Ing. Josef Steinberger, Ph.D.
**Katedra:** Katedra informatiky a výpočetní techniky

Plzeň, 2024

# Declaration

I hereby declare that this Doctoral Thesis is completely my own work and that I used only the cited sources, literature, and other resources. This thesis has not been used to obtain another or the same academic degree.

I acknowledge that my thesis is subject to the rights and obligations arising from Act No. 121/2000 Coll., the Copyright Act as amended, in particular the fact that the University of West Bohemia has the right to conclude a licence agreement for the use of this thesis as a school work pursuant to Section 60(1) of the Copyright Act.

In Pilsen, on 5 March 2024

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Pavel Přibáň

The names of products, technologies, services, applications, companies, etc. used in the text may be trademarks or registered trademarks of their respective owners.

# Abstract

Natural language processing has become an essential part of the artificial intelligence field that is used daily in industry and by millions of people. Sentiment analysis as a fundamental part of natural language processing is no exception.

This thesis presents a detailed study of cross-lingual sentiment analysis. The main goal is to explore, evaluate and propose methods to perform sentiment analysis in languages other than English, with a special focus on Czech. This is achieved by proposing cross-lingual methods and techniques for knowledge transfer between languages.

The core of the thesis consists of performing the zero-shot polarity detection task in a cross-lingual setting. Namely, we use modern multilingual Transformer-based models and linear transformations combined with CNN and LSTM neural networks. We evaluate their performance for Czech, French, and English. We aim to compare and assess the models' ability to transfer knowledge across languages and discuss the trade-off between their performance and training and inference speed. We build strong monolingual baselines comparable with the current SotA approaches, achieving new state-of-the-art results for Czech and French. Furthermore, we compare our results with the outputs of the latest large language models (LLMs), i.e., Llama 2 and ChatGPT.

We show that the large multilingual Transformer-based XLM-R model consistently outperforms all other cross-lingual approaches in zero-shot cross-lingual sentiment classification. We also show that the smaller Transformer-based models are comparable in performance to older but much faster methods with linear transformations. Notably, this performance is achieved with just approximately 0.01 of the training time required for the XLM-R model. It underscores the potential of linear transformations as a pragmatic alternative to resource-intensive and slower Transformer-based models in real-world applications. The LLMs achieved impressive results that are on par or better, at least by 1% – 3%, but with additional hardware requirements and limitations. Overall, we contribute to understanding cross-lingual sentiment analysis and provide valuable insights into the strengths and limitations of cross-lingual approaches for sentiment analysis.

Additionally, a new dataset for Czech subjectivity classification is introduced to partly fulfil this thesis's objectives. Next, we present a novel multi-task approach to improve the results of Czech aspect-based sentiment analysis by leveraging information from the semantic role labeling task. Lastly, we applied prompt-based learning to aspect-based sentiment analysis and sentiment classification in the context of LLMs and the Czech language.

# Abstrakt

Zpracování přirozeného jazyka se stalo důležitou součástí umělé inteligence, kterou denně využívají miliony lidí i firmy v průmyslu. Analýza sentimentu jako přirozená součást zpracování přirozeného jazyka není výjimkou. Tato práce představuje podrobnou studii, která se věnuje mezijazčné analýze sentimentu. Hlavním cílem je prozkoumat, vyhodnotit a navrhnout mezijazyčné metody pro analýzu sentimentu, které dovolují řešit tuto úlohu v jiných jazycích než v angličtině, se zvláštním zaměřením na češtinu.

Jádro práce spočívá v mezijazyčných experimentech s úlohou detekce polarity v tzv. *zero-shot* nastavení, ve kterém jsou k dispozici anotovaná data pouze pro jeden jazyk (zdrojový). Konkrétně v práci využíváme moderní vícejazyčné modely založené na architektuře Transformer a dále modely využívající lineární transformace v kombinaci s neuronovými sítěmi CNN a LSTM. Tyto modely vyhodnocujeme na datových sadách v češtině, francouzštině a angličtině. Naším cílem je porovnat schopnost modelů přenášet znalosti napříč jazyky a zhodnotit kompromis mezi jejich úspěšností a rychlostí trénování a predikce. Pro porovnání jsou vytvořeny základní modely, které dosahují současných state-of-the-art výsledků pro češtinu a francouzštinu. Dále jsou naše výsledky porovnány s výstupy nejnovějších velkých jazykových modelů, tj. modely Llama 2 a ChatGPT.

Ukazujeme, že velký vícejazyčný model XLM-R založený na architektuře Transformer konzistentně překonává všechny ostatní mezijazyčné přístupy při tzv. zero-shot detekci polarity. Dále je ukázáno, že menší modely založené na architektuře Transformer jsou výkonnostně srovnatelné se staršími, ale mnohem rychlejšími metodami používající lineární transformace. Této úspěšnosti je dosaženo jen s přibližně 0,01 času potřebného pro natrénování velkého modelu XLM-R. Tyto výsledky podtrhují potenciál metod založených na lineárních transformacích jako pragmatické alternativy. A to zejména v reálných aplikacích používajících modely založených na architektuře Transformer, které jsou pomalejší a náročné na výpočetní zdroje. Velké jazykové modely (Llama 2 a ChatGPT) dosáhly působivých výsledků, které jsou srovnatelné nebo lepší minimálně o 1% – 3%, ale přinášejí další omezení a požadavky. Celkově přispíváme k pochopení mezijazyčné analýzy sentimentu a poskytujeme cenné zkušenosti o silných stránkách a omezeních mezijazyčných přístupů.

Dále je představena nová česká datová sada pro detekci subjektivity a navrhnuta nová metoda pro zlepšení výsledků aspektově orientované analýzy sentimentu s vyžitím informací z úlohy značkování sémantických rolí. Nakonec jsme použili moderní techniku nazvanou *prompting* pro úlohy aspektově orientované analýzy sentimentu a klasifikaci sentimentu.

# Acknowledgement

I am deeply grateful to my supervisor Doc. Ing. Josef Steinberger, Ph.D. for his guidance, advice and valuable comments. I would also like to thank my family and friends for their endless support.

Computational resources were provided by the e-INFRA CZ project (ID:90254), supported by the Ministry of Education, Youth and Sports of the Czech Republic.

# Contents

# PART I

# Introduction

# Introduction 1

In the very recent past, we have seen exponential growth of interest and usage of natural language processing (NLP) mainly due to the introduction of large language models (LLMs) such as ChatGPT (OpenAI, 2022) or Llama (Touvron, Lavril, et al., 2023; Touvron, Martin, et al., 2023) models. In NLP, sentiment analysis (SA) holds significant prominence.

The goal of SA is to extract opinion, sentiment, attitude and other subjective information from textual data (B. Liu et al., 2010). It is applied daily in real-world applications, allowing companies in the industry to perform product and service review evaluation, customer feedback analysis, analysis of public opinions during elections or monitoring of a commercial brand on social media. These applications underscore the importance of SA.

While SA has been extensively studied, the research has been devoted almost exclusively to individual languages in a monolingual setting, with the lion's share dedicated to English. However, the advent of the Internet and the enormous volume of user-generated text across various languages via communication channels, social media, and other mediums have proven the significance of analyzing text in multiple languages. Consequently, organizations and companies have begun acknowledging the importance of leveraging multilingual and cross-lingual approaches in their text analysis endeavors. Thus, the research focus has partly shifted towards exploring the possibilities of multilingual and cross-lingual approaches.

Cross-lingual sentiment analysis (CLSA) has received considerably less attention than traditional monolingual SA approaches. Typically, existing SA methodologies rely on the supervised machine learning paradigm, necessitating annotated data to train machine learning models. The CLSA is a challenging task that aims to enable SA in other languages (target languages) with limited or no annotated data by transferring knowledge from a language (called source language), typically English, where the annotated data are available. The advanced and more difficult version of CLSA is the so-called zero-shot CLSA, in which only data from the source language (e.g., English) are used to build the system.

The core part of this thesis is devoted to zero-shot cross-lingual sentiment classification. Sentiment classification, also known as polarity detection, is a classification task where the goal is to assign a sentiment polarity of a given text, usually with the labels *positive*, *negative* and *neutral*. To address the challenge of CLSA, we propose several approaches aligned with the thesis goals. Our proposed approaches encompass a spectrum of techniques, including recurrent neural networks, convolutional neural networks, BERT-like models, and the latest Large Language Models (LLMs). We thoroughly evaluate and compare these approaches and discuss their strengths and limitations. Through this exploration, we aim to shed light on the efficacy of each approach and provide insights into their applicability in real-world scenarios.

Further, we introduce a new Czech dataset for subjectivity classification, designed for utilization in cross-lingual benchmarks. Next, we focus on monolingual SA in Czech by achieving new state-of-the-art results. We also propose a novel multi-task method to improve the results of Czech aspect-based sentiment analysis (ABSA) by leveraging information from semantic role labeling. Lastly, we applied prompt-based learning to the ABSA and sentiment classification tasks in the context of LLMs and the Czech language.

## 1.1 Motivation

As we mentioned, the CLSA remains a less studied part of SA, although it has a very important and beneficial possible usage in practice. The real applications for SA are still challenging because of several aspects, such as multilinguality and domain dependence. Another aspect that makes SA difficult is that most of the available datasets are annotated for English texts and low-resourced languages suffer from a lack of annotated datasets on which the available approaches could be trained.

The newest LLMs in zero-shot setting provide similar performance for the polarity detection task compared to the SotA results obtained by fine-tuned Transformer-based models. However, these SotA results are redeemed by substantial computational resources. Also, as shown in W. Zhang et al. (2023), the LLMs are significantly outperformed in more complex tasks, such as aspect-based sentiment analysis by the fine-tuned models. Another limitation of the very recent works with LLMs is their exclusive focus on sentiment analysis evaluations conducted nearly solely in English. We aim to compare the latest approaches with older yet significantly faster methods based on linear transformations to address these limitations and provide a comprehensive evaluation. This comparative analysis aims to delineate the trade-offs between computational efficiency and performance in SA.

The results of existing cross-lingual methods for SA can hardly be compared with each other because each work usually uses a different dataset or pairs of languages. In contrast, considering all performance aspects, we aim to compare different cross-lingual methods regarding accuracy, training and inference speed in three languages. The existing cross-lingual works are usually restricted only to English, French, Spanish or Chinese and are merely dedicated to accuracy while completely ignoring other aspects, such as training or inference speed, which are crucial in real-world deployment.

Our main motivation is to contribute by partly filling the mentioned research gaps. We propose approaches for CSLA that can be applied in real-world applications. We aim to evaluate them in different aspects (e.g., speed and performance) and discuss their advantages and disadvantages. This comprehensive evaluation will enable a more nuanced understanding of the trade-offs between accuracy and efficiency, thereby facilitating informed decision-making for real-world deployment of cross-lingual SA systems. Additionally, at the time of specifying the thesis goals, there was little progress for the Czech monolingual SA in the context of recent Transformer-based models, so another motivation was to apply the most recent approaches to the task of Czech sentiment analysis.

## 1.2 Thesis Goals

The objectives of this thesis were specified and set in the author's Ph.D. thesis exposé (Přibáň, 2020). The main goal is to explore, evaluate and propose methods to perform SA in languages other than English, specifically in the Czech language. This can be achieved by employing existing approaches or by proposing novel cross-lingual methods and techniques for knowledge transfer between languages. Additionally, new datasets may be introduced to fulfil the objective of this thesis. Consequently, the thesis aims at the following research tasks:

1. Tackle the problem of lack of annotated data in languages other than English by introducing new resources.

2. Perform sentiment analysis (and other related) tasks in languages other than English by applying cross-lingual methods and transforming knowledge between languages.

3. Apply recent state-of-the-art pre-trained models and transfer learning approaches to sentiment analysis (and other related) tasks to textual data other than English.

## 1.3 Contributions

Regarding the defined thesis goals, we provide a brief overview of our contributions to each goal. In Section 9.2, we provide more details on the fulfilment of the defined goals of this thesis

1. We created the first Czech manually annotated dataset for subjectivity classification (Přibáň & Steinberger, 2022). Next, in Šmíd and Přibáň (2023) and Šmíd et al. (2024), we completely reannotated the existing Czech dataset for aspect-based sentiment analysis to the same format as its counterparts in other languages. Consequently, the dataset can be used for cross-lingual experiments between Czech and several other languages. Additionally to this goal, we also built new multilingual resources for named entity recognition (Piskorski et al., 2019, 2021; Yangarber et al., 2023) and fact-checking (Přibáň et al., 2019) tasks.

2. We perform zero-shot cross-lingual sentiment analysis between English, Czech and French by using linear transformations that allow external knowledge transfer between the languages in Přibáň et al. (2022). In our subsequent investigation, detailed

in Přibáň et al. (2024), we study the usage of linear transformations more deeply. The fulfilment of this goal is also partly supported in Přibáň and Balahur (2023), where we compared multilingual systems for a real-world application.

3. In Přibáň and Steinberger (2021), we delved into the cross-lingual sentiment analysis by leveraging recent multilingual Transformer-based models in the context of Czech and English. Building upon this exploration, our subsequent work in Přibáň et al. (2024) expands the scope to include the French language. We employed the most recent LLMs, such as Llama 2 and ChatGPT. In the same work, we provide a comprehensive overview of current approaches for cross-lingual sentiment analysis, offering an exhaustive evaluation of selected methods and a discussion of their merits and drawbacks. Additionally, we performed cross-lingual experiments on the newly built subjectivity dataset (Přibáň & Steinberger, 2022) with multilingual Transformer-based models.

## 1.4 Outline

The thesis is structured into three main parts: (1) Introduction, (2) Theoretical Background and (3) Thesis Contributions. The theoretical background part is designed to supplement and provide additional information and context for the main thesis contributions. It is intended for readers seeking deeper insights into the concepts and background behind CLSA and the specific approaches adopted in our research.

In the theoretical part in Chapter 2, we introduce the theory related to SA. Chapter 3 is focused on supervised machine learning, usually applied to text classification and SA. Chapter 4 is devoted to the transfer learning techniques, models and approaches. In Chapter 5, we describe common datasets and techniques employed in SA. Chapter 6 contains a description of multilingual approaches and the underlying cross-lingual techniques related to SA.

The thesis contributions part provides the core results for CLSA in Chapter 7. Chapter 8 presents additional publications related to SA along with other research contributions. Finally, in Chapter 9, we summarize our contributions, provide a brief overview of our publications and provide a conclusion for fulfilling the thesis goals.

# PART II

# Theoretical Background

# Sentiment Analysis 2

Sentiment analysis plays a crucial role in NLP as it aims to detect, understand and extract subjective information (e.g., opinions, sentiments and emotions) expressed in text (B. Liu et al., 2010). At the beginning of the 21st century[1], it has become one of the fastest-growing research areas in NLP (M. V. Mäntylä et al., 2018).

## 2.1 Tasks Overview

Generally, the field of *sentiment analysis*, also known as *opinion mining*, can be seen as a collection of distinct tasks related to subjective information extraction and other sub-tasks that are relevant and linked to these tasks. In this section, we summarize, describe and define the most common tasks.

B. Liu et al. (2010) define and describe several tasks within the field of SA, i.e., polarity detection at *document-level, sentence-level, aspect-based-level* and *comparative SA*. From B. Liu (2012), we can also add *subjectivity classification* and *opinion spam detection*. Further, we briefly describe these tasks and later in this chapter, we define some of them more precisely.

Similar to the terminology used by B. Liu (2012), we use the terms *opinion* and *sentiment* interchangeably to refer to opinion, sentiment, attitude and emotion. However, it is important to note that they are not always equivalent and we will distinguish them when necessary. In general, the research of SA has been conducted mainly for polarity detection tasks at three levels of granularity: *document-level, sentence-level* and *entity/aspect-based level*. In this chapter, we will collectively refer to these three types jointly as *sentiment analysis* or *polarity detection* tasks and again distinguish between them when needed. These tasks can also be considered as text classification problems, typically solved using conventional supervised machine learning techniques, see Chapter 3 and 5.2. Based on Feldman (2013), B. Liu et al. (2010), B. Liu (2012), Medhat et al. (2014), and Pang and Lee (2008), we summarize the tasks as follows:

[1] According to M. V. Mäntylä et al. (2018), nearly 7,000 papers related to SA have been published and indexed in the Scopus database (https://scopus.com) between 2004 and 2016. Google Scholar search engine (https://scholar.google.com) returns around 348,000 records for the query *"sentiment analysis"* in February 2024.

1. **Polarity detection:** The objective is to identify the sentiment polarity (*positive, negative, neutral*) expressed towards a given target. The polarity can also be defined with a different number of labels, i.e., *very positive, positive, neutral, negative, very negative*, which is often referred to as *fine-grained sentiment analysis*. Alternatively, polarity can be simplified to a binary text classification problem with only *positive* and *negative* labels. Polarity detection can be further divided into the three levels[2]:

   - **Document-level:** At this level, the task involves assigning an overall sentiment polarity to a given document. For example, given a short Twitter text *"I love the new Zombieland movie #cinema"* which is a review about a particular movie written by a user, the task is to decide whether the user has a positive sentiment (likes the movie) or a negative sentiment (dislikes the movie). In this task, it is assumed that the document contains only one opinion about a single entity.

   - **Sentence-level:** This task is almost identical to the *document-level* task but performed on individual sentences instead of entire documents. The goal is to classify whether a sentence expresses a positive, negative or neutral sentiment. Again, it is assumed that the sentence contains only a single sentiment or opinion.

   - **Entity and Aspect-based level:** The *aspect-based* sentiment analysis (ABSA) task[3] evaluates sentiments associated with individual entities and/or their aspects. Consider the following review of a hotel: *"The room was very comfortable and the breakfast was great."*. There are two aspects of the hotel – *room* and *breakfast* and both of them are positive. This task allows the evaluation of sentiment in any text with multiple sentiments and multiple entities and their aspects.

2. **Subjectivity Classification:** The sentence-level SA is usually performed only on sentences with the sentiment, opinion or subjective views. The goal of the *subjectivity classification* task (J. Wiebe & Riloff, 2005; J. Wiebe et al., 1999) is to identify *objective sentences* that convey factual information and *subjective sentences* that express subjective views and opinions. However, it is important to note that subjective sentences may not necessarily convey any sentiment, as discussed in Section 2.5.

3. **Emotion Detection (Analysis):** In the *emotion detection* task, the system intended for this task must detect a person's emotion expressed in a text. Emotions represent the subjective feelings and thoughts of human beings. Parrott (2001) distinguishes six primary emotions, i.e., *love, joy, surprise, anger, sadness* and *fear*. These emotions can further be subdivided into additional sub-categories.

[2] Although the definitions of *sentence-level* and *document-level* are almost identical, they are listed separately based on convention in the literature.

[3] In this context, the word *aspect* can be used interchangeably with the word *feature*; thus the task is also called *feature-based sentiment analysis*.

4. **Other:** There are several additional tasks related to SA like *Comparative Sentiment Analysis, Sentiment Lexicon Acquisition (Generation), Sarcasm Detection (Analysis), Opinion Summarization, Opinion Spam Detection,* and others. We describe some of them in Section 2.6.

For the next parts of this thesis, it is necessary to establish clear definitions and explanations of opinions and other related terms.

## 2.2 Opinion Definition

We define *opinion* for the SA task according to B. Liu (2012) as a quadruple:

$$(g, s, h, t) \tag{2.1}$$

where $g$ is an opinion target, $s$ is an opinion polarity (sentiment), $h$ is an opinion holder and $t$ is a time when the opinion was expressed. For the explanation of components of the opinion, we will use a similar example to the one stated in B. Liu (2012). The example:

> Author: Nick Newman, 25/10/2019
>
> "(1) *I really like my new Samsung TV.* (2) *I cannot live without it.* (3) *The resolution is unbelievable.* (4) *But the price is not so good as the resolution.* (5) *My friends love it too.* (6) *This Samsung TV is definitely better than my old Philips TV."*

The opinion target $g$ can be any entity or aspect of the entity about which the opinion has been expressed. For example, in sentence (1), the target of the opinion is *Samsung TV* with positive sentiment. The example of a target, which is an aspect, is in sentence (3), where the target is *resolution*.

Secondly, the example contains opinions of two entities. Sentences (1), (2), (3), (4), (6) are opinions of the author of the review (*Nick Newman*) and in the sentence (5) expressed an opinion of the author's friends. These entities are referred to as *opinion sources* or *opinion holders* (S.-M. Kim & Hovy, 2004; J. Wiebe et al., 2005). Lastly, the date of the example is 25/10/2019 and the reason why the opinion definition contains time $t$ (or date) is that the sentiment can evolve over time and it is useful to observe these changes.

### 2.2.1 Entity Definition

Next, we define the term ***entity*** as the target object that is being evaluated. An ***entity e*** can be a product, service, topic, issue, person, organization or event. Formally it is defined as a pair *e:(T,W)*, where *T* is a hierarchy of *parts* (or *components*) and *sub-parts* of the entity and *W* is a set of *attributes* of *e*. For example, one part of the *Samsung TV* is a screen which is composed of other sub-parts like screen glass, LED display, frame etc. The root node is the entity itself (*Samsung TV*) and other nodes contain parts and sub-parts. Each part or sub-part has its own set of attributes. For example, a resolution is an attribute of the LED display.

An opinion can be expressed on any node or on any attribute of the node. In the previous example, in sentence (1), the author expressed his opinion on the *Samsung TV* itself (root node) and in sentence (2), he expressed his opinion on one of its attributes (*resolution*).

This hierarchical description of an entity with any number of levels and nested relations is universal but often too complex for some real applications. The difficulty of applying SA for such a universal hierarchical definition is tough and challenging. Thus, we simplify the hierarchy according to B. Liu et al. (2010) to two levels and use the term ***aspects*** to denote both parts (sub-parts) and attributes, see Figure 2.1. After the simplification, the root of the node is still the entity[4] itself and the other nodes are aspects of the entity.

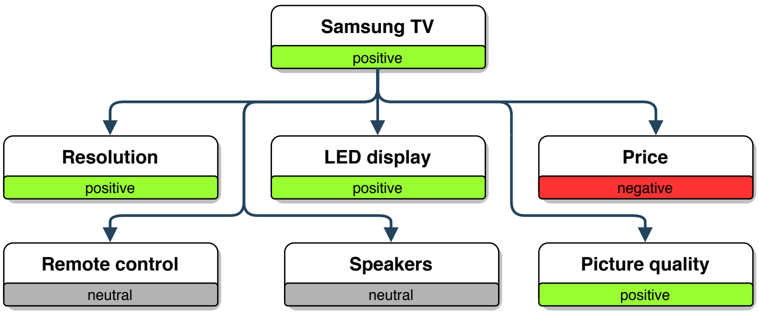


Figure 2.1: Simplified example of hierarchical representation of an entity (*Samsung TV* entity).

## 2.2.2 Opinion for Aspect-based Sentiment Analysis

The previous definition of opinion in Section 2.2 was sufficient for a text unit (document, sentence, paragraph) with one opinion towards a single entity. However, in the context of the ABSA task, the objective is to discover all or multiple opinions expressed towards individual entities and/or their aspects within a given opinion document $d$. Therefore, we expand the previous opinion definition by incorporating the entity definition from the preceding section, following B. Liu (2012), resulting in a quintuple:

$$(e_i, a_{ij}, s_{ijkl}, h_k, t_l) \tag{2.2}$$

where $e_i$ is a name of an entity, $a_{ij}$ is a $j$-$th$ aspect of entity $e_i$, $h_k$ is an opinion holder, $t_l$ is the time when the opinion was expressed and $s_{ijkl}$ is a sentiment on aspect $a_{ij}$ of entity $e_i$ in time $t_l$ expressed by opinion holder $h_k$. In the case where the overall opinion is expressed towards the entity itself, a special aspect named *GENERAL* is used to denote it. The $e_i$ and $a_{ij}$ pair substitute the target $g$ from the definition 2.1.

[4] Entity is sometimes also called *object* and aspects can also be called *features, facets, attributes* or *topics*.

The opinion document $d$ (or another unit of text like paragraph, sentence) is then composed of a set of opinion quintuples $O = \{o_1, o_2, \ldots, o_m\}$ expressed on a set of entities $E = \{e_1, e_2, \ldots, e_r\}$ and their aspects with a set of opinion holders $H = \left\{h_1, h_2, \ldots, h_p\right\}$ at some certain time point. An entity $e_i$ is represented by itself and by a set of aspects $A_i = \{a_{i1}, a_{i2}, \ldots, a_{in}\}$.

Aspects of a specific entity can be categorized as either *implicit* or *explicit*. Explicit aspects are typically conveyed through nouns or noun phrases. For example, in the sentence *"The **resolution** of the Samsung TV is impressive."*, the word *resolution* represents the explicit aspect. On the other hand, implicit aspects are usually expressed with adverbs, adjectives or even verbs. For example, the sentence *"This Samsung TV is really expensive."* implies that there is an aspect *price* with negative sentiment, although the price was not explicitly mentioned in the sentence.

Both definitions of opinions (2.1 and 2.2) are not able to handle all possible options and cases in which opinion can be expressed, but they are sufficient for most applications. Examples in which these definitions fail are shown in B. Liu (2012).

### 2.2.3 Opinion Types

In the preceding sections, we described only one type of opinion, which is called *regular opinion* (definition 2.2), but there are two main types of opinions – ***regular opinion*** and ***comparative opinion***.

- **Regular opinion** or just *opinion* can be divided into two categories (B. Liu, 2006):
  - **Direct opinion** is expressed directly towards an entity or its aspect, for example, "The resolution is unbelievable."
  - **Indirect opinion** is expressed indirectly on an entity or its aspect on some other entities. For example, *"Once I finished the lunch I had a stomachache and I was vomiting the whole day."* implies that the food was spoiled and the person (the other entity) vomited, which implies a negative opinion towards the food.

- **Comparative opinion** expresses a relation of similarities or differences between two or more entities and/or a preference of the opinion holder based on some of the shared aspects of the entities (Jindal & Liu, 2006a, 2006b; B. Liu, 2006). For example, the sentence (6) from the example at the beginning of this Section 2.2, contains comparative opinion. Comparisons can be divided into two main groups ***gradable comparison*** and ***non-gradable comparison*** which are described in more detail in B. Liu (2006, 2012).

Next, we recognize ***explicit opinion*** and ***implicit opinion*** which are defined according to B. Liu (2012) as follows:

- **Explicit opinion** is a ***subjective statement*** that gives a regular or comparative opinion, for example, *"I really like my new Samsung TV."* or *"This Samsung TV is definitely better than my old Philips TV."*

- **Implicit opinion** is an ***objective statement*** that implies a regular or comparative opinion. For example, *"My new Samsung TV has stopped working after a few days"* or *"The resolution of my new Samsung TV is higher than my old Philips TV."* Implicit opinions often express some desirable or undesirable features, defects, properties, attributes or consequences for target entities or their aspects.

### 2.2.4 Author and Reader Standing Point

The opinion can be considered from two perspectives, the author of the opinion (opinion holder) or the reader of the opinion. This duality allows for the possibility that a single sentence can be interpreted as either negative or positive, depending on the perspective. For example, in a sentence: *"Our national team lost against Germany, which is really bad"* the author expresses a negative opinion about the loss against Germany, but for a reader who is a fan of Germany, it holds the positive sentiment. Usually, the opinion holders are assumed to be the consumers unless otherwise specified (B. Liu, 2012).

## 2.3 Polarity Detection

Here, we summarize the three primary SA tasks (document-level, sentence-level and aspect-based level) from Section 2.1. These tasks aim to a polarity detection[5] in text.

### 2.3.1 Document-level

The goal of *document-level* polarity detection is to detect the overall sentiment polarity $s$ expressed towards a particular entity in a given opinion document $d$ by an opinion holder at a specific time. According to B. Liu (2012), the task involves extracting an opinion given by the quintuple defined in 2.2 with aspect GENERAL in the following way:

$$(\_, GENERAL, s, \_, \_)$$

assuming that the entity $e$, opinion holder $h$ and time of the opinion $t$ are either known or considered irrelevant. This definition also assumes that the opinion expressed in document $d$ is aiming only on one entity $e$ (if known) and there is only one opinion holder $h$.

Because of this assumption and because the aspect is always GENERAL, we can use the simpler definition of opinion given by the quadruple defined in 2.1 and redefine the task as obtaining only the overall sentiment $s$ for a given document $d$, resulting in the following quadruple:

[5] The *polarity detection* task can also be referred to as *sentiment analysis* or *sentiment classification*.

$$(\_, s, \_, \_)$$

and once again, this assumes that $g$, $h$, and $t$ are known or irrelevant.

### 2.3.2 Sentence-level

The *sentence-level* polarity detection aims to detection of sentiment in individual sentences. In this task, we still assume that the sentence contains only one opinion towards one entity[6]. One sentence can be considered as a single document, thus, similar or identical approaches to *document-level* task can be applied. The *sentence-level* polarity detection task can be employed for longer documents, where each sentence is evaluated independently as a separate document. This approach results in a set of sentences, each assigned a sentiment label. Alternatively, the assigned sentiments can be summarized to represent the overall sentiment of the entire document.

B. Liu (2012) defines the *sentence-level* polarity detection as follows: given a sentence $x$, determine whether $x$ expresses a positive, negative or neutral opinion. If there is no opinion, the sentence is considered neutral. It is important to note that the definition of the neutral class may vary in different datasets. In some cases, neutral sentences may be defined as sentences with a mild sentiment (e.g., slightly negative or positive) and only sentences with strongly expressed sentiment are considered to be either positive or negative. This concept applies to other polarity detection tasks as well.

The sentence-level polarity detection is closely related to a task known as *subjectivity classification*. Subjectivity classification aims to differentiate between sentences that provide factual information (objective sentences) and sentences that convey subjective views and opinions (subjective sentences). However, we should note that subjectivity is not equivalent to sentiment as many objective sentences can imply opinions, see Section 2.5.

For most cases in practice, the *sentence-level* and *document-level* polarity detection tasks are suitable for short reviews, social media posts or other short text with one or few sentences expressing one opinion towards one entity.

### 2.3.3 Entity and Aspect-based Level

The two previously described tasks were focused on capturing the overall sentiment towards a single entity in a document or sentence. However, in many cases, documents and sentences contain multiple opinions or sentiments expressed towards one or multiple entities or their aspects. This is particularly relevant in the analysis of product reviews, where individuals often express opinions towards specific aspects or attributes of the product. In such cases, ABSA becomes crucial, as it allows for the detection and evaluation of individual sentiments or opinions towards each aspect or attribute of the entity. ABSA provides a more detailed

[6] Despite the fact that this assumption is incorrect in many examples.

and comprehensive understanding of the sentiments expressed in the text, enabling a more fine-grained analysis of opinions.

The complete definition of the task is introduced in Section 2.2.2. To recall, the goal of this task is to obtain all quintuples $(e_i, a_{ij}, s_{ijkl}, h_k, t_l)$ for document $d$, where $e_i$ is a name of an entity, $a_{ij}$ is a *j-th* aspect of entity $e_i$, $h_k$ is an opinion holder, $t_l$ is the time when the opinion was expressed and $s_{ijkl}$ is a sentiment on aspect $a_{ij}$ of entity $e_i$ in time $t_l$ expressed by opinion holder $h_k$. In the case where the overall opinion is expressed towards the entity itself, a special aspect named *GENERAL* is used to denote it. In practice, some members of the quintuple 2.2 can be omitted because they are unimportant, irrelevant or known. ABSA is a complex task that consists of several sub-tasks (B. Liu, 2006; B. Liu et al., 2010; B. Liu, 2012):

1. **Entity extraction and categorization:** Find and extract all mentions and synonyms of entities in a given document $d$ and assign them corresponding categories. Each category then represents one entity $e_i$ from a set of entities $E = \{e_1, e_2, \ldots, e_r\}$.

2. **Aspect extraction and categorization:** Find and extract all aspect expressions for all entities obtained in the first task and classify the aspect expressions. Each entity $e_i$ has its own set of aspects (categories) $A_i = \{a_{i1}, a_{i2}, \ldots, a_{in}\}$ where $a_{ij}$ represents one unique aspect of entity $e_i$.

3. **Opinion holder extraction and categorization:** Find and extract opinion holders or their mentions from text or structured data and assign them corresponding categories. The output of this task is a set of opinion holders $H = \left\{h_1, h_2, \ldots, h_p\right\}$.

4. **Time extraction and standardization:** Extract the times when opinions were expressed and standardize different time formats.

5. **Aspect sentiment classification:** Classify an opinion on aspect $a_{ij}$ as positive, negative or neutral or assign another predefined sentiment classes.

6. **Opinion quintuple generation:** For document $d$, generate all opinion quintuples $(e_i, a_{ij}, s_{ijkl}, h_k, t_l)$, with results from previous tasks.

In reality, the sub-tasks are often redefined with slight modifications. Thus, new definitions and combinations of ABSA sub-tasks (Barnes et al., 2022; L. Dong et al., 2014; Pontiki et al., 2014, 2015, 2016; Saeidi et al., 2016; H. Wan et al., 2020) emerge over time.

## 2.4 Emotion Analysis

Emotions are our subjective feelings and thoughts that can be experienced or expressed with varying levels of intensity[7]. The intensity denotes the degree or quantity of the emotion, for

[7] Intensity differs from *arousal* dimension from the valence-arousal-dominance model. Arousal represents whether an emotion is calming or exciting.

example, *really excited, very happy* or *little bit angry* etc. (Bostan & Klinger, 2018; Canales & Martínez-Barco, 2014; B. Liu et al., 2010; B. Liu, 2012; Mohammad et al., 2018; Shrivastava et al., 2019).

## 2.4.1 Categorical Model

Emotions have been studied in different research areas, e.g., psychology, philosophy, sociology and more recently in NLP. Humans can perceive many different emotions. According to the *basic emotion model* (also called *categorical model*) (Ekman, 1992; Frijda, 1988; Parrott, 2001; Plutchik, 1980) emotions can be categorized into distinct emotion classes. For example, emotions like *joy, sadness, anger, fear* are considered to be more basic than others, i.e., physiologically, cognitively and in terms of the mechanisms to express these emotions (Mohammad et al., 2018).

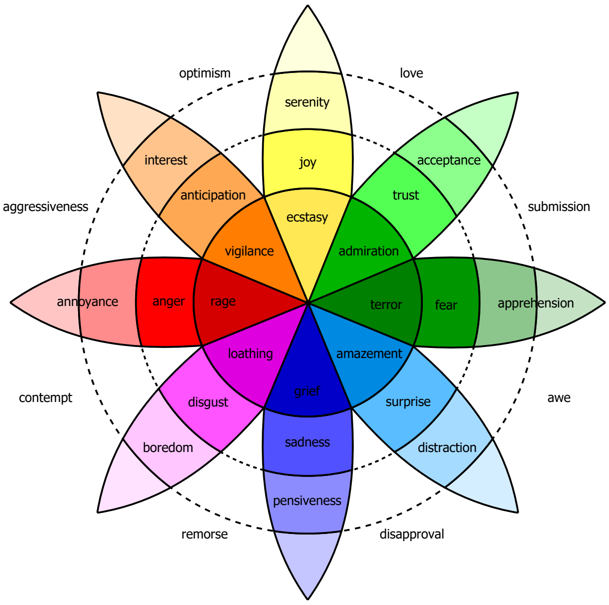


Figure 2.2: Plutchik's wheel of emotions. Picture taken from Commons (2020).

The definitions in different publications can vary slightly, but the basic idea remains the same. For example, Parrott (2001) distinguishes six primary emotions, i.e., *love, joy, surprise, anger, sadness* and *fear*, which can be further divided into other sub-categories. Ekman (1992) also recognizes six (but slightly different) basic emotions, i.e., *anger, disgust, fear, joy, surprise* and *sadness*. Plutchik (1980) claims that there are eight basic emotions (in the *Plutchik's wheel of emotions*, see Figure 2.2), i.e., *joy, sadness, anger, fear, trust, disgust, surprise* and *anticipation* (the inner circle), and other more complex emotions are in the outer circles, outer circles are also composed of emotions with a smaller degree of intensity. Each primary emotion has a polar opposite, e.g., anticipation is the opposite of surprise. The *Plutchik's wheel of emotions* can be seen as a hybrid model between the categorical and dimensional models. However, here we treat it as a categorical model because the emotions are expressed discretely and not as a continuous number in the $n$-dimensional space as it is in the *Valence-Arousal-Dominance* model. *Emotions* and *opinions* are not equivalent but are closely related and have a significant intersection.

### 2.4.2 Dimensional Model

*Dimensional emotion model* represents emotions in $n$-dimensional space. In *Valence-Arousal-Dominance* (VAD) dimensional model, the emotions are points in a three-dimensional space. The model states that there are three largely independent emotional dimensions of word meaning, see Figure 2.3.

The *valence* dimension (positiveness-negativeness / pleasure-displeasure) reflects the attractiveness or sentiment of an emotion. The *arousal* dimension (active-passive) represents an activation level of the emotion. The *dominance* dimension (dominant-submissive) represents a level of control over the emotion (M. Mäntylä et al., 2016; Mohammad, 2018; Osgood et al., 1957; Russell, 1980, 2003). For example, the word *birthday* indicates more positiveness than the word *death*; *nervous* indicates more arousal than *lazy*; and *fight* indicates more dominance than *fragile*.

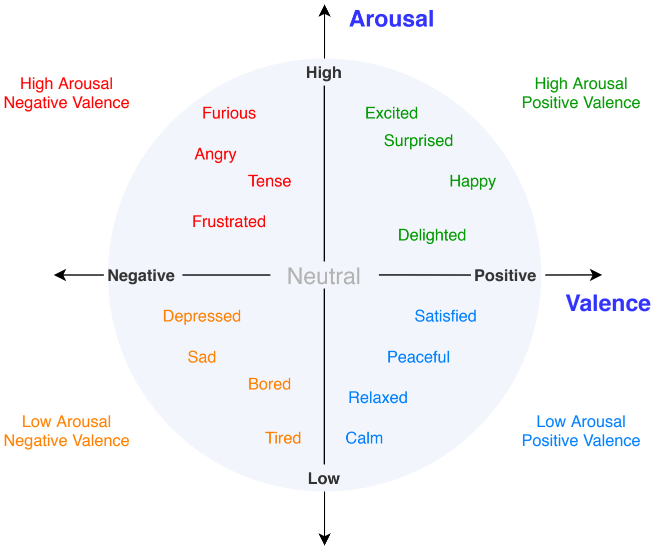


Figure 2.3: Joint visualization of the *Arousal* and *Valence* dimensions with examples of emotions.

The existing approaches for emotion analysis often adopt categorical models because of their simplicity, where emotions can be categorized into distinct classes or categories.

### 2.4.3 Emotion Analysis Tasks

The primary task in *Emotion Analysis* is the ***emotion detection***, where the goal is to detect various emotions in a given text (B. Liu, 2012; Medhat et al., 2014; Shrivastava et al., 2019). Another related task is called ***emotion intensity detection*** task. In this task, the intensity of a given text and emotion need to be detected.

The individual tasks, along with the corresponding datasets, are frequently introduced in public competitions. The emotion intensity task was part of *SemEval-2018 Task 1: Affect in Tweets* competition (Mohammad et al., 2018). There were also other shared tasks related to emotion intensity; SemEval-2007 Task 14 (Strapparava & Mihalcea, 2007) and WASSA-2017 shared task on Emotion Intensity (Mohammad & Bravo-Marquez, 2017).

In the shared competition called *Implicit Emotion Shared Task*[8] (IEST) (Klinger et al., 2018) the participants were asked to create a system that should infer one of six emotions only from the context of a particular emotion word removed from the text. For example, *"It's [#TARGETWORD#] when you feel like you are invisible to others."*, the missing word was *sad* and the system should detect *sadness* emotion.

## 2.5 Subjectivity Classification

The concept of *subjectivity* is closely related to SA and opinion mining. According to B. Liu (2012) *subjective* sentence expresses personal feelings, views or beliefs, whereas *objective* sentence holds some factual information about the world. The subjective sentence can be expressed in many ways, e.g., opinions, emotions, stances, allegations, desires, beliefs, suspicions or speculations (Riloff et al., 2006; J. Wiebe, 2000; J. Wiebe et al., 1999). For example, *"There is one police station in our town."* is an example of an objective sentence and *"Our police are really bad at their job."* is an example of a subjective sentence.

The goal of ***subjectivity classification*** task is to determine whether a given sentence is subjective or objective (Feldman, 2013; B. Liu, 2012; Pang & Lee, 2008; Riloff & Wiebe, 2003; J. Wiebe & Riloff, 2005; J. Wiebe et al., 1999). The subjectivity classification task is considered by some researchers (Medhat et al., 2014) as the first step in sentiment classification to filter out objective sentences that are assumed (incorrectly) to express or imply no opinion.

It is important to note that *subjective* text and *opinionated* text are not equal, although both concepts have a wide intersection. The *opinionated* text expresses or implies positive or negative sentiment. For example, a subjective sentence does not necessarily have to contain any sentiment, as shown in the following sentence *"I think he should visit his doctor"*. Similarly, an objective sentence can imply opinion or sentiment thanks to desirable or undesirable facts (Feldman, 2013; B. Liu, 2012). For example, *"I did not have to repair my Ford for ten years."* implies positive sentiment towards the car because of the desirable fact that the car is reliable, the same applied for the following objective sentence *"In XY store I bought a new computer*

[8] It was a part of *9th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis* (*WASSA 2018*).

*and they gave me five PC games for free."* also implies positive sentiment. The opinions in the previous objective sentences were *implicit opinions*. Of course, it is much easier to detect sentiment in subjective text because it is much more often expressed directly, unlike in the case of sentiment in objective text. Researchers often do not distinguish between *subjective* and *opinionated* sentences and treat them as equal.

## 2.6 Other Tasks

In this section, we briefly describe some other tasks related to sentiment analysis in general.

### 2.6.1 Comparative Sentiment Analysis

Sentiment or opinion in a text does not necessarily have to be expressed directly, but a comparison can be used instead. For example, *"Apple iPhone Xs is super reliable"* is a typical *regular opinion* and *"Apple iPhone Xs is much more reliable than Samsung Galaxy S9"* is a typical *comparative opinion* as we defined in Section 2.2.3. Such sentence contains the *comparative opinion*. The goal of this task is to identify sentences that contain comparative opinions, extract the comparative opinions expressed in the sentences and select the preferred entities (Apple iPhone Xs, in our example) (Feldman, 2013; B. Liu, 2006, 2012). See Přibáň (2020) for details about this task.

### 2.6.2 Lexicon Generation

Words that carry information about sentiment are crucial for SA and other related tasks. These words are called *sentiment words* or *opinion words*. For example, *excellent, nice, beautiful* and *cool* are positive sentiment words and *awful, bad* and *nasty* are negative sentiment words. Along with sentiment words, there are some specific phrases and idioms that also hold sentiment, for example, *"He was on cloud nine"* means that someone is happy, which implies positive sentiment. A list of such words, along with their sentiment orientation, is called *sentiment lexicon*. Sentiment lexicon can be directly used for solving one of the SA tasks or as a source for feature extraction for supervised learning algorithms.

Sentiment lexicon consists of a set of tuples of a *lexical unit* (word, phrase or idiom) and its *sentiment score* (B. Liu, 2012; Medhat et al., 2014; Singh et al., 2018). The sentiment score can be represented in the following ways:

1. A binary indication of positive or negative sentiment polarity, for example: excellent = 1, nice = 1, bad = 0.

2. A set of predefined values, like *very negative, negative, neutral, positive* and *very positive*, for example: excellent = *very positive*, nice = *positive*, bad = *negative*.

3. A real number from a predefined interval, for example, $[-1, 1]$ where negative values refer to negative sentiment and positive values refer to positive sentiment, for example: excellent = 0.78, bad = −0.35.

There are generally three main approaches for generating or obtaining sentiment lexicon (B. Liu, 2012; Medhat et al., 2014). The *manual approach* is very time-consuming and expensive. Thus, it is often used as a verification for the other two automatic approaches, i.e., *dictionary-based approach* and *corpus-based approach*. See B. Liu (2012), Medhat et al. (2014), and Přibáň (2020) for a description of these approaches.

### 2.6.3 Sarcasm Detection

Recognizing the real meaning of some expressions in a natural language, like irony, sarcasm or satire, is a challenging task not only for computers but sometimes even for humans. These terms are closely related and (sometimes are considered interchangeable) we do not distinguish between them (Reyes et al., 2012). The goal of *Sarcasm Detection* is to identify sarcastic sentences or other pieces of text (Davidov et al., 2010; B. Liu et al., 2010; Reyes et al., 2012). In the context of SA, the meaning of a positive expression is usually intended to be negative and vice versa (B. Liu, 2012). This is the main reason why the task is so challenging and difficult. Other related work can be found in Van Hee et al. (2018) and L. Zhang et al. (2018) and for Czech in Ptáček et al. (2014).

## 2.7 Sentiment Analysis Tasks in Thesis

Here, we provide an overview of our research contributions in the realm of Sentiment Analysis tasks. In this thesis, we mainly focus on the *polarity detection* task and its versions described in Section 2.3.

Our works presented in Přibáň and Balahur (2023), Přibáň and Steinberger (2021), Přibáň et al. (2022, 2024), and Sido et al. (2021) are primarily oriented towards document and sentence-level polarity detection. Additionally, our publications (Přibáň & Pražák, 2023; Šmíd & Přibáň, 2023) are dedicated to the field of aspect-based sentiment analysis.

Regarding the subjectivity classification task, a new Czech subjectivity dataset and cross-lingual experiments are presented in Přibáň and Steinberger (2022). In Přibáň and Martínek (2018) and Přibáň et al. (2018) we focus on the emotion analysis tasks. A detailed description of our research contributions is presented in Chapter 7 and 8.

# Machine Learning for Sentiment Analysis

# 3

Sentiment Analysis tasks are typically approached as text classification problems. Consequently, in this chapter, we describe machine learning algorithms, models and concepts that are related to SA and classification. We focus on *supervised machine learning*, given its predominant role in SA.

## 3.1 Text Classification

*Text classification* also known as *text categorization* is a fundamental task in NLP. It involves labeling or categorizing text into predefined $n$ classes or categories. Formally, the text classification can be defined as follows: for a given input $x$ and a predefined set of output classes $Y = \{y_1, y_2, \ldots, y_n\}$ the goal is to predict an output class $y \in Y$ (Jurafsky & Martin, 2009).

This definition refers to a type of classification called *multi-class classification* because the input $x$ is classified to exactly one of the $n$ possible classes. *Binary classification* is a special case of multi-class classification in which $Y$ contains only two classes, i.e., $Y = \{y_1, y_2\}$. In contrast, in the *multi-label classification* the predicted output can contain zero or more output predictions $y$ from the set of possible classes $Y$. Supervised machine learning algorithms are usually applied to this problem, but alternative approaches, such as rule-based or unsupervised techniques, can also be applied, see Figure 5.1.

In supervised machine learning, the training data $X = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ of $N$ training examples (e.g., sentences, text documents, reviews, tweets etc.) is given[1]. The goal is to build (train) a model represented by a function $f$ using the training examples from $X$. The mapping function $f$ (model) maps input $x$ to output $y$ and can be rewritten as follows:

$$f : X \longrightarrow Y \tag{3.1}$$

### 3.1.1 Evaluation Metrics

The typical evaluation metrics in text classification encompass *accuracy*, $F_1$ *score* (or *F-measure*), *precision* and *recall*. Let us define some result cases that may occur during clas-

[1] In the multi-label scenario the $y_i$ labels would be replaced with a set.

sification. The tested examples are classified into one of the possible classes[2] and based on the predicted class and the gold label (the actual true class of the example) they can be categorized into four types: (1) *true positive* (*tp*), i.e., positive example was predicted as positive, (2) *false positive* (*fp*), i.e., negative example was predicted as positive, (3) *false negative* (*fn*), i.e., positive example was predicted as negative and (4) *true negative* (*tn*), i.e., negative example was predicted as negative. Accuracy is a metric that summarizes the overall performance of the evaluated model:

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{3.2}$$

For a given class $c$, *precision* is the ratio of the number of correctly classified examples as class $c$ to the total number of examples classified as class $c$. For a given class $c$, *recall* is the ratio of the number of correctly classified examples as class $c$ to the total number of examples that are actually labeled with class $c$. Precision $P$ and recall $R$ for class $c$ are computed as follows:

$$P^c = \frac{tp}{tp + fp} \tag{3.3}$$

$$R^c = \frac{tp}{tp + fn} \tag{3.4}$$

$F_1$ score for class $c$ is computed with precision $P^c$ and recall $R^c$ as their harmonic mean, which is given by:

$$F_1^c = \frac{2 \times P^c \times R^c}{P^c + R^c} \tag{3.5}$$

In multiclass classification, the precision, recall and $F$-measure for each class can be *macro averaged*. The average metrics summarize the overall performance of the model. Macro recall $R^M$ and macro precision $P^M$ are computed as follows:

$$P^M = \frac{\sum_i^n P_i}{n} \tag{3.6}$$

$$R^M = \frac{\sum_i^n R_i}{n} \tag{3.7}$$

where $n$ is a number of classes, $P_i$ and $R_i$ is precision and recall of individual classes. The macro $F$-measure is computed using $P_i$ and $R_i$ with formula 3.5. Usually, in classification, when the recall is improved, the precision drops and vice versa.

[2] Here we consider binary classification, i.e., each example can be classified either as *positive* or *negative* but in general, any number of classes.

## 3.2 Logistic Regression

*Logistic regression* is a supervised classification algorithm (despite the word *regression* in its name). The basic logistic regression can be used for binary classification. The *multinomial logistic regression* allows classification into more classes. Here, we describe the basic version for two classes. We describe this method in relative detail because most of its basic principles, concepts and components are applied in more complex algorithms, concretely in neural networks. Along with logistic regression, we also explain general terms and concepts like *cost function, gradient descent* and *regularization* that are common in machine learning in general.

### 3.2.1 Generative and Discriminative Classifiers

Logistic regression is a type of classifier referred to as a *discriminative* classifier. The classifiers of the second type are called *generative* classifiers[3]. The generative classifier models (*"generate"*) the distribution of individual classes. On the other hand, discriminative algorithms learn a decision boundary between the classes, i.e., they only learn how to distinguish between the classes based on their features.

More formally, both classifiers predict the conditional probability $p(c \mid d)$ of class $c$ given the input document $d$ (technically by features representing the document), but both of them compute the probability differently. Generative models learn to model the joint probability distribution $p(d, c)$ and compute the conditional probability $p(c \mid d)$ to predict the class $c$. Such an example is the Naive Bayes classifier. While discriminative classifiers directly model the conditional probability $p(c \mid d)$ (Jurafsky & Martin, 2009; Ng & Jordan, 2002).

### 3.2.2 Logistic Regression Model

Generally, logistic regression predicts (estimates) the most likely class for the input vector of features $\mathbf{x}^{(i)}$ representing the input document $d_i$ by computing the probability $p(y \mid \mathbf{x}^{(i)})$.

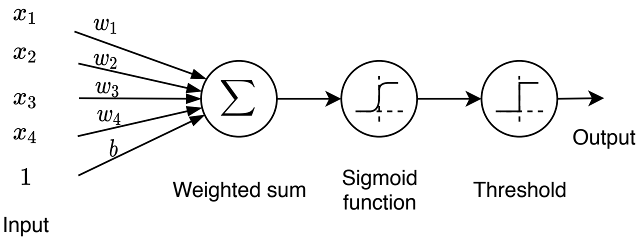


Figure 3.1: Logistic regression model.

[3] Here, for the explanation, we will consider document classification, i.e., the input for any classifiers is a document $d$ and output is its class $c$.

First, let us introduce a notation we follow for a training dataset that consists of the following training examples $\{(\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(m)}, y^{(m)})\}$, where $y^{(i)} \in \{0, 1\}$, $m$ is a size of the training dataset and $\mathbf{x}^{(j)} \in \mathbb{R}^n$ is a vector of features $[x_1, x_2, \dots, x_n]$ thus a particular feature $i$ of training example $\mathbf{x}^{(j)}$ is referred to as $\mathbf{x}_i^{(j)}$. All vectors of training examples can also be written as the matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$ and all labels as a vector $\mathbf{y} \in \mathbb{R}^m$.

The logistic regression model optimizes vector of weights $\mathbf{w}$ and a bias parameter $b$. The model is learning by minimizing an error on training examples. The error is computed by the objective function, e.g., *cross-entropy loss* function. The algorithm that optimizes the parameters according to the objective function is, for example, *stochastic gradient descent* or *gradient descent* (Jurafsky & Martin, 2009).

For input vector $\mathbf{x}$ we want to compute the probability $p(y = 1 \mid \mathbf{x})$, i.e., the input $\mathbf{x}$ belongs to class $y = 1$ which could, for example, indicate that the document is *spam*. Conversely, for $y = 0$, it implies that the input document is *non-spam*. To be exact, the model estimates the probability $\hat{p}(y = 1 \mid \mathbf{x})$ of the true probability $p(y = 1 \mid \mathbf{x})$. The prediction of the classifier on a test example is computed in two steps. First, the $z \in \mathbb{R}$ scalar term is computed from the input $\mathbf{x}$ and model's parameters $\mathbf{w}$ and $b$ in the following way:

$$z = \mathbf{w}^\mathsf{T}\mathbf{x} + b \tag{3.8}$$

Since the $z$ is a real number, we need to convert it into a probability output $\hat{y}$. The $\hat{y}$ represents the estimation of the true $y$. The estimation is achieved by applying the *sigmoid* function as follows:

$$\hat{p}(y = 1 \mid \mathbf{x}) = \hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}} \tag{3.9}$$

The sigmoid function maps the input real number to the range $[0, 1]$, see Figure 3.2. The sigmoid function is also called the *logistic function*, hence the name *logistic regression*.

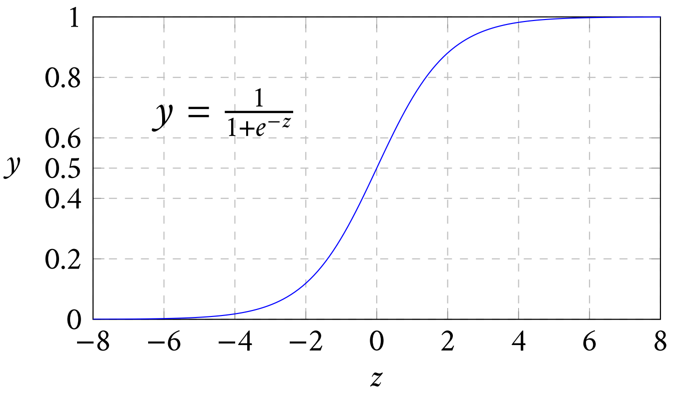


Figure 3.2: Sigmoid function illustration.

As we mentioned, the output of the model is the estimated probability $\hat{p}(y = 1 \mid \mathbf{x})$ of the input $\mathbf{x}$ being assigned to the class $y = 1$ thus for the binary classification the probability $\hat{p}(y = 0 \mid \mathbf{x})$ can be computed as follows

$$\hat{p}(y = 0 \mid \mathbf{x}) = 1 - \hat{p}(y = 1 \mid \mathbf{x}) \tag{3.10}$$

and $\hat{p}(y = 1 \mid \mathbf{x})$ plus $\hat{p}(y = 0 \mid \mathbf{x})$ sum up to one:

$$\hat{p}(y = 1 \mid \mathbf{x}) + \hat{p}(y = 0 \mid \mathbf{x}) = 1 \tag{3.11}$$

Finally, the predicted class $y$ is 1 if the estimated probability $\hat{p}(y = 1 \mid \mathbf{x})$ is greater than threshold 0.5 (which is called the *decision boundary*), 0 otherwise, see Figure 3.1, it can also be written as follows:

$$\text{prediction} = \begin{cases} 1 & \text{if } \hat{p}(y = 1 \mid \mathbf{x}) > 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{3.12}$$

### 3.2.3 Cost Function

In the previous section, we assumed that the parameters $\mathbf{w}$ and $b$ are already optimized. Logistic regression is a supervised machine learning algorithm, so in order to learn its parameters, two components are needed, i.e., the *cost function* and the *optimization algorithm*. The *cost* or *loss* function is a metric that tells us how different the outputs are for the model's training data compared to the true (gold) labels. Logistic regression uses *cross-entropy* cost function. The second component is the optimization algorithm that updates the model's parameters to minimize the cost function. Usually, the *gradient descent* or *stochastic gradient descent* is used as the optimization algorithm (Jurafsky & Martin, 2009).

Given one training example $\mathbf{x}$, the gold label $y$ and the prediction of the model $\hat{y}$ (i.e., the $\hat{p}(y = 1 \mid \mathbf{x})$ probability), the *cross-entropy* loss function $\mathcal{L}_C(\hat{y}, y)$ is defined as follows:

$$\mathcal{L}_C(\hat{y}, y) = -(1 - y)\log(1 - \hat{y}) - y \log \hat{y} \tag{3.13}$$

Using equations 3.9 and 3.8 we can rewrite the cross-entropy loss function as follows:

$$\mathcal{L}_C(y, \mathbf{x}; \mathbf{w}, b) = -(1 - y)\log(1 - \sigma(\mathbf{w}^\mathsf{T}\mathbf{x} + b)) - y \log \sigma(\mathbf{w}^\mathsf{T}\mathbf{x} + b) \tag{3.14}$$

The goal of the optimization algorithm is to optimize the parameters $\mathbf{w}$ and $b$. The previous definition of cross-entropy loss (equations 3.13 and 3.14) is focused only on a single training example. The overall cost function $J(\Theta)$ for the entire training dataset that we want to minimize is defined as the average cross-entropy computed over all training examples:

$$J(\Theta) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}_C(y^{(i)}, \mathbf{x}^{(i)}; \Theta) \tag{3.15}$$

where $\Theta = (\theta_1, \theta_2, \ldots, \theta_n, \theta_{n+1})$ is a vector that represents the model parameters $\mathbf{w}$ and the bias term $b$ that are being optimized. The dimension of $\mathbf{w}$ is $n$.

### 3.2.4 Learning and Gradient Descent

The learning (optimization of the parameters) is then performed by the gradient descent algorithm. Gradient descent computes the gradient of the cost function by computing partial derivative $\frac{\partial J(\Theta)}{\partial \theta_i}$ with respect to each parameter $\theta_i$. Each parameter $\theta_i$ is updated in the following way:

$$\theta_i^{t+1} = \theta_i^t - \alpha \frac{\partial J(\Theta)}{\partial \theta_i} \tag{3.16}$$

where $\theta_i^t$ is the current parameter value, $\theta_i^{t+1}$ is the new parameter value after the update, $\alpha$ is a learning rate and $\frac{\partial J(\Theta)}{\partial \theta_i}$ is the partial derivative of the cost function with respect to the parameter $\theta_i$.

The computed gradient is represented by a vector $\nabla_\Theta J(\Theta)$ where each element corresponds to the element in the vector of parameters $\Theta$ and contains the partial derivative with respect to that parameter. Using the cost function from equation 3.15, the gradient vector $\nabla_\Theta J(\Theta)$ is computed as the average of partial derivatives for each training example:

$$\nabla_\Theta J(\Theta) = \frac{1}{m} \sum_{i=1}^{m} \nabla_\Theta \mathcal{L}_C(y^{(i)}, \mathbf{x}^{(i)}; \Theta) \tag{3.17}$$

Then, one step (update) of the optimization can be rewritten as follows:

$$\Theta^{t+1} = \Theta^t - \alpha \nabla_\Theta J(\Theta) \tag{3.18}$$

Gradient descent is the iterative algorithm that can be stopped when the gradients are smaller than some predefined value $\epsilon$ or when the cost function does not change by a predefined value over the iterations or when the cost function starts to grow on some held-out data (Jurafsky & Martin, 2009). The cross-entropy loss function for logistic regression is convex. Thanks to this property, it is guaranteed that the gradient descent algorithm always finds the (global) minimum.

The described basic version of gradient descent is computationally expensive since to make one iteration (update), the gradient needs to be computed for all training examples. *Stochastic gradient descent* is another version of gradient descent that performs the update of the parameters for each training example $\mathbf{x}^{(i)}$. The gradient $\nabla_\Theta J(\Theta)$ for one training example $\mathbf{x}^{(i)}$ is then computed (technically estimated) as follows:

$$\nabla_\Theta J(\Theta) = \nabla_\Theta \mathcal{L}_C(y^{(i)}, \mathbf{x}^{(i)}; \Theta) \tag{3.19}$$

Another property of stochastic gradient descent is that, to a certain extent, it can be used even for non-convex loss functions (e.g., neural networks) since it can get out of some local optima of the function.

Alternatively, *mini-batch gradient descent* can be applied as well. Instead of computing gradient for only one training example or all training examples, mini-batch gradient descent computes gradient for a batch of $l$ training examples and updates the model's parameters after each batch, as shown below in Equation 3.20.

$$\nabla_{\Theta} J(\Theta) = \frac{1}{l} \sum_{i=1}^{l} \nabla_{\Theta} \mathcal{L}_C(y^{(i)}, \mathbf{x}^{(i)}; \Theta) \tag{3.20}$$

### 3.2.5 Regularization

In order to prevent the model from *overfitting*, a regularization technique is often used. It allows the model to *generalize* on unseen test data (Jurafsky & Martin, 2009). The typical way of implementing the regularization is by adding a new regularization term $\Sigma(\Theta)$ to the cost function. The cost function is then given by:

$$J(\Theta) = \frac{1}{m} \sum_{i=1}^{m} (\mathcal{L}_C(y^{(i)}, \mathbf{x}^{(i)}; \Theta) + \lambda \Sigma(\Theta)) \tag{3.21}$$

where $\lambda$ is a hyper-parameter that controls the strength of the regularization. If $\lambda = 0$ we get the original cost function without any regularization. There are two standard methods for computing the regularization term $\Sigma(\Theta)$. (1) $\ell_1$ *regularization* is the sum of absolute values of the parameters and it is given by:

$$\Sigma(\Theta) = ||\Theta||_1 = \sum_{i=1}^{n+1} |\theta_i| \tag{3.22}$$

(2) The $\ell_2$ *reguralization* is computed as follows:

$$\Sigma(\Theta) = ||\Theta||_2^2 = \sum_{i=1}^{n+1} \theta_i^2 \tag{3.23}$$

### 3.2.6 Multinomial Logistic Regression

Until now, we have described logistic regression for binary classification. For multi-class classification, the *multinomial logistic regression* learns a separate set of parameters $\theta_k \in \Theta$ for each class $c_k \in C$, the number of classes is $l$, i.e., $|C| = l$. Again, the goal is to estimate probability $\hat{p}(y = c \mid \mathbf{x})$ of the true probability $p(y = c \mid \mathbf{x})$ that the input $\mathbf{x}$ belongs to the class $c$. First, vector $\mathbf{z} = [z_1, z_2, \ldots z_l]$ is computed, where each component of the vector is computed[4] from a set of parameters $\theta_k$ for a class $c_k$ and input $\mathbf{x}$ using the equation 3.8. Next, the vector $\mathbf{z}$ is passed through the *softmax* function that produces the estimated probabilities $\hat{y} = \hat{p}(y = c \mid \mathbf{x}; \Theta)$ for each class. The estimated probability $\hat{p}(y = c \mid \mathbf{x}; \Theta)$ for a specific class $c_k$ is computed as follows:

[4] In practice, all these operations are vectorized and parameters are in matrices.

$$\text{softmax}(z_k) = \hat{p}(y = c_k \mid \mathbf{x}; \Theta) = \frac{e^{z_k}}{\sum_{j=1}^{l} e^{z_j}} = \frac{e^{\Theta_k^\top \mathbf{x} + b_k}}{\sum_{j=1}^{l} e^{\Theta_j^\top \mathbf{x} + b_j}} \tag{3.24}$$

After applying the softmax function on the vector $\mathbf{z}$, the output is a vector of probabilities for the input $\mathbf{x}$ assigned to the corresponding classes. The cross-entropy loss function $\mathcal{L}_{CRE}(\hat{y}, y, \mathbf{x})$ for one training example $\mathbf{x}$ is given by:

$$\mathcal{L}_{CRE}(\hat{y}, y, \mathbf{x}) = -\sum_{k=1}^{l} 1\{y = k\} \log \hat{p}(y = k \mid \mathbf{x}; \Theta) \tag{3.25}$$

where $1\{y = k\}$ is equal to 1 if $y = k$, zero otherwise. In other words, it is equal to 1 if the input $\mathbf{x}$ is labeled with the gold class $c_k$.

## 3.3 Neural Networks

We describe *neural networks* and their underlying concepts since, nowadays, neural networks have become the fundamental machine learning tool for NLP. The term *neural* in their name originates from the first proposal of an artificial neuron, called *McCulloch-Pitts neuron* (McCulloch & Pitts, 1943). This artificial neuron was inspired by a simplification of the biological neuron.

In general, neural networks are built from individual units (called neurons) and stacked into layers, collectively forming the entire neural network. Logistic regression and neural networks are closely related since a neural network can be seen as a composition of multiple logistic regression models (or other functions) stacked on top of each other, where the units are the individual logistic regression classifiers. Alternatively, logistic regression can be considered as a simple neural network (Jurafsky & Martin, 2009). In the following sections, we discuss different types of neural network architectures, i.e., *feed-forward neural network*, *recurrent neural network* and *Transformer* architecture.

### 3.3.1 Deep Learning

In recent years, a very popular term *deep learning* has emerged in the context of AI and neural networks. It refers to neural networks with many layers, regardless of the layer type, hence *deep learning*. The fundamental concept behind deep learning is that the model can learn representations of data, i.e., features are extracted by the network itself (automatically) without any explicit or manual feature engineering. For example, neural networks for image recognition or computer vision can contain dozens of layers, where the lower layers identify simpler features, such as edges, while the higher layers recognize more complex features (e.g., parts or even entire objects like digits, letters or faces) based on the outputs of the lower layers. The deep neural networks are built by composing individual layers, which are typically implemented by the *feed-forward neural network*, *recurrent neural network* or *convolutional* *neural network*, although any neural network type can be incorporated in general. Deep learning has proven to be a powerful machine learning technique capable of producing state-of-the-art results not only in NLP but also in various other domains, including computer vision or speech recognition (Goodfellow et al., 2016; Jurafsky & Martin, 2009).

## 3.3.2 Feed-forward Neural Network

*Feed-forward neural network*, also known as *multilayer perceptron* (MLP) is composed of individual layers and each layer is built of individual units (neurons). Feed-forward network consists of one *input layer*, one *output layer* and one or more *hidden layers*, see Figure 3.3. Each neuron from one layer is connected with all neurons in the consecutive layer. Hence, this network architecture is sometimes called *fully-connected*. These connections are called *weights* and they are parameters of the entire network.

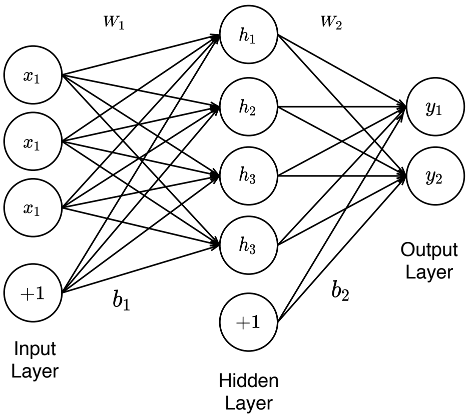


Figure 3.3: Example of feed-forward neural network.

The MLP with one hidden layer (see Figure 3.3) takes as input vector $\mathbf{x}$ and passes it through the entire network, i.e., through each neuron. It can be written as follows:

$$\begin{aligned}\mathbf{h} &= \sigma(\mathbf{W_1 x} + \mathbf{b_1})\\ \hat{\mathbf{y}} &= \text{softmax}(\mathbf{W}_2\mathbf{h} + \mathbf{b_2})\end{aligned} \tag{3.26}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are weight matrices, $\mathbf{b}_1$ and $\mathbf{b}_2$ are bias vectors, $\mathbf{h}$ is an output vector of the hidden layer, $\sigma(\cdot)$ is an activation function and $\hat{\mathbf{y}}$ is a vector of a probability distribution over possible output classes.

First, for each neuron, the weighted sum (scalar) is computed, then the weighted sum is passed through a non-linear function $\sigma(\cdot)$ that is called the *activation function*. An example of the activation function is the sigmoid function (used in logistic regression) but other functions like ReLU (Rectified Linear Unit), see equation 3.28, or hyperbolic tangent, see equation 3.27, can be used as well. Next, the output vector $\mathbf{h}$ of the hidden layer is passed through[5] the

[5] In case of more than one layer, the output is passed into the next hidden layer instead.

softmax function that produces the output vector of probabilities $\hat{\mathbf{y}}$. The sequence of these operations is called *forward propagation*.

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.27}$$

$$ReLU(x) = \max(0, x) \tag{3.28}$$

Since the MLP can generally contain more than one layer, we can rewrite the expression 3.26 representing the network in Figure 3.3 using the common notation (Jurafsky & Martin, 2009):

$$\begin{aligned}
\mathbf{z}^{(1)} &= \mathbf{W}^{(1)}\mathbf{a}^{(0)} + \mathbf{b}^{(1)} \\
\mathbf{a}^{(1)} &= g^{(1)}(\mathbf{z}^{(1)}) \\
\mathbf{z}^{(2)} &= \mathbf{W}^{(2)}\mathbf{a}^{(1)} + \mathbf{b}^{(2)} \\
\mathbf{a}^{(2)} &= g^{(2)}(\mathbf{z}^{(2)}) \\
\hat{\mathbf{y}} &= \mathbf{a}^{(2)}
\end{aligned} \tag{3.29}$$

where the number in superscript refers to the $n$-th hidden layer starting from 1 and specifically the layer 0 means input, so $\mathbf{a}^{(0)}$ is the input vector $\mathbf{x}$, $g^{(i)}(\cdot)$ is the activation function in the $i$-ith layer, $\mathbf{a}^{(i)}$ is an output of the $i$-ith layer, $\mathbf{z}^{(1)}$ is the weighted sum in the $i$-ith layer and $\hat{\mathbf{y}}$ is the predicted output probability distribution. The activation function $g^{(2)}$ in the last (second) layer represents the softmax function in equation 3.26.

### 3.3.3 Learning of Neural Networks

As in the case of logistic regression, the goal is to optimize the parameters $\mathbf{W}^{(i)}$ and $\mathbf{b}^{(i)}$ in a way that the outputs $\hat{\mathbf{y}}$ for training data produced by the model are similar as much as possible to the true labels $\mathbf{y}$. The learning is done by the same *gradient descent* algorithm that was described in Section 3.2.4 and by the *back-propagation* algorithm (Rumelhart et al., 1986). The same cross-entropy loss function (as for logistic regression, see equation 3.25) can be rewritten more comprehensively as follows:

$$\mathcal{L}_{CRE}(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_{i=1}^{l} y_i \log \hat{y}_i \tag{3.30}$$

where $l$ is number of classes, $y_i$ is the gold label for the class $i$ and $\hat{y}_i$ is the prediction of the model for the class $i$.

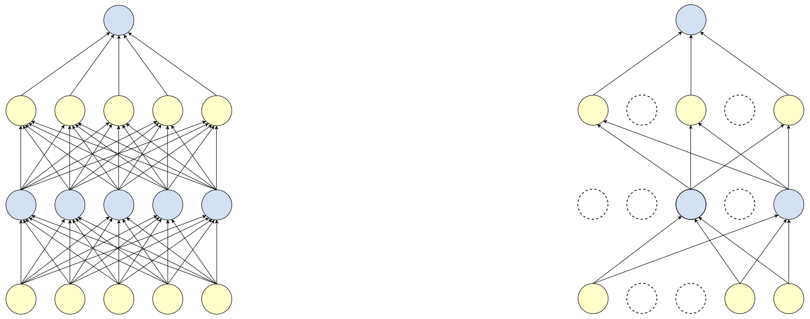
With the growing number of layers and parameters, the computation of the gradient becomes a complex and non-trivial task. The *back-propagation* algorithm (Rumelhart et al., 1986) allows computing the gradient. The back-propagation relies on the *chain rule*, given a composite function $f(x) = g(u(w(x)))$ the derivative $\frac{df}{dx}$ of the function $f(x)$ with respect to $x$ is computed in a following way:

$$\frac{df}{dx} = \frac{dg}{du} \cdot \frac{du}{dw} \cdot \frac{dw}{dx} \tag{3.31}$$

Using the chain rule, the back-propagation algorithm computes the gradient composed of the partial derivative of the cost function with respect to each parameter of the model. The computed gradient is used by gradient descent to update the parameters of the model as described in 3.2.4.

### 3.3.4 Dropout

We introduced the technique of regularization in Section 3.2.5. Another approach used to prevent the overfitting of neural networks is called *dropout* (Srivastava et al., 2014). This technique randomly drops units (neurons) and their connections to a selected layer of the network. In other words, the connections of units between two layers, i.e., weights, are not used at a given learning step (when dropout is applied), see Figure 3.4. The choice of which units to drop is random and given by probability $p$. Dropout is applied only during the training of the neural network and disabled when test predictions are being made.



(a) Standard Neural Network.

(b) Neural Network after applying dropout.

Figure 3.4: Illustration of the dropout technique. Left: (a) standard neural network with two hidden layers. Right: (b) An example of a thinned network produced by applying dropout to the network on the left. Empty units have been dropped. Inspired by Srivastava et al. (2014).

## 3.4 Recurrent Neural Network

The Recurrent Neural Network (RNN) (Elman, 1990) is intended for processing of sequential data. Text is sequential in nature. RNN allows processing sequences of different lengths, unlike the feed-forward neural network, where the input is always fixed-size. RNN also allows *"remember"* information from the previous steps of the processed sequence because RNN takes as input not only the current input but also a hidden state of the network from the previous step, as shown in Figure 3.5.

More formally, RNN processes the input sequence $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2 \ldots \mathbf{x}_T]$ and for each element $\mathbf{x}_t$ at time step $t$ it computes new hidden state $\mathbf{h}_t$ from the input $\mathbf{x}_t$ and the previous hidden state $\mathbf{h}_{t-1}$. The new hidden state $\mathbf{h}_t$ is computed by hidden layer function $\mathcal{H}$:

$$\mathbf{h}_t = \mathcal{H}(\mathbf{x}_t, \mathbf{h}_{t-1}) \tag{3.32}$$

In the simplest case, the hidden layer function $\mathcal{H}$ is defined as:

$$\mathbf{h}_t = \sigma\left(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h\right) \tag{3.33}$$

where the $\mathbf{W}$ terms correspond to weight matrices (e.g., $\mathbf{W}_{xh}$ is the input-hidden weight matrix) and $\mathbf{b}_h$ term is hidden bias vector. The concrete implementation of the $\mathcal{H}$ function depends on the type of the used RNN unit (Graves et al., 2013), for example, Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997) or Gated Recurrent Unit (GRU) (Cho, van Merriënboer, Gulcehre, et al., 2014). Each RNN unit shares the parameters (weights) across all time steps.

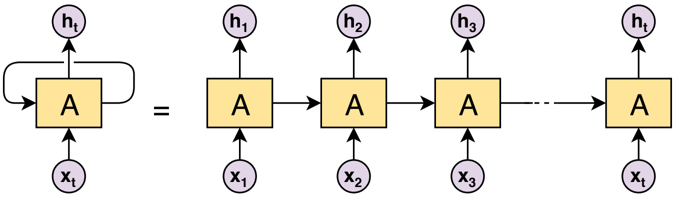


Figure 3.5: Basic RNN architecture[6].

The common practice is also to use *bidirectional* RNN (BiRNN) (Schuster & Paliwal, 1997). BiRNN processes the sequence in both directions, which has shown to be beneficial because the output at time $t$ can depend on the previous and future elements of the sequence. It is usually implemented with two RNN units, where one processes the sequence in the original order and the second RNN processes the sequence in reverse order. The outputs $\overrightarrow{\mathbf{h}}_t$ (for the original sequence direction, i.e., left to right) and $\overleftarrow{\mathbf{h}}_t$ (for the reversed direction) of these two RNN units are usually concatenated and producing one output $\mathbf{h}_t$ as follows:

$$\mathbf{h}_t = [\overrightarrow{\mathbf{h}}_t; \overleftarrow{\mathbf{h}}_t] \tag{3.34}$$

The disadvantage of BiRNN is that the entire input sequence must be present when it is being processed, which can be problematic for some specific tasks. An example of BiRNN is a bidirectional LSTM (BiLSTM) (Graves & Schmidhuber, 2005).

There is one common issue with the simplest RNN implementation (described above) called *vanishing* or *exploding gradients*. When RNN processes longer sequences during the training, the weights inside the RNN are multiplied in each time step. In the back-propagation step, there is also a large amount of multiplication and thanks to that, the gradients either *"explodes"* (become very large) or *"vanishes"* (become very small) and thus the model is not able to learn, i.e., instead of converging it diverges.

[6] Image is based on http://colah.github.io/posts/2015-08-Understanding-LSTMs.

### 3.4.1 Long Short-Term Memory

Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997) is an implementation of the RNN capable and intentionally designed to *"remember"* or *"store"* some long-term dependency information from the previous time steps. The architecture of LSTM is shown in Figure 3.6.

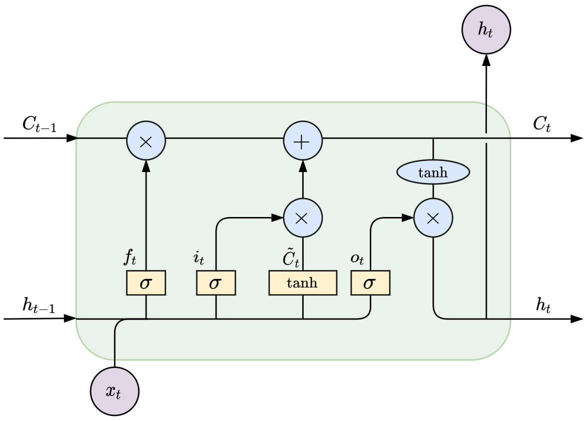


Figure 3.6: LSTM architecture visualisation[7].

The LSTM unit (or *cell*) is composed of structures called *gates* that can decide which information should be stored and which information should be deleted. The gates take the input $\mathbf{x}_t$ and previous hidden state $\mathbf{h}_{t-1}$ and produce output that is a part of the hidden state. In addition, the LSTM also takes the cell state $\mathbf{C}_{t-1}$ from the previous time step. More concretely, each LSTM unit contains *input*, *forget* and *output* gates. Using the gates, the input $\mathbf{x}_t$, previous hidden state $\mathbf{h}_{t-1}$ and previous cell state $\mathbf{C}_{t-1}$, the LSTM produces new hidden state $\mathbf{h}_t$ and new cell state $\mathbf{C}_t$. The entire model of LSTM can be written as follows:

$$\begin{aligned}
\mathbf{f}_t &= \sigma\left(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{b}_f\right) \\
\mathbf{i}_t &= \sigma\left(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{b}_i\right) \\
\mathbf{o}_t &= \sigma\left(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{b}_o\right) \\
\tilde{\mathbf{C}}_t &= \tanh\left(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c\right) \\
\mathbf{C}_t &= \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{C}}_t \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh\left(\mathbf{C}_t\right)
\end{aligned} \tag{3.35}$$

where $\odot$ is element-wise multiplication, $\mathbf{x}_t$ is the current input vector, $\mathbf{W}$ terms correspond to weight matrices and $\mathbf{b}$ terms are bias vectors, $\mathbf{i}_t$, $\mathbf{f}_t$, $\mathbf{o}_t$ are the outputs of the input, forget and output gates, respectively, $\tilde{\mathbf{C}}_t$ is the vector of new candidate information that can be added and $\sigma$ is the sigmoid function.

[7] Image based is on https://colah.github.io/posts/2015-08-Understanding-LSTMs.

The forget gate $\mathbf{f}_t$ decides what information will be removed from the cell state. It produces values between 0 and 1 multiplied by the values from $\mathbf{C}_{t-1}$. The amount of information that will be deleted is controlled by the output where 0 means forgot (drop, delete) the entire information from the previous cell state $\mathbf{C}_{t-1}$ and 1 do not delete anything.

Next, the input gate $\mathbf{i}_t$ decides which values will be updated and $\tilde{\mathbf{C}}_t$ computes the new candidates' values (information) that could be potentially added. These two vectors are then point-wise multiplied together and the result is summed with $\mathbf{f}_t \odot \mathbf{C}_{t-1}$ that gives the new cell state $\mathbf{C}_t$. Finally, the new hidden state $\mathbf{h}_t$ is computed from the output gate $\mathbf{o}_t$ and the new cell state $\mathbf{C}_t$. A slightly simplified variant of LSTM is called Gated Recurrent Unit (Cho, van Merriënboer, Gulcehre, et al., 2014).

## 3.5 Sequence to Sequence

Specific neural network architectures can also be used to model and solve *sequence* problems (Sutskever et al., 2014). Sequence problems can be divided into four categories: *One-to-One*, *One-to-Many*, *Many-to-One* and *Many-to-Many*, as shown in Figure 3.7. The *many-to-many* sequence problem is also called *sequence to sequence* or *seq2seq*.

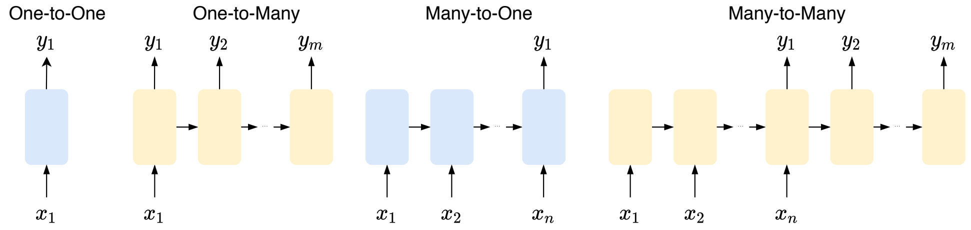


Figure 3.7: Sequence problems visualization[8].

We summarize the four categories as follows:

1. **One-to-One:** The *one-to-one* problem can be seen as a special case of sequence problem, where there is only a single input and output[9] with a fixed size. An example is image classification, where the input is always an image with a fixed size and the output is one category. Since text is in nature sequential, there are not many one-to-one examples in NLP. A common text classification problem could be potentially considered as one-to-one because the problem used to be treated as a one-to-one problem. The reason is that the input document (sequence) was transformed into one feature vector with a fixed size regardless of the length of the document and then the document was classified into one category.

[8] Image is based on http://karpathy.github.io/2015/05/21/rnn-effectiveness.

[9] By one input we mean one vector of features.

2. **One-to-Many:** There is only one input and the output is a sequence. For example, the input can be one word representing some topic and the output a sequence of words about the topic.

3. **Many-to-One:** In many-to-one problems, there is only a single output for the input sequence. Nowadays, a typical example is text classification, for example, the input is a sequence of words and the output is a class label.

4. **Many-to-Many:** Many-to-many or sequence-to-sequence problems consist of sequence input and sequence output, the lengths of the input and output sequences may differ. The most typical example is machine translation, where the input can be a sentence in Czech and the output is its translation in English.

### 3.5.1 Encoder-decoder

The *many-to-one* and *many-to-many* problems are the most common in NLP. The common practice for modeling the *many-to-many* sequence problems in NLP is to use the *encoder-decoder* architecture (Cho, van Merriënboer, Gulcehre, et al., 2014; Sutskever et al., 2014), see Figure 3.8 for basic visualization and Figure 3.9 for machine translation example implemented by RNN.

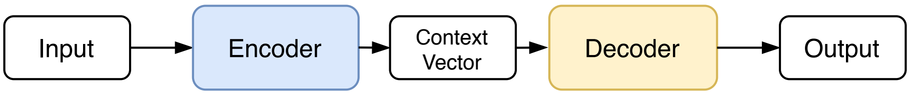


Figure 3.8: Basic visualization of the encoder-decoder architecture.

The *encoder* encodes the variable-length input sequence into the fixed-length vector representation **C** (the inner state representing the input, also called context vector) and the *decoder* decodes the fixed-length vector representation **C** and generates the output, see Cho, van Merriënboer, Gulcehre, et al. (2014) for more detailed mathematical description. The outputs of the encoder are discarded.

The encoder-decoder architecture is usually implemented by RNN (see Section 3.4) or by the Transformer model (see Section 3.6). Typically, the encoder and decoder are implemented by the same type of neural network. Both the encoder and decoder can be composed of multiple stacked layers of neural networks.

The simplest solution for obtaining the context vector **C** is to use the last hidden state $\mathbf{h}_n^e$ of the encoder and the context vector **C** is then used as the initial hidden state $\mathbf{h}_0^d$ of the decoder (Jurafsky & Martin, 2009), that can be written as follows:

$$\begin{aligned} \mathbf{C} &= \mathbf{h}_n^e \\ \mathbf{h}_0^d &= \mathbf{C} \end{aligned} \tag{3.36}$$

During the generation of the output sequence, the hidden state $\mathbf{h}_t^d$ and the output probability distribution $\mathbf{y}_t$ at time step $t$ is given by:

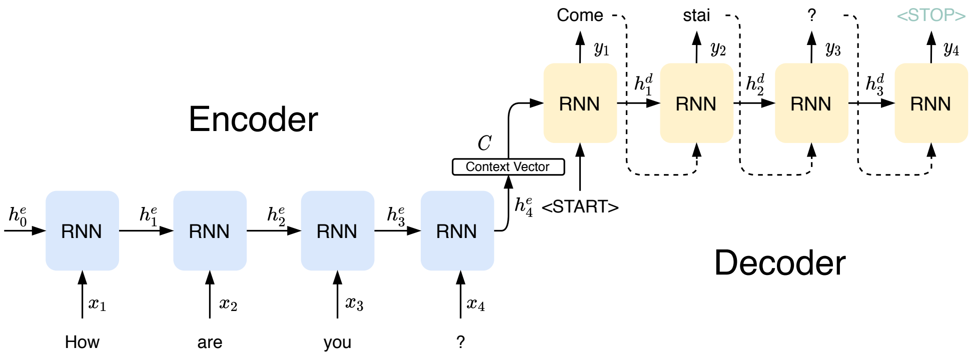


Figure 3.9: The example of the encoder-decoder architecture for machine translation. The input sentence is encoded into context vector **C**. The output sentence is generated until the end of sentence tag <STOP> is generated.

$$\begin{aligned} \mathbf{h}_t^d &= \mathcal{H}(\mathbf{y}_{t-1}, \mathbf{h}_{t-1}^d) \\ \mathbf{y}_t &= \text{softmax}(\mathbf{h}_t^d) \end{aligned} \tag{3.37}$$

where the function $\mathcal{H}$ represents the RNN cell. Eventually, the generation at each time step $t$ can be conditioned by the context vector **C** and the previous output $\mathbf{y}_{t-1}$ as follows:

$$\begin{aligned} \mathbf{h}_t^d &= \mathcal{H}(\mathbf{y}_{t-1}, \mathbf{h}_{t-1}^d, \text{C}) \\ \mathbf{y}_t &= \text{softmax}(\mathbf{h}_t^d, \mathbf{y}_{t-1}, \mathbf{C}) \end{aligned} \tag{3.38}$$

### 3.5.2 Attention Mechanism

The vanilla approach of the encoder-decoder architecture uses the hidden state of the last unit (the last RNN cell in the encoder in Figure 3.9) as the context vector **C**. This solution is not optimal since the context vector is the last hidden state $\mathbf{h}_n$ of the encoder and the decoder is forced to produce the output using only the last state. Thus, the last state must contain all necessary information about the entire sequence. It is problematic in the case of long dependencies where the information from the beginning of the sequence can fade away, but it can be important to produce the output at the end of the sequence.

The *attention mechanism* (Bahdanau et al., 2015) allows modeling the long dependencies without regard to their distance in the input or output sequences (Vaswani et al., 2017). The attention mechanism allows the decoder to use all hidden states of the encoder and also learn their importance (i.e., *"pay attention"*) in order to produce the output at the current time step.

## 3.6 Transformer

A relatively recent type of neural network architecture is called *Transformer*, originally introduced in Vaswani et al. (2017), where its effectiveness was demonstrated on machine translation. The Transformer is an example of the encoder-decoder architecture, as illustrated in Figure 3.10. Most of the recent *generalized language models* like BERT or GPT-2 (see Section 4.1.5) leverage the Transformer as a basic building block. These models have consistently achieved state-of-the-art results in a variety of NLP tasks, thereby establishing the Transformer architecture as the de facto standard for NLP applications. The Transformer architecture is shown in Figure 3.11. The Transformer is able to handle long dependencies in sequences using the attention mechanism without any RNN. Furthermore, the Transformer's design offers the advantage of easy parallelization, in stark contrast to architectures like LSTM.

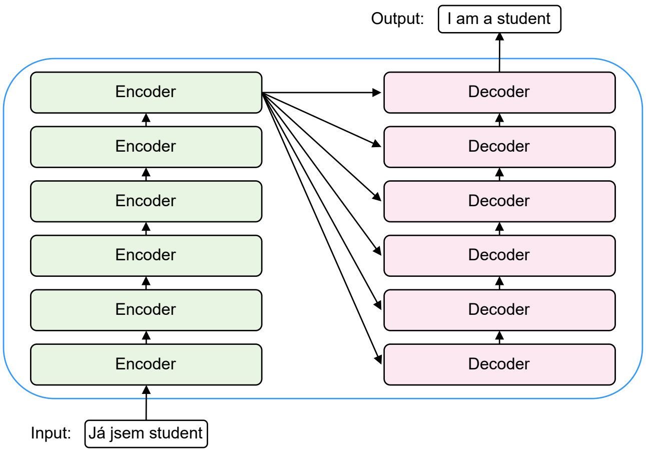


Figure 3.10: Visualization of stacked layers in the Transformer architecture[10].

The reason behind the rise of the Transformer architecture and its success lies in the *self-attention* mechanism (see Section 3.6.2) and the Transformer's ability to be pre-trained and consequentially fine-tuned. Typically, the Transformer is pre-trained via a language modeling task with a tremendous quantity of unlabeled text. Thanks to the pre-training, the model acquires a general knowledge of the language (e.g., syntax and semantics). The knowledge acquired during pre-training is then utilized in the fine-tuning phase of a specific NLP task. Thanks to prior knowledge, Transformer consistently outperforms other models and exhibits superior performance compared to models lacking the pre-training phase. This strategy is commonly known as *transfer learning* and it is discussed in greater detail in Chapter 4. Here, we describe the common underlying principles and the Transformer architecture.

### 3.6.1 Transformer Architecture

The Transformer architecture follows the encoder-decoder principle. For an input sequence $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$, the encoder produces a continuous representation $\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n)$. Then, the decoder generates an output sequence $(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m)$.

The encoder (left part of Figure 3.11) of the Transformer consists of $N$ identical layers[11] that are stacked on each other ($N$ can be seen as a hyper-parameter of the model) as shown in Figure 3.10. Each layer is composed of another two sub-layers. The first sub-layer is a multi-head self-attention mechanism and the second sub-layer is a fully connected feed-forward neural network. The output of each sub-layer is added to a residual connection (He et al., 2016) vector and a layer normalization (Ba et al., 2016) is performed. All sub-layers in the model and also the input embedding layers produce outputs with a dimension $d_{\text{model}} = 512$.

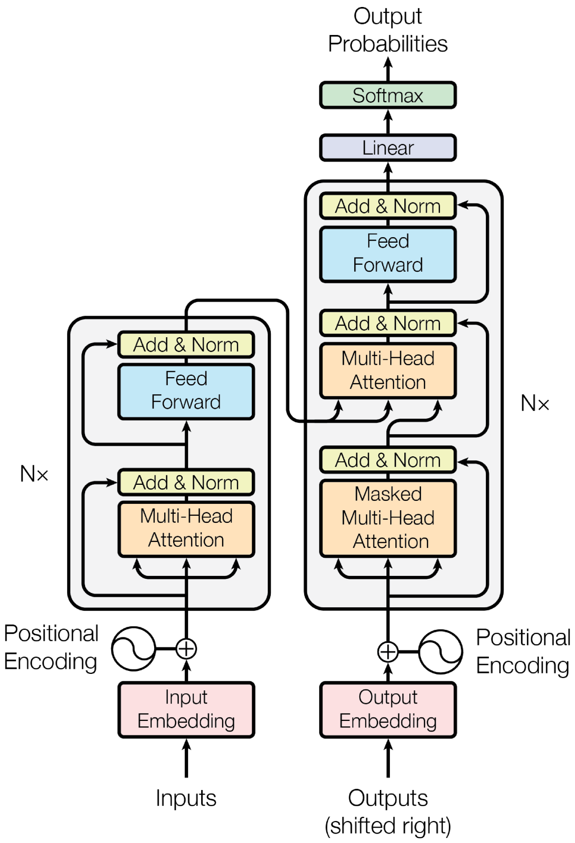


Figure 3.11: The architecture of the Transformer model. The left part represents the *encoder* and the right part represents the *decoder*. Image is taken from Vaswani et al. (2017).

The decoder (right part of Figure 3.11) of the Transformer also contains $N$ stacked layers identical to the encoder part besides the two following modifications. (1) One extra sub-layer is added and it performs multi-head attention over the output of the encoder. (2) The first sub-layer (masked multi-head attention) is a modified self-attention mechanism to ensure that the model predictions for position $i$ depend only on the previous known outputs.

In each Transformer layer block, the fully connected feed-forward network is applied to each sequence position separately. In each layer, the feed-forward neural network has its own parameters. It consists of two linear transformations with ReLU activation function between them. The feed-forward layer $\text{FFN}(x)$ is given by:

[11] In the original Transformer paper (Vaswani et al., 2017) the authors used $N = 6$.

$$\mathrm{FFN}(\mathbf{x}) = \max(0, \mathbf{x}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2 \tag{3.39}$$

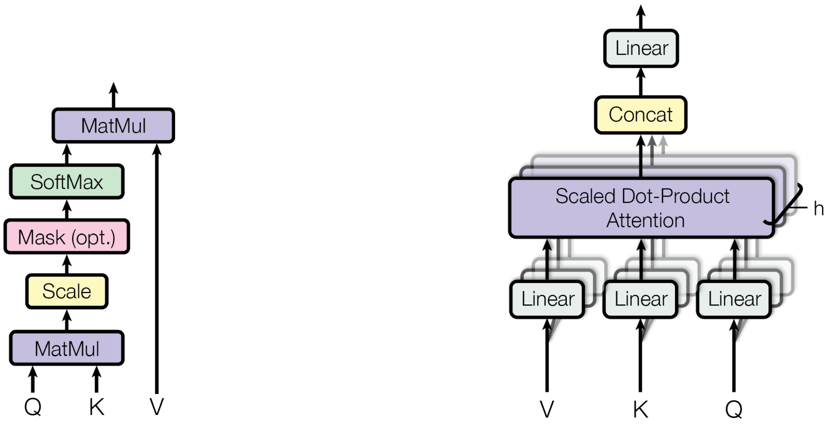
The input sequence of the entire model is firstly used as an input for the two embedding layers that produce vector representations for the tokens, the two embedding layers share parameters. There is no information about the token position in the produced vector representation. Thus, the position is encoded and added to this vector. Each position is encoded with sine and cosine functions, the produced positional encoding has the same dimension as the $d_{\text{model}}$.

The output of the final decoder layer is passed to the learned linear transformation layer and then the softmax function is used in order to produce the output tokens probabilities.

As we mentioned, the original Transformer architecture is composed of encoder and decoder parts. However, not every model that is based on Transformer architecture uses both of these components. Some use only the encoder or decoder parts. We discuss these differences and describe the underlying models in Section 4.3.

### 3.6.2 Self-attention

*Self-attention* or *intra-attention* is an attention mechanism relating different positions of a single sequence to compute a representation of the same sequence (Vaswani et al., 2017). It allows the model to capture dependencies and correlations between the current output at time step $t$ and the other parts of the sequence. In other words, the model is able to attend to different parts of the input sequence to better learn long-range dependencies between tokens.



(a) Scaled Dot-Product Attention visualization.

(b) Multi-head attention visualization that consists of several attention layers running in parallel.

Figure 3.12: Attention visualization. Images are taken from Vaswani et al. (2017).

The attention mechanism used in the Transformer is called *Scaled Dot-Product Attention*, see Figure 3.12a. This attention takes as an input *query* and *key* vectors of dimension $d_k$ a *values* vector of dimension $d_v$. Firstly, the dot product of the query with all keys (i.e., size of the input sequence) is computed (the *MatMul* part in Figure 3.12a) and each of the computed dot products is divided (scaled) by $\sqrt{d_k}$. Next, the softmax function is applied[12]. The result is then multiplied with the *values* vector using the dot product.

In reality, the queries are stacked into a matrix $\mathbf{Q}$, the keys and values are also stacked into matrix $\mathbf{K}$ and $\mathbf{V}$, respectively. The computation is then done by matrix multiplication and the matrix of outputs can be written as follows:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \tag{3.40}$$

### 3.6.3 Multi-head Attention

Instead of computing the attention only once, multiple parallel attentions can be employed. In this approach, the attention is computed $h$ times with separate parameters for each individual attention, where $h$ denotes the number of heads (i.e., number of parallel attentions), see Figure 3.12b. The benefit of multi-head attention is that it gives an opportunity to the model to learn different types of information. For example, one attention head may specialize in capturing syntactic patterns, while another may focus on semantic content[13]. The outputs of the individual attention heads are concatenated, linearly transformed and the result is passed to higher layers. The multi-head attention mechanism can be written as follows:

$$\begin{aligned} \text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) &= \text{Concat}(\text{head}_1, \text{head}_2, \dots, \text{head}_h)\mathbf{W}^O \\ &= [\text{head}_1, \text{head}_2, \dots, \text{head}_h]\mathbf{W}^O \end{aligned} \tag{3.41}$$

where $\mathbf{W}^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$ is the matrix with parameters for the final linear transformation. The head $\text{head}_i$ is given by:

$$\text{head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V) \tag{3.42}$$

where $\mathbf{W}_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $\mathbf{W}_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $\mathbf{W}_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ are matrices with parameters corresponding to each head. The authors of Vaswani et al. (2017) used $h = 8$ and $d_k = d_v = d_{\text{model}}/h = 64$.

### 3.6.4 Tokenization

The raw text cannot be processed directly by the Transformer model. Firstly, it has to be converted into a form the model can process. Such a process is called *tokenization*. Tokenization is the procedure of transforming text into a sequence of individual units, such as words or subwords, and assigning a unique numerical identifier to each unit (Jurafsky & Martin, 2009) as shown in Figure 3.13[14]. The process results in a sequence of numbers. The Transformer

[12] In the case of decoder, the masking is also applied, see Vaswani et al. (2017) for details.

[13] This example is just to give you an idea; in reality, it does not have to be so clear.

[14] The examples are inspired by https://huggingface.co/docs/transformers/tokenizer_summary.

model further processes the sequence by converting each number into a word vector (also called word embedding).

Input: Don’t you like the sea? The happiest fish live in sea.

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Output: | don’t | you | like | the | sea? | the | happiest | fish | live | in | sea. |

Figure 3.13: An example of tokenization by space.

The issue associated with simple tokenization, where complete words are considered as tokens, is the potential explosion in vocabulary size, which can result in the vocabulary containing hundreds of thousands or even millions of unique entries. Since each token is mapped to a word vector, the resulting word embeddings matrix would require a tremendous amount of memory. As an alternative, the smaller parts of words are used as tokens, which are called *subwords* as shown in Figure 3.14. Subword tokenizers typically operate with a more manageable vocabulary size, usually in the order of tens of thousands. For example, the BERT (Devlin et al., 2019) model has a vocabulary size of 30k.

Input: Don’t you like the sea? The happiest fish live in sea.

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Output: | do | n’t | you | like | the | sea | ? | the | ha | ##pp | ##iest | fish | live | in | sea | . |

Figure 3.14: An example of subword tokenizer.

The idea of subword tokenizers is to tokenize the most frequent words as single tokens and rare words are tokenized into subwords with some particular meaning. For example, consider the word “*unhappiness*”. A subword tokenizer may break it down into “*un-*”, “*happi-*”, and “*-ness*” as subword units, which can provide more flexibility in handling similar words like “*happiness*” or “*unhappy*”. This approach can effectively handle languages with complex morphology, rare, long or unknown words, and out-of-vocabulary (OOV) words. This process allows the tokenizer to create subword units that capture common word fragments or recurring patterns in the data. Subword tokenizers such as Byte-Pair Encoding or WordPiece are nowadays a standard for Transformer-based models. The subword tokenizer is usually built specifically for each model. The tokenizer is trained on a raw training corpus (Jurafsky & Martin, 2024), during this process, the vocabulary is built.

### 3.6.4.1 Byte-Pair Encoding

The *byte-pair encoding* (BPE) algorithm (Sennrich et al., 2016) usually expects that the training corpus is already pre-tokenized into words. The pre-tokenization can be done simply by splitting the text by spaces. The algorithm starts with a vocabulary that contains all individual characters present in a given corpus. Then, it iterates over the training corpus and chooses the two most common adjacent characters (subwords), merges them and adds them to the vocabulary, creating a new subword and merging rule. Additionally, the algorithm replaces each occurrence of the selected adjacent characters (subwords) in the corpus with the new subword.

This process of creating and merging new subwords continues until a predefined number $k$ of new subwords has been formed, where $k$ is a hyper-parameter of the tokenizer. At the end of the training, the vocabulary contains the initially used characters and all newly created subwords. When any text is to be tokenized, the tokenizer just applies the merging rules in the order in which they were created.

#### 3.6.4.2 WordPiece

The *WordPiece* tokenization algorithm (Schuster & Nakajima, 2012) is similar to the BPE tokenizer. It also uses every character from the training corpus to create an initial vocabulary. The difference lies in the choice of the pair of subwords (characters) that are merged. Instead of choosing the most frequent subword pair, the algorithm selects the pair that maximizes the likelihood of the training data when added to the vocabulary.

#### 3.6.4.3 Other Subword Tokenizers

Another subword tokenization algorithm is called *unigram* (Kudo, 2018). In contrast to the BPE and WordPiece, the initial vocabulary of the unigram algorithm is set to a large number of subwords, which can be the words split by space. The algorithm then iteratively reduces the vocabulary. The *SentencePiece* algorithm (Kudo & Richardson, 2018) mitigates the problem of the previous algorithms that assumed that the input is already pre-tokenized, usually by spaces. SentencePiece uses the input as a raw input stream including spaces.

# Transfer Learning 4

*Transfer learning* is a technique used in machine learning, especially in computer vision and NLP and it has proven to be highly effective. The core idea is to pre-train a model on a large volume of data and then adapt this pre-trained model to a specific task or domain, effectively transferring and leveraging the acquired knowledge. The detailed description and categorization of transfer learning techniques can be found in Pan and Yang (2010) and Zhuang et al. (2020).

The motivation for this technique is that for most supervised NLP tasks, the labeled data are limited (insufficient amount of training data). Transfer learning allows the utilization of general knowledge acquired during pre-training, leading to improved performance on tasks with limited data compared to approaches reliant solely on that limited data. This transferred knowledge often encompasses insights and patterns that would be challenging to learn from the limited data available for a specific task. The pre-trained models for transfer learning can be split according to their usage in downstream tasks into two groups (Devlin et al., 2019):

1. *Feature-based approach*: The pre-trained model produces vector representations of text as additional features for another custom model for a specific task. The word2vec (Mikolov, Chen, et al., 2013), fastText (Bojanowski et al., 2017) word embeddings or ELMo (Peters et al., 2018) architecture belongs under this category.

2. *Fine-tuning approach*: The pre-trained model (its parameters) is directly fine-tuned on the downstream task and no additional model is needed. Examples of this approach are BERT (Devlin et al., 2019) or GPT (Radford et al., 2018).

The feature-based approach is older, but it has shown to be very efficient. With the rise of the Transformers, the fine-tuning approach nowadays become a de facto standard. We describe the fine-tuning approach for classification with the Transformer-based models in Section 4.3.6. Very recently, a new technique called *prompt-based* learning emerged, see Section 4.3.7.

## 4.1 Word Embeddings

To use machine learning algorithms, it is necessary to convert text into numerical vectors, as these algorithms work with numerical inputs. These vectors should effectively capture both the syntax and semantic information.

Word embeddings, sometimes called word vectors or simply embeddings, have emerged as a pivotal component for various NLP tasks. They have become an essential part of many NLP systems since they can capture words' meaning (semantics). Probably the most famous methods for learning word embeddings are word2vec (Mikolov, Chen, et al., 2013), GloVe (Pennington et al., 2014) and fastText (Bojanowski et al., 2017). They are based on the *Distributional Hypothesis* (Harris, 1954) that says that words that occur in similar contexts tend to have similar meanings. It was popularized by (Firth, 1957) and his famous quote, *"a word is characterized by the company it keeps"*.

The word embeddings can be divided into two groups *static word embeddings* and *contextual word embeddings* also called *dynamic word embeddings*. Static word embeddings always assign the same vector to each word regardless of the context in which the word occurs. In contrast, contextual word embeddings generate a new word vector based on the context in which the word appears.

### 4.1.1 Static Word Embeddings

Static word embeddings such as word2vec, GloVe or fastText are typically pre-trained and represented by an $n$-dimensional vector space, also called semantic space. Each word $w_i$ from vocabulary $V$ is represented by a static vector $\mathbf{h}_i \in \mathbb{R}^n$ (Hewitt, 2019). They are usually stored in one matrix, used as a lookup table that maps words to vectors. It can be expressed as a mapping function:

$$f_{vocabulary} : w_i \longrightarrow \mathbf{h}_i \tag{4.1}$$

The disadvantage of static word embeddings is that they cannot handle polysemy. In other words, one word can have multiple meanings and the specific interpretation relies on the context in which the word is used. For example, consider these two sentences: *"I love Coca-Cola in the new can"* and *"I can buy Coca-Cola for you tonight"*. In both sentences, the word *"can"* is used, but each time, depending on the context, it means something different and using the static word embeddings, it will always be represented by the same vector. In contrast, the *contextual word embeddings* can handle different contexts, see Section 4.1.5.

### 4.1.2 Similarity Between Word Vectors

The similarity between word vectors is usually measured with cosine similarity, which corresponds to the cosine of the angle $\alpha$ between the two vectors $\mathbf{x}$ and $\mathbf{y}$ with dimension $d$ and it is computed as follows:

$$\cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{||\mathbf{x}|| \, ||\mathbf{y}||} = \frac{\sum_{i=1}^{d} x_i y_i}{\sqrt{\sum_{i=1}^{d} x_i^2} \sqrt{\sum_{i=1}^{d} y_i^2}} = \cos(\alpha) \tag{4.2}$$

### 4.1.3 Word2vec

The famous *word2vec* is a pair of two models for efficient learning of word embeddings, i.e., *continuous bag-of-words* (CBOW) and *Skip-gram* proposed by Mikolov, Chen, et al. (2013). Both word2vec models (CBOW and Skip-gram) are actually neural networks[1] with three layers: *input layer*, *projection layer* and *output layer*, see Figure 4.1. The models are inspired by the feed-forward neural network for language modeling (NNLM) proposed in Bengio et al. (2003). The proposed NNLM consists of input, projection, hidden and output layers, but the network is also computationally expensive, which is caused by the non-linear hidden layer. In word2vec, the hidden layer is removed and thus, the proposed models are computationally less expensive.

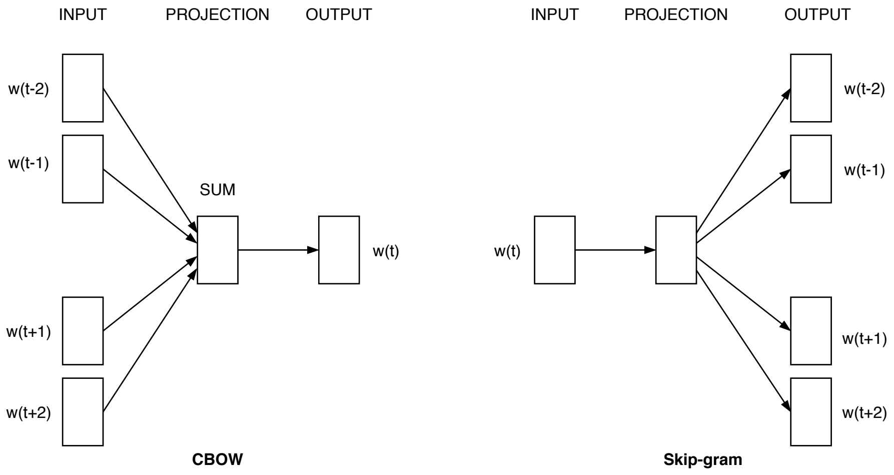


Figure 4.1: The CBOW architecture predicts the current word based on the context and the Skip-gram predicts surrounding words given the current word.

#### 4.1.3.1 Skip-gram with Negative Sampling

Skip-gram model learns to predict the surrounding context words within a certain range $C$ (context size) before and after the current word $w_t$ as shown on the right side of Figure 4.1. The model's input is only the current word $w_t$ encoded with a one-hot vector and the output

[1] Sometimes word2vec is not considered as a neural network because of the removed non-linearity that is so characteristic for neural networks.

is a probability distribution over a vocabulary $V$. The probability distribution denotes how likely the words will occur as context words around the word $w_t$.

The model is represented by an embedding matrix $\mathbf{X} \in \mathbb{R}^{|V| \times d}$ (between the *input* and *projection* layer) and a context embedding matrix $\hat{\mathbf{X}} \in \mathbb{R}^{|V| \times d}$ (between the *projection* and *output* layer), where $d$ is specified dimension. These two matrices are the model's parameters $\Theta$. Given a sequence of training words $w_1, w_2, \ldots, w_T$, the model is optimized by minimizing the following objective function[2] $J_{SGNS}(\Theta)$:

$$J_{SGNS}(\Theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{-C \leq j \leq C, j \neq 0} \log p(w_{t+j} \mid w_t) \tag{4.3}$$

and $p(w_{t+j} \mid w_t)$ is computed using the softmax function:

$$p(w_{t+j} \mid w_t) = \frac{\exp(\hat{\mathbf{x}}_{w_{t+j}}{}^{\top} \mathbf{x}_{w_t})}{\sum_{i=1}^{|V|} \exp(\hat{\mathbf{x}}_{w_i}{}^{\top} \mathbf{x}_{w_t})} \tag{4.4}$$

where $\hat{\mathbf{x}}_i$ is the context word vector from matrix $\hat{\mathbf{X}}$ for word $w_i$ and $\mathbf{x}_i$ is the embedding word vector from matrix $\mathbf{X}$ for word $w_i$.

The original formulation of $\log p(w_{t+j} \mid w_t)$ is very expensive to compute (because of the denominator) and thus the *negative sampling* as an alternative solution for estimating the probability was proposed by Mikolov, Sutskever, et al. (2013). The idea of the negative sampling is to help the model distinguish the target word $w_t$ from words (called negative samples) taken from a noise distribution $P_n(w)$. Words from the noise distribution $P_n(w)$ are unlikely to occur as the context words of the target word $w_t$. The noise distribution $P_n(w)$ was empirically estimated and set to the unigram distribution raised to the $3/4^{th}$ power. The usual number of negative samples is $5 - 20$. The negative sampling is given by:

$$p(w_{t+j} \mid w_t) = \log \sigma(\hat{\mathbf{x}}_{w_{t+j}}{}^{\top} \mathbf{x}_{w_t}) + \sum_{i=1}^{N} \mathbb{E}_{w_i \sim P_n(w)} \left[ \log \sigma(-\hat{\mathbf{x}}_{w_i}{}^{\top} \mathbf{x}_{w_t}) \right] \tag{4.5}$$

where $N$ is a number of negative samples and $\sigma$ is the sigmoid function.

#### 4.1.3.2 Continuous Bag-of-Words

The *continuous bag-of-words* architecture is similar to the Skip-gram architecture. The goal of the CBOW model is to predict the target center word $w_i$ using all surrounding context words given by context size $C$. The CBOW objective function to be minimized is defined as:

$$J_{CBOW}(\Theta) = -\frac{1}{T} \sum_{t=1}^{T} \log p(w_t \mid w_{t-C}, \ldots, w_{t-1}, w_{t+1}, \ldots w_{t+C}) \tag{4.6}$$

and $p(w_t \mid w_{t-C}, \ldots, w_{t-1}, w_{t+1}, \ldots w_{t+C})$ is defined as:

[2] SGNS stands for *Skip-gram with negative sampling.*

$$p(w_t \mid w_{t-C}, \dots, w_{t-1}, w_{t+1}, \dots w_{t+C}) = \frac{\exp(\hat{\mathbf{x}}_{w_t}{}^{\top} \overline{\mathbf{x}}_{w_t})}{\sum_{i=1}^{|V|} \exp(\hat{\mathbf{x}}_{w_i}{}^{\top} \overline{\mathbf{x}}_{w_t})} \tag{4.7}$$

where the vector $\overline{\mathbf{x}}_{w_t}$ is the sum of vectors of context words $w_{t-C}, \dots, w_{t-1}, w_{t+1}, \dots w_{t+C}$ defined as follows:

$$\overline{\mathbf{x}}_{w_t} = \sum_{-C \leq i \leq C, i \neq 0} \mathbf{x}_{w_{t+j}} \tag{4.8}$$

The negative sampling technique can be used for the CBOW as well. According to experiments presented in Mikolov, Chen, et al. (2013), the CBOW model works slightly better than the Skip-gram model in capturing syntactic information, but the Skip-gram significantly outperforms the CBOW model in capturing semantic information.

### 4.1.4 FastText

FastText (Bojanowski et al., 2017) is based on the skip-gram model with negative sampling and employs sub-words (character n-grams). FastText uses character n-grams (hereinafter only n-grams) instead of entire words for training. Each n-gram has its own vector representation, the vector representation of words is computed as a sum of its character n-grams. Employing the sub-word information improves the vector representation for morphologically rich languages. The advantage of this approach is that the model can obtain vector representation even for words that did not appear in the training corpus.

### 4.1.5 Contextual Word Embeddings

Unlike static word embeddings, *contextual word embeddings* embed the word representation of word $w$ into a vector, based on its actual context (surrounding text). Thus, the vector representation will be different for different contexts. Contextual embeddings are usually generated by pre-trained language models like BERT or ELMo. However, the Transformer-based models such as BERT or GPT are, in principle, not intended to generate word embeddings, although they can do so. This is because the entire model is directly fine-tuned for a specific downstream task. ELMo, on the other hand, is supposed to be used to extract contextual word embeddings, which will then be employed in another model without modifying the weights of the ELMo model. Models that produce word embeddings differ in architecture and many other aspects, but usually, they have two main properties in common. (1) They are intended to learn and build a strong contextual representation of language. (2) They are trained on objectives similar or closely related to language modeling (Hewitt, 2019).

The dynamic embeddings can be formalized as a function of an entire sequence of text, unlike the static embeddings where the input is only a single word. These models take into account not only the word itself but also the context of the word and thus, they eliminate the polysemy problem from static word embeddings. We can define such a model as the function:

$$f_{context} =: (w_1, w_2, \ldots, w_N) \longrightarrow (\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_N) \tag{4.9}$$

where $(w_1, w_2, \ldots, w_N)$ is a text sequence of words, $(\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_N)$ are vectors for the corresponding words and each word $w_i \in V$. In this case, the word embeddings are not pre-trained, but they are produced by the pre-trained model on demand.

The disadvantage is that models for dynamic embeddings usually need much more computational resources to be trained. Thus, even though dynamic embeddings outperform static embeddings, static embeddings are still used because they can be easily trained for any language or domain using only a fraction of computation power compared to models for dynamic embeddings.

## 4.2 Language Modeling for Transformers

As we mentioned, the Transformer-based models are usually language models pre-trained on a huge amount of text. The idea behind using large amounts of text for pre-training is based on the *self-supervised* learning paradigm. Generally speaking, the data are utilized in the way in which the model is learned on a supervised task with unlabeled data. Usually, the model is trained to predict or reconstruct missing parts of the input data, such as artificially masked or removed words in the text or parts of images. Such an approach allows the model to extract useful representations from unlabeled data without explicit human annotations and learn meaningful representations that can be later transferred to downstream tasks. The obvious advantage is that it enables models to learn from vast amounts of unannotated data, reducing the reliance on expensive labeled datasets (Jurafsky & Martin, 2024).

In NLP, the *language modeling* (LM) tasks in different variants are used for self-supervised learning. The goal of a language model is to assign a probability to a sequence of words. Thanks to language modeling, the model gains general knowledge about the language (e.g., semantic and syntax information) that can be later exploited for fine-tuning. According to Jurafsky and Martin (2009), the language modeling tasks can be roughly categorized into two types: *causal language modeling* and *masked language modeling* also called *cloze task* (W. L. Taylor, 1953). Transformer-based models use both of these tasks for pre-training.

### 4.2.1 Causal Language Modeling

The goal of *causal language modeling* task is to predict the next word or token in text sequence based on the preceding context, as illustrated in Figure 4.2. Models built on causal language modeling are often used for text generation. Such a model works left-to-right, generating one output token at a time, conditioning its predictions solely on the history of previously generated tokens. This approach of using a language model to incrementally generate words by repeatedly sampling the next word conditioned on our previous choices is nowadays called *autoregressive generation* or *causal LM generation* (Jurafsky & Martin, 2024). The typical examples are the models from the Generative Pre-trained Transformer (GPT) family, see Section 4.3.1.

*The happiest fish live in* ________

Figure 4.2: An example of causal language modeling task.

The older n-gram language models (Jurafsky & Martin, 2009) used the chain rule of probability to estimate the probability of the next token $x_k$ in a sequence of $n$ tokens $(x_1, x_2, \ldots, x_n)$ based on the history $(x_1, \ldots, x_{k-1})$ as follows:

$$p(x_1, x_2, \ldots, x_n) = \prod_{k=1}^{n} p(x_k \mid x_1, x_2, \ldots, x_{k-1}) \tag{4.10}$$

## 4.2.2 Masked Language Modeling

*Masked language modeling* (MLM) models predict a masked token or word in textual sequence based on the entire sequence (context), as shown in Figure 4.3. It means that the model has access to all tokens on the left and right sides of the masked token, unlike in the case of causal language modeling, where only the previous tokens (history) were accessible. The task is also referred to as *cloze task* (W. L. Taylor, 1953).

*The happiest* ________ *live in sea.*

Figure 4.3: An example of masked language modeling task.

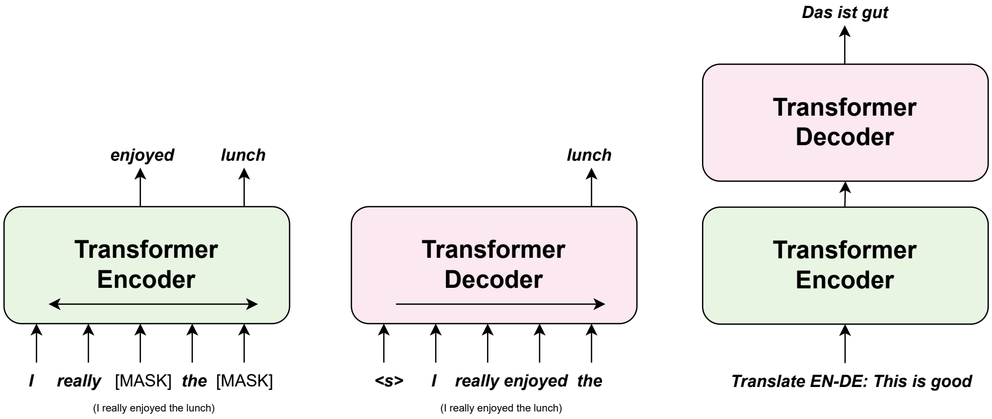
The example of a model that uses MLM is the BERT model (Devlin et al., 2019). The model is trained on a large unlabeled text corpus. From the sentences of the corpus, random tokens are sampled to be masked and predicted. The "masking" procedure includes three possibilities:

- The sampled token is replaced (masked) with the artificial `[MASK]` token.
- The sampled token is replaced with another randomly sampled token from the model's vocabulary.
- The sampled token is unchanged.

In BERT 15% input tokens are sampled for the masking procedure. Then 80% out of this 15% are masked as the `[MASK]` token, 10% are replaced with a random existing token and 10% remain unchanged. Then, the training objective is to predict the original tokens before the masking phase and only the masked tokens are used for learning, i.e., only these tokens are used in the cross-entropy loss function.

## 4.3 Transformer-based Models

Transformer-based models are built upon the Transformer architecture described in Section 3.6 and trained on a huge amount of unannotated text with the language modeling objective or its variant explained in the previous sections. Transformer-based models brought a significant improvement in performance in almost any NLP task (Raffel et al., 2020). Individual models use different parts of the Transformer architecture and different variants of the LM task. For example, as shown in Figure 4.4, the BERT model uses only the encoder part of the Transformer with MLM objective, GPT models use the causal language modeling objective with the decoder part of the Transformer, whereas the T5 model (Raffel et al., 2020) uses the original entire encoder-decoder architecture of the Transformer with a modified version of MLM.



(a) Encoder-only BERT.

(b) Decoder-only GPT.

(c) Encoder-decoder T5.

Figure 4.4: Language modeling objective and models comparison.

### 4.3.1 GPT Models Family

*Generative Pre-trained Transformer* (GPT) (Radford et al., 2018) follows the idea of training a language model on a huge amount of text data, which brings strong contextualized language representation that can be used in downstream tasks. GPT is a multi-layer Transformer decoder. The model is firstly pre-trained using the causal language modeling task. Then, for its usage in downstream tasks (e.g., sentiment analysis), the model is fine-tuned for the specific task. Fine-tuning means learning to modify the model's weights to produce desired outputs for the given downstream task.

GPT is the predecessor of the GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020), GPT-4 (OpenAI, 2023) and the famous ChatGPT (OpenAI, 2022) models. Unlike the first GPT model, the later models are not directly fine-tuned for the downstream tasks, but the *prompting* or *prompt-based learning* is used instead, see Section 4.3.7 for details. Loosely speaking,

[3] Image is based on https://lilianweng.github.io/lil-log.

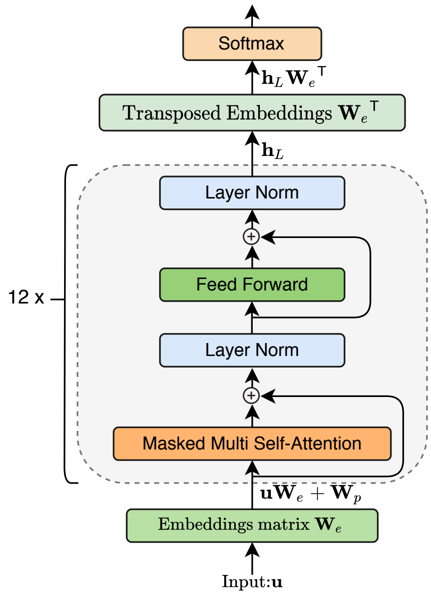

Softmax
$\mathbf{h}_L \mathbf{W}_e^\top$
Transposed Embeddings $\mathbf{W}_e^\top$
$\mathbf{h}_L$
Layer Norm
Feed Forward
Layer Norm
Masked Multi Self-Attention
12 x
$\mathbf{u}\mathbf{W}_e + \mathbf{W}_p$
Embeddings matrix $\mathbf{W}_e$
Input:$\mathbf{u}$

Figure 4.5: Architecture of the GPT model[3].

when prompting is applied, the description of the downstream task is part of the input and the model itself has to deduce the task and produce the desired output. Such an approach is used in *zero-shot learning*, where the model should be able to produce outputs for the task that has not been explicitly trained and for which it has not seen any examples. A similar concept is *few-shot learning*, where the model is fine-tuned only on a small number of examples, usually up to dozens of examples. The new GPT models (since GPT-2) demonstrated outstanding abilities in zero-shot learning on various NLP tasks. Next, we describe the first GPT model (Radford et al., 2018) since the newer models are built on its architecture.

The architecture of GPT is based on the original Transformer (Vaswani et al., 2017), but it is modified and it uses only the decoder part of the Transformer architecture, which is called *Transformer decoder* (P. J. Liu et al., 2018). The proposed model stacks 12 layers of Transformers followed by the final softmax layer that produces a distribution over the target tokens, see Figure 4.5 for the architecture overview.

During pre-training, the model for a given training sequence of tokens $\mathcal{X} = (x_1, x_2, \ldots, x_n)$ minimizes the following objective function $J_{GPT}(\mathcal{X}, \Theta)$:

$$J_{GPT}(\mathcal{X}, \Theta) = -\sum_{k=1}^{n} \log p(x_k \mid x_{k-j}, \ldots, x_{k-1}; \Theta) \tag{4.11}$$

where $\Theta$ are the optimized parameters of the model, $j$ is the size of the context window and

$k$ is the currently predicted token. The model can be expressed as follows:

$$\begin{aligned} \mathbf{h}_0 &= \mathbf{u}\mathbf{W}_e + \mathbf{W}_p \\ \mathbf{h}_\ell &= \text{transformer\_block}(\mathbf{h}_{\ell-1}), \forall \ell \in [1, L] \\ p(V) &= softmax(\mathbf{h}_L {\mathbf{W}_e}^\top) \end{aligned} \tag{4.12}$$

where $\mathbf{h}_0$ is the hidden state (output) of the input layer, $\mathbf{u} = (u_{-j}, \ldots u_{-1})$ is a context vector of tokens, $L$ is a number of Transformer layers[4], $\mathbf{h}_\ell$ are outputs of stacked Transformer layers, $\mathbf{W}_e$ is the token embeddings matrix and $\mathbf{W}_p$ is the position embeddings matrix and $p(V)$ is a probability distribution over the vocabulary $V$.

After the unsupervised training with the objective from Equation 4.11, the resulting pre-trained model can be fine-tuned for a certain task, for example, text classification. As a classification token, GPT uses the representation of the last token in the input sequence. The representation of the last token is obtained from the output $\mathbf{h}_L$ of the last layer of the model. Please see Section 4.3.6 for an explanation of the classification token and details about the classification with Transformers.

### 4.3.2 BERT

BERT stands for *Bidirectional Encoder Representations from Transformers* (Devlin et al., 2019). It is a model for language representation based on the Transformer architecture. BERT language modeling pre-training objective is to predict the masked tokens using the left context (previous tokens) and the right context (following tokens) jointly at once. This property is called *bidirectionality* (hence, the *bidirectional* word in BERT name).

The authors of BERT proposed two models. BERT$_{\text{BASE}}$ has the same size as GPT (i.e., 12 stacked layers of Transformer blocks, it contains 110 million parameters in total) in order to be comparable with GPT. BERT uses the WordPiece subword tokenizer with a vocabulary size set to 30$k$, the size of the hidden layers is set to 768 and it uses 12 attention heads. The second proposed model, BERT$_{\text{LARGE}}$ consists of 24 stacked layers of Transformer blocks with a total of 340 million parameters.

BERT pre-training is similar to GPT, but it differs mainly due to the mentioned ability to learn jointly from both (backward and forward) directions when iterating over text. BERT is trained on a large unlabeled text corpus with two auxiliary tasks instead of the basic language task. (1) *Masked Language Modeling*, for a given input word sequence, a certain portion of words are replaced by a special symbol `[MASK]` and the goal of the task is to recover the replaced original words without any information about them. (2) *Next Sentence Prediction* (NSP) is a task where for a given sentence pairs $A$ and $B$, the goal is to decide whether the $B$ sentence follows the $A$ sentence in a training corpus. The authors generated training corpus so that 50% of sentence pairs remained in the correct order and for 50% of sentence pairs, the $B$ sentence was replaced by other random sentences from the corpus. Then, the trained model is utilized for a specific downstream task in a similar way as GPT.

[4] The authors used 12 layers, but any number could be used.

An issue with the original Transformers, in general, is that the size of the input layer decides the complexity of the model. Both the time and memory requirements in the Transformer grow quadratically with the length of the input. Therefore, it is necessary to set a fixed input length long enough to provide sufficient context for the model to function and yet still be computationally tractable. For BERT, a fixed input size of 512 subword tokens was used which is also a common value for other Transformer-based models.

### 4.3.3 RoBERTa

RoBERTa model (Y. Liu et al., 2019) comes from *A Robustly Optimized BERT Pretraining Approach* and is a descendant of the BERT model. The authors stated that the original BERT model was significantly undertrained. They compared different key hyper-parameters and training data sizes and showed their important impact on the model's performance.

The architecture of RoBERTa is the same as the BERT model, but they modify the pretrain phase in multiple aspects. In the original BERT pre-training, the masking of tokens was done once during the pre-processing of training data and remained the same during the entire training. In RoBERTa, the masking is performed multiple times during the training, so the models do not always predict the same tokens. Secondly, they removed the next sentence prediction task. Next, they trained the model with a large batch size of $8k$ examples and they used a much larger text corpus for pre-training, 16GB of text was used for BERT and 160GB was used for RoBERTa. Finally, they increased the vocabulary size from $30k$ to $50k$.

### 4.3.4 T5

T5 model (Raffel et al., 2020) (from *Text-to-Text Transfer Transformer*) is a sequence-to-sequence model that processes the input text and as an output, it also produces text. Unlike traditional models trained for specific tasks, T5 follows a "text-to-text" framework where all tasks are reformulated as text generation problems. T5 model uses both parts of the Transformer architecture, i.e., encoder and decoder, with minor modifications.

The unsupervised pre-training language modeling objective differs from classical MLM in the way that instead of masking single tokens, T5 masks (corrupts) multiple tokens at once (spans) and replaces them with one sentinel token as shown in Figure 4.6.

The goal of the model is to generate and reconstruct the dropped-out spans (tokens sequences) delimited by the sentinel tokens used to replace them in the input plus a final sentinel token. Further, the model is trained (fine-tuned) with supervised tasks. Since each dataset is converted into a unified text-to-text format, as illustrated in Figure 4.7, they use a concatenation of all different datasets as one big supervised multitask dataset. The concatenated dataset consists of various NLP tasks such as machine translation, text summarization, text classification, question answering etc. Some are part of the GLUE (A. Wang et al., 2018) and SuperGLUE (A. Wang et al., 2019) benchmarks. In summary, while BERT focuses on bidirectional representations and specific pretraining objectives, T5 takes a text-to-text framework,

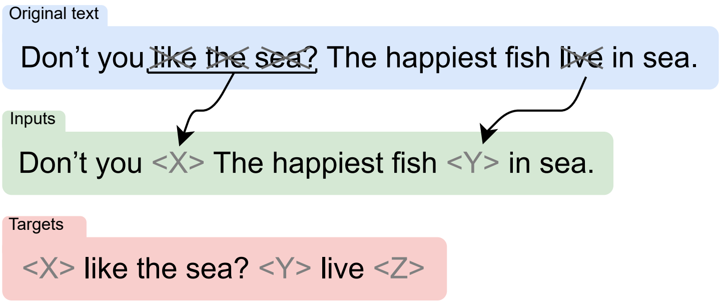

Original text
Don't you ~~like the sea?~~ The happiest fish ~~live~~ in sea.
Inputs
Don't you <X> The happiest fish <Y> in sea.
Targets
<X> like the sea? <Y> live <Z>

Figure 4.6: T5 model language modeling objective.

incorporating both encoder and decoder components, and handles a broader range of tasks through a unified text generation approach.

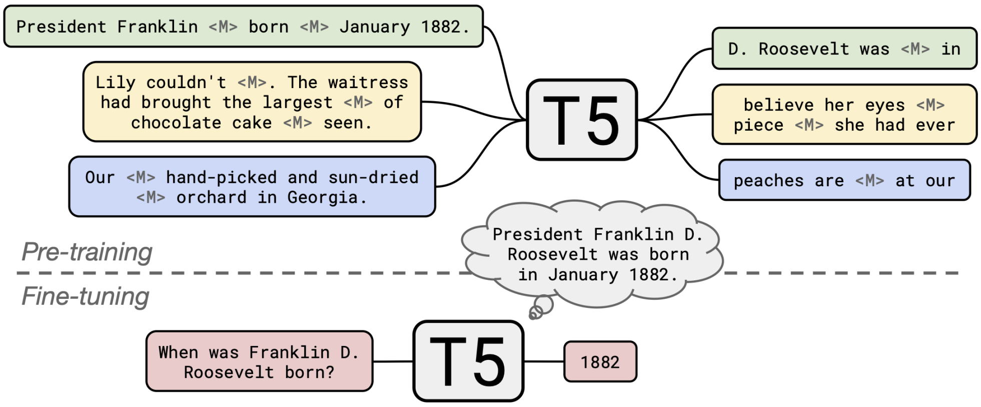

President Franklin <M> born <M> January 1882.
Lily couldn't <M>. The waitress had brought the largest <M> of chocolate cake <M> seen.
Our <M> hand-picked and sun-dried <M> orchard in Georgia.
T5
D. Roosevelt was <M> in
believe her eyes <M> piece <M> she had ever
peaches are <M> at our
President Franklin D. Roosevelt was born in January 1882.
Pre-training
Fine-tuning
When was Franklin D. Roosevelt born?
T5
1882

Figure 4.7: T5 pre-training and fine-tuning visualization. Image taken from Jurafsky and Martin (2009)

Later, a modified version of T5 were released, such as *byT5* model (Xue et al., 2022) or *T5v1.1* model which is an improved version of T5. T5v1.1 has some architectural updates and it is only pre-trained using the MLM without incorporating the supervised tasks. The BART model (Lewis et al., 2020) is a similar sequence-to-sequence model, although it is not based on the T5 model.

### 4.3.5 Multilingual Models

In the field of NLP, English is the prevailing and primarily used language. Thus, all the mentioned models were English monolingual models, i.e., they were trained and intended to be used only for English. However, there are many other languages and the need for models in these languages quickly emerged. To tackle the lack of models in other languages, there are two possible approaches: (1) train a monolingual model in the required language or (2) train a multilingual model that can handle multiple languages at once.

There are many monolingual models in other languages, for example, Czech (Kocián et al., 2022; Sido et al., 2021; Straka et al., 2021), French (Le et al., 2020; Martin et al., 2020), Arabic (Safaya et al., 2020), Romanian (Dumitrescu et al., 2020), Dutch (Delobelle et al., 2020; de Vries et al., 2019), Finnish (Virtanen et al., 2019), Polish (Kłeczek, 2020), Turkish (Schweter, 2020) or German (Chan et al., 2020).

Alternatively, a multilingual model can be trained. The multilingual model is pre-trained in multiple languages at once. The model can be used for multiple languages, i.e., it can be fine-tuned for different downstream tasks in different languages. In addition, thanks to the multilingual pre-training, the model usually gains cross-lingual capabilities to transfer knowledge from one language to another. Such cross-lingual property is very useful because, for example, the model can be trained for text classification in English and can be applied to evaluate the same task on data from another language. Examples of multilingual Transformer-based models are mBERT (Devlin et al., 2019), XLM (Conneau & Lample, 2019), XLM-R (Conneau et al., 2020), SlavicBERT (Arkhipov et al., 2019), mT5 (Xue et al., 2021) or mBART (Y. Liu et al., 2020). We discuss the cross-lingual capability in more detail in Section 6.3.

### 4.3.6 Classification with Transformers

The process of fine-tuning a pre-trained Transformer-based model for classification typically treats it as a sequence classification task. In this context, the input sequence (sequence of tokens) is passed through the entire model and the information important for classification is accumulated into one vector used for final predictions.

More formally, given a training corpus $\mathcal{T}$, each training sample $t_i \in \mathcal{T}$ contains a sequence of tokens $(x_1, \ldots, x_m)$ along with a label $y$. The training sample is then passed through the pre-trained model. The hidden vector $\mathbf{h} \in \mathbb{R}^H$ of the *classification token* `[CLS]` is often extracted from the pooling layer, which is usually a fully-connected layer of size $H$ and a hyperbolic tangent activation function. The dropout is applied to the vector, which is then passed through a task-specific linear layer represented by the matrix $\mathbf{W} \in \mathbb{R}^{|C| \times H}$, where C is a set of classes and $H$ is the hidden size of the model. The output class $c \in C$ is computed as $c = argmax(\mathbf{hW}^T)$. To obtain the probability distribution $\mathbf{y}$ of the possible output labels, the softmax activation is used as follows:

$$p(\mathbf{y} \mid x_1, \ldots, x_m) = softmax(\mathbf{hW}^T) \tag{4.13}$$

The model is then fine-tuned by minimizing the following objective $J(\mathcal{T}, \Theta)$:

$$J(\mathcal{T}, \Theta) = -\sum_{k=1}^{n} \log p(\mathbf{y} \mid x_1, \ldots, x_m; \Theta) \tag{4.14}$$

where $\Theta$ are the model's optimized parameters (weights). Usually, all parameters of the model are optimized. The cross-entropy loss function is usually used. The additional matrix $\mathbf{W}$ along with the softmax is often referred to as *classification head*, as illustrated in Figure

4.8. Generally, any introduced neural network or layer added at the top of the underlying Transformer can be called a classification head.

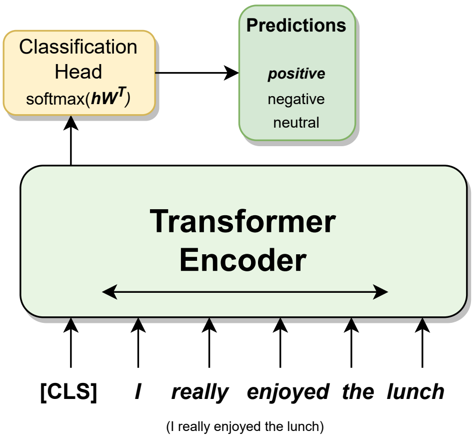


Figure 4.8: Visualization of classification principle with Transformer-based models.

The described approach is applicable for binary or single-label multi-class classification. For multi-label classification, the softmax activation is replaced with the sigmoid function. Then, each class $c$ will obtain its own independent probability of the classified example, denoting if it belongs to the given class. These probabilities are then used in the objective function. The classes whose probability overcomes a predefined threshold (usually 0.5) are assigned as the predictions.

The classification token is usually referred to as `[CLS]` token[5] or for example `<s>` in the case of RoBERTa model. This token is artificially added to the input sequence's beginning or end (usually in the case of causal language models such as GPT). The idea behind the classification token is that during the fine-tuning, the model learns to accumulate any information important for the classification from the entire sequence into the classification token. The classification token is directly used for the classification as described above.

The sequence-to-sequence models like T5 or BART can be fine-tuned in the same way, but more often, they are fine-tuned to generate the predictions or output as plain text.

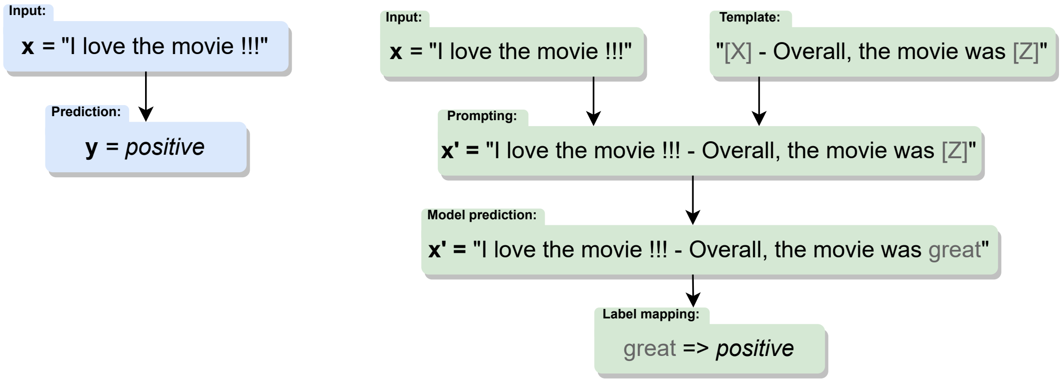
### 4.3.7 Prompt-based Learning

An alternative paradigm to the fine-tuning approach called *prompting* or *prompt-based learning* emerged recently. Prompting is a technique based on language models that modifies the original input (text) to contain a prompt that encourages the model to produce output for a given task. In the fine-tuning paradigm, the language model is firstly pre-trained and then utilized (fine-tuned) for a specific downstream task. However, in prompting, the model is usually[6] directly used for predictions because the downstream tasks are reformulated to be

[5] In the model's vocabulary, it is also literally represented as `[CLS]`.

[6] It can also be further trained (fine-tuned) for the specific task, but it follows the prompting paradigm.

similar to the original language modeling task with the help of the added prompt (P. Liu et al., 2023). For example, when performing polarity detection of the following sentence: *"I love the movie!!!"*, we can add a prompt like this: *"Overall, the movie was* _____*"* and let the model fill in the blank with a word that can be mapped to sentiment, for example, *"great"* would indicate the positive sentiment, as shown in Figure 4.9.


Input:
x = "I love the movie !!!"
Prediction:
y = positive
Input:
x = "I love the movie !!!"
Template:
"[X] - Overall, the movie was [Z]"
Prompting:
x' = "I love the movie !!! - Overall, the movie was [Z]"
Model prediction:
x' = "I love the movie !!! - Overall, the movie was great"
Label mapping:
great => positive

(a) Classical fine-tuning.

(b) Prompting.

Figure 4.9: Comparison of classical fine-tuning approach and prompt-based learning.

To define the prompting formally, let us first recall the goal of traditional supervised learning in NLP, particularly in text classification. For input text $\mathbf{x} = (x_1, \ldots, x_m)$ and assigned label $y$[7] being class $c$, the model produces the estimation of $\hat{y}$ of the true $y$. To predict the $\hat{y}$, the model computes the probability $\hat{p}(y = c \mid \mathbf{x})$ which is the estimation of the true probability $p(y = c \mid \mathbf{x})$. The model's estimation is obtained as follows:

$$\hat{y} = \hat{p}(y = c \mid \mathbf{x}; \Theta) \tag{4.15}$$

where $\Theta$ are the parameters of the model. The model learns the parameters from a dataset of annotated examples, where each input has assigned the expected output. Supervised data are required to train the model and utilize its parameters, but in some cases, getting a supervised dataset with sufficient examples can be problematic.

The prompt-based approaches tackle this problem by directly modeling the probability $p(\mathbf{x}; \Theta)$ of text $\mathbf{x}$ itself and using this probability to predict $y$, eliminating the need for the annotated dataset. Prompting modifies the input text $\mathbf{x}$ into a prompted input $\mathbf{x'}$. A *template* with two slots is used to modify the input. The *input slot* `[X]` for the input $\mathbf{x}$ and an *answer slot* `[Z]` for a generated answer $z$ by the model that will be mapped into $y$. Then, the template for polarity detection may look as follows: "`[X]` - *Overall, the movie was* `[Z]`". Then the model is asked to fill in the answer slot `[Z]` using its language modeling ability and based on the answer $z$, the output $y$ is derived with a mapping function $P_y(z)$ that maps the answer $z$ to the label $y$, the mapping function may look as follows:

[7] It can be a scalar in case of binary classification or a probability distribution, i.e., vector of probabilities for each class.

$$P_y(z) = \begin{cases} \mathit{positive} & \text{if } z \in \{\mathit{great, nice, amazing}\} \\ \mathit{neutral} & \text{if } z \in \{\mathit{ok, decent}\} \\ \mathit{negative} & \text{if } z \in \{\mathit{bad, terrible,boring}\} \end{cases} \tag{4.16}$$

The number of input and answer slots can be changed according to the solved task (P. Liu et al., 2023).

# 5 Datasets and Approaches for Sentiment Analysis

This chapter introduces datasets for SA and other related tasks. Further, we provide an overview of the tasks' approaches and related work that is important according to us.

In general, all approaches can be placed into three groups: *lexicon-based approaches, machine learning approaches* and *hybrid approaches*. The machine learning approach can be further divided into *supervised learning approach* and *unsupervised learning approach* (Giachanou & Crestani, 2016; B. Liu, 2012; Maynard & Funk, 2011; Medhat et al., 2014) as is shown in Figure 5.1 that illustrates the categorization of approaches that for SA.

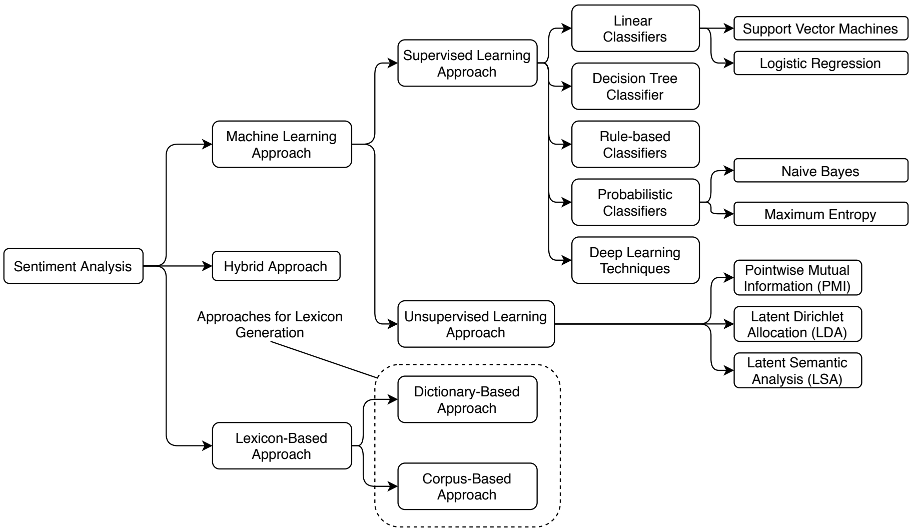

Sentiment Analysis
Machine Learning Approach
Hybrid Approach
Approaches for Lexicon Generation
Lexicon-Based Approach
Supervised Learning Approach
Unsupervised Learning Approach
Dictionary-Based Approach
Corpus-Based Approach
Linear Classifiers
Decision Tree Classifier
Rule-based Classifiers
Probabilistic Classifiers
Deep Learning Techniques
Support Vector Machines
Logistic Regression
Naive Bayes
Maximum Entropy
Pointwise Mutual Information (PMI)
Latent Dirichlet Allocation (LDA)
Latent Semantic Analysis (LSA)

Figure 5.1: Sentiment analysis techniques overview.

Since our work and contributions in this thesis are mainly focused on methods based on supervised machine learning, we do not describe other approaches such as *unsupervised learning*, *lexicon-based* or *hybrid* approaches. These techniques are usually obsolete and they perform much worse compared to the latest supervised methods, although in some situations (lack of data, unusual data domain), they may still be useful. For more detailed descriptions, please see Přibáň (2020). Section 5.2 focuses on the methods used before the Transformer era. In Section 5.3, we describe the latest techniques that utilize models based on the Transformer architecture that fall under the *deep learning* category in Figure 5.1.

## 5.1 Datasets

Datasets are required not only to train the supervised machine learning models but also to evaluate the performance of any system or approach dealing with any SA related task. This section summarises well-known and popular English datasets for the polarity detection task, emotion detection and subjectivity classification. We also include examples of datasets for other languages, even though most of the research was focused on English.

Process of manual annotation of datasets for NLP tasks is usually very expensive and time-consuming; therefore, researchers are trying to find approaches to get labeled data in another way. In SA, there are two main approaches to obtain annotated data: (1) *manual annotation* and (2) *distant supervision* (Giachanou & Crestani, 2016).

### 5.1.1 Manual Annotation

For tasks like polarity detection, the *manual annotation* can be done in any spreadsheet (MS Excel, Google Sheets etc.) or software specialized for data annotating, for example, *Amazon Mechanical Turk*[1]. The manual annotation is usually applied for more complex tasks like ABSA. Examples of manually annotated datasets can be found in L. Dong et al. (2014), Habernal et al. (2013), Saeidi et al. (2016), and Socher et al. (2013) or Rosenthal et al. (2017). The advantage of manual annotation is that the labels are usually more reliable and less erroneous.

### 5.1.2 Distant Supervision Annotation

The second option is the *distant supervision* approach. It automatically allows labeling data with minimal human interaction or completely without incorporating humans into the annotating process. The distant supervision techniques use metadata or other specific data properties to obtain the label. There are two main sources for distant supervision datasets.

The first one is a review of any type. The textual reviews usually contain additional explicit ratings (e.g., number of stars). This explicit rating is used as a label for the textual review. Such an approach was used, for example, in Maas et al. (2011), Pang and Lee (2004), and Pang et al. (2002) or Habernal et al. (2013).

The second source is social media websites (e.g., Twitter, Facebook). In this case, predefined emoticons, emojis or hashtags are used as a noisy label. The predefined emoticons or hashtags are tied up with a certain label (class) and based on their presence in the post, the corresponding class is used as the label. For example, in Tweet *"I'm so happy :) #cool #amazing"*

[1] https://www.mturk.com

emoticon *":)"* the hashtag *"#amazing"* is assigned to the positive sentiment. Thus, the Tweet is labeled as positive. A similar method was used in Go et al. (2009) and Speriosu et al. (2011).

With these approaches, a huge amount of annotated data can be obtained, but the reliability is lower than the manual approach.

We applied both the manual and the distant supervision approaches in our paper (Přibáň & Steinberger, 2022), where we present the dataset for subjectivity classification, see Section 8.3.

### 5.1.3 Corpora for Sentiment Analysis

The datasets for *sentence-level* and *document-level* tasks are listed in Table 5.1 and datasets for *aspect-based* task are shown in Table 5.2.

Here, we mention one well-known and commonly used[2] *Cornell Movie Review Dataset*[3] (Pang & Lee, 2004, 2005; Pang et al., 2002). The corpora consist of two datasets for SA and one for the classification of sentence subjectivity. The dataset for subjectivity classification consists of 5,000 subjective and 5,000 objective sentences obtained in a distant supervision manner. To gather subjective sentences, they downloaded movie review sentences or phrases. To obtain objective data, they took sentences from plot summaries of movies from the Internet Movie Database[4]. As mentioned above, we built the Czech dataset (Přibáň & Steinberger, 2022) for subjectivity classification. See the corresponding papers for further details of the datasets.

[2] According to authors, more than 100 papers used their dataset until 2012. The paper describing the dataset has more than 4, 600 citations according to https://scholar.google.com (July 2023).

[3] Available at http://www.cs.cornell.edu/people/pabo/movie-review-data/

[4] https://www.imdb.com

| Paper | Name | Size | Classes | Text | Domain | Source | Annotation | Language |
|---|---|---|---|---|---|---|---|---|
| (Pang et al., 2002) | Movie Reviews v1.0 | 1,400 | P, N | review | movie reviews | IMDb | ratings | English |
| (Pang & Lee, 2004) | Movie Reviews v2.0 | 2,000 | P, N | review | movie reviews | IMDb | ratings | English |
| (Pang & Lee, 2005) | Sentence Polarity | 10,662 | P, N | sentence | movie reviews | Rott. Tom. | ratings | English |
| (Go et al., 2009) | Sentiment140 | 1,600,000 | P, N | tweet | multiple | Twitter | emoticons | English |
| (Go et al., 2009) | Sentiment140 Test | 359 | P, N | tweet | multiple | Twitter | manual | English |
| (Shamma et al., 2009) | Obama-McCain Debate | 1,904 | P, N | tweet | Obama McCain | Twitter | manual | English |
| (Maas et al., 2011) | IMDb | 50,000 | P, N | review | movie reviews | IMDb | ratings | English |
| (Socher et al., 2013) | SST-5 | 11,855 | P+, N+, O | sentence | movie reviews | Rott. Tom. | manual | English |
| (Socher et al., 2013) | SST-2 | 9,613 | P, N | sentence | movie reviews | Rott. Tom. | manual | English |
| (X. Zhang et al., 2015) | Yelp-Fine | 140,000 | P+, N+, O | review | multiple | Yelp | ratings | English |
| (X. Zhang et al., 2015) | Yelp-Binary | 299,000 | P, N | review | multiple | Yelp | ratings | English |
| (Rosenthal et al., 2017) | SemEval-2017 | 62,617 | P, N, O | tweet | multiple | Twitter | manual | English |
| (Speriosu et al., 2011) | Health Care Reform | 2,394 | P, N, O | tweet | health care tweets | Twitter | emoticons | English |
| (Habernal et al., 2013) | Czech Social Media | 10,000 | P, N, O | FB post | multiple | Facebook | manual | Czech |
| (Habernal et al., 2013) | Czech Movie Reviews | 91,381 | P, N, O | review | movie reviews | CSFD | ratings | Czech |
| (Habernal et al., 2013) | Czech Product Reviews | 145,307 | P, N, O | review | product reviews | MALL.cz | ratings | Czech |
| (Villena-Román, 2013) | TASS-2013 | 68,017 | P+, N+, O, X | tweet | multiple | Twitter | mixed | Spanish |
| (Barbieri et al., 2016) | Evalita-2016 | 9,410 | P, N, O, M | tweet | multiple | Twitter | mixed | Italian |
| (Rosenthal et al., 2017) | SemEval-2017-AR | 9,455 | P, N, O | tweet | multiple | Twitter | manual | Arabic |
| (Théophile, 2020) | Allocine | 200,000 | P, N | review | movie reviews | Allociné | ratings | French |

Table 5.1: Overview of datasets for sentiment polarity classification. Values in column *Size* denote the number of examples in the dataset. Values in column *Classes* refer to the following classes: [P]: *positive*, [N]: *negative*, [P+]: *very positive* and *positive,* [N+]: *very negative* and *negative*, [O]: *neutral*, [M]: *mixed*, [X]: *none*. Values in column *Text* denote the type (granularity) of textual examples in the dataset (*FB post* stands for Facebook post). Values in column *Domain* represent a domain of the text. Values in column *Source* refer to the source web pages where the data comes from: (IMDb: www.imdb.com, Rott. Tom.: www.rottentomatoes.com, Twitter: www.twitter.com, Yelp: www.yelp.com, Facebook: www.facebook.com, CSFD: www.csfd.cz, MALL.cz: www.mall.cz), Allociné: https://www.allocine.fr. *Annotation* Column denotes the approach used for obtaining labels; *ratings* and *emoticons* values refer to the *distant supervision method*, *mixed* values mean that a combination of manual and distant supervision was used.

| Paper | Name | Size | Classes | Text | Domain | Source | Annotation | Language |
|---|---|---|---|---|---|---|---|---|
| (Saeidi et al., 2016) | SentiHood | 5,215 | P, N | answers | quiestion answering | Yahoo | manual | English |
| (Pontiki et al., 2014) | SemEval-2014 | 7,686 | P, N, O, C | sentence | laptops and restaurants reviews | - | manual | English |
| (Pontiki et al., 2015) | SemEval-2015 | 5,596 | P, N, O | sentence, review | multiple | - | manual | English |
| (Hercig, Brychcín, Svoboda, Konkol, & Steinberger, 2016) | Czech ABSA | 2,149 | P, N, O | sentence | restaurants | - | manual | Czech |
| (Pontiki et al., 2016) | SemEval-2016 | 70,790 | P, N, O | sentence | multiple | - | manual | Multilingual |
| (Barnes et al., 2022) | SemEval-2022 | 32,030 | P, N, O | sentence | multiple | - | manual | Multilingual |
| (L. Dong et al., 2014) | Target Dependent | 6,940 | P, N, O | tweet | multiple | Twitter | manual | English |

Table 5.2: Overview of datasets for aspect-based SA. Values in all columns have the same meaning as in Table 5.1. The letter *C* in column *Classes* refers to the *conflict* class and the *Yahoo* string refers to the www.yahoo.com website.

### 5.1.4 Datasets for Emotion Analysis

As we mentioned in Section 2.4.3, there are two main tasks: *emotion detection* and *emotion intensity detection*. Overviews of datasets for both tasks are organized in Tables 5.3 and 5.4. See the corresponding papers for further details of the datasets. Bostan and Klinger (2018) summarize and describe other existing and available datasets for emotion detection and map them to a common format in a way that can be used for future research.

| Paper | Dataset | Text | Size | Topic | Emotions | Multi |
|---|---|---|---|---|---|---|
| (Strapparava & Mihalcea, 2007) | SemEval 2007 | Headlines | 1,250 | News | E*, R | yes |
| (Scherer & Wallbott, 1994) | ISEAR | Descriptions | 7,667 | Events | E*, G, M | no |
| (Roberts et al., 2012) | EmpaTweet | Tweets | 7,000 | General | E*, R, L+N | no |
| (Mohammad et al., 2018) | SemEval 2018 | Tweets | 10,983 | General | E*, P*, R+O, N | yes |
| (Chatterjee et al., 2019) | SemEval 2019 | Dialogues | 38,424 | General | A, H, S+O | no |
| (Shrivastava et al., 2019) | TV-Charmed | Utterances | 13,354 | TV show | E*, R+O | no |
| (Schuff et al., 2017) | SSEC | Tweets | 4,868 | General | E*, R, T, U | yes |
| (Buechel & Hahn, 2017) | EmoBank | Sentences | 10,548 | General | - | - |

Table 5.3: Overview of datasets for the emotion intensity detection task. Values in column *Multi* denote whether the dataset contains multi-label annotations. Values in column *Emotions* refer to the following emotions and classes: [E*]: *anger, disgust, fear, joy, sadness*, [P*]: *trust, anticipation, love, optimism, pessimism*, [R]: *surprise*, [G]: *guilt*, [M]: *shame*, [L]: *love*, [H]: *happy*, [S]: *sadness*, [T]: *trust*, [U]: anticipation, [A]: *anger*, [O]: *other* or *neutral* class, [N]: *no emotion*.

| Paper | Dataset | Text | Size | Topic | Emotions |
|---|---|---|---|---|---|
| (Mohammad et al., 2018) | SemEval 2018 | Tweets | 12,634 | General | anger, fear, joy, sad. |
| (Mohammad & Bravo-Marquez, 2017) | WASSA 2017 | Tweets | 7,097 | General | anger, fear, joy, sad. |

Table 5.4: Overview of datasets for the emotion detection task.

## 5.2 Classical Supervised Approaches

Supervised learning is the most common way to solve the SA tasks (B. Liu, 2012; Medhat et al., 2014). There are numerous papers using supervised machine learning. For example, (Balahur & Turchi, 2012; Baziotis et al., 2017; Go et al., 2009; Y. Kim, 2014; Kiritchenko, Zhu, & Mohammad, 2014; Martineau & Finin, 2009; Pak & Paroubek, 2010; Pang et al., 2002; Socher et al., 2013; Sun, Qiu, et al., 2019) and many more.

### 5.2.1 Feature-based Approaches

The initial supervised approaches (Balahur & Turchi, 2012; Go et al., 2009; Habernal et al., 2013; Kiritchenko, Zhu, & Mohammad, 2014; Martineau & Finin, 2009; Pak & Paroubek, 2010; Pang et al., 2002) heavily relied on *feature engineering*. The input of supervised machine learning algorithms is a vector of numbers, but in NLP, there is usually only unstructured text as the input for any task. To use text as the input for traditional machine learning methods, features from the text must be extracted and converted to vectors. In feature engineering, the domain knowledge of an engineer or researcher is used to define and extract characteristic and important features from raw data (i.e., text) for the given task. We list the commonly used features in SA:

- **Terms Presence and Frequency**. One of the most common and basic features is the presence of individual words (*unigrams*) or other *n-grams* with their frequency. Another option is to use a weighting method for the number of occurrences, for example, *tf-idf* weighting scheme (Manning et al., 2010).

- **Part of Speech**. The part-of-speech (POS) tags of words can hold valuable information related to SA, such as adjectives. So, the presence of certain POS tags, their combination or counts can be used as separate features.

- **Sentiment Words and Phrases**. Words, phrases and idioms from sentiment lexicons can also be used as features. The presence of positive (*good, nice*) or negative (*bad, poor*) words is a beneficial indicator of sentiment. Most sentiment words are adjectives or adverbs, but verbs (e.g., *hate, like* or *love*) and nouns (e.g., *junk, rubbish*) hold some sentiment as well (B. Liu, 2012).

- **Negations**. Negations in text change the sentiment orientation. Thus, their presence can be used as a feature.

- **Syntactic Features**. Other syntactic features besides the POS tags can be generated from parsing or dependency trees. These features capture word dependencies and the structure of sentences.

- **Stylistic Features**. Stylistic features (Giachanou & Crestani, 2016) are typically used for a text from social media websites (Facebook, Twitter). They capture non-standard writing styles like emoticons, emojis, abbreviations, slang expressions or specific punctuation.

Work presented in Pang et al. (2002) is one of the earliest that uses supervised machine learning. They classified movie reviews as positive or negative with the Naive Bayes classifier, SVM and Maximum Entropy classifier. They experimented with the following features: unigrams, bigrams, adjectives and POS tags.

Go et al. (2009) focused on classifying Tweets with distant supervision using the Naive Bayes classifier, Maximum Entropy classifier and SVM. The work was based on Pang et al. (2002) and used similar features and machine learning algorithms. They automatically created a training dataset of 1.6M tweets (50% negative and 50% positive). Pak and Paroubek

(2010) built a Naive Bayes classifier based on traditional n-gram and POS features that can classify Tweets as positive, negative or neutral. Habernal et al. (2013) created three Czech datasets for polarity detection and performed initial experiments with SVM and Maximum Entropy classifier using n-gram and POS features.

So far, we mentioned works focused solely on the *document-level* or *sentence-level* polarity detection between which we do not distinguish. Similar feature-based approaches can also be utilized for other tasks, including emotion analysis or subjectivity classification as shown in Balahur et al. (2012), Barbosa and Feng (2010), and Roberts et al. (2012).

The *aspect-based* SA can usually be divided into several sub-tasks, such as *aspect extraction* or *aspect sentiment classification*, as discussed in Section 2.3.3. The aspect sentiment classification is very similar to the sentence and document-level. Often, it can be transformed into classification tasks, and approaches similar to the ones used for sentence- and document-level SA can be applied. The common methods for aspect extraction are based on *sequential learning*, for example, *Hidden Markov Models* (HMM) (Rabiner, 1990) or *Conditional Random Fields* (CRF) (Lafferty et al., 2001). The HMM approach was used in Jin and Ho (2009) and usage of CRF can be found, for example, in Choi and Cardie (2010), Hercig, Brychcín, Svoboda, and Konkol (2016), and Jakob and Gurevych (2010).

We used the classical feature-based approach supervised learning for polarity detection in Přibáň and Balahur (2023) and for emotion intensity detection task in Přibáň et al. (2018).

### 5.2.2 Deep Learning and Neural Networks

With the progress and more frequent usage of neural network models, the focus has shifted to *architecture engineering*. In these approaches (Baziotis et al., 2017; Y. Kim, 2014; Socher et al., 2013), the model is designed for one specific task in such a way that features are learned internally by the model itself instead of manually defining and extracting features from the text (P. Liu et al., 2023). Such approaches are also referred to as *deep learning*. These approaches are usually used with the pre-trained word embeddings discussed in Section 4.1 that represent the input text.

Socher et al. (2013) introduced neural network architecture called *Recursive Neural Tensor Network* and nowadays very known *Stanford Sentiment Treebank* dataset. The model was able to capture accurately the effect of negation and its scope at various tree levels for both positive and negative phrases. They used a corpus of movie reviews from Pang and Lee (2005) and parsed the dataset's sentences into o parse trees, which were subsequently annotated by human judges. Using the novel Recursive Neural Tensor Network and the created dataset, they were able to push state-of-the-art results in sentence positive/negative classification from 80% of accuracy up to 85.4%.

Another breakthrough work is presented in Y. Kim (2014). He was the first, who effectively used a *convolutional neural network* (CNN) and pre-trained word embeddings for polarity detection and other sentence-level classification tasks. The proposed model improved the state-of-the-art results on 4 out of 7 tasks.

Baziotis et al. (2017) won with their deep learning system the SemEval-2017 Task 4 competition called *Sentiment Analysis in Twitter* (Rosenthal et al., 2017). They employed LSTM network with an attention mechanism on top of pre-trained word embeddings (they classified Tweets as positive/negative/neutral).

Huang et al. (2019) used a combination of CNN and BiLSTM (Graves & Schmidhuber, 2005) for emotion detection and predicting emotion intensity. Abdul-Mageed and Ungar (2017) applied Gated Recurrent Neural Network (Cho, van Merriënboer, Gulcehre, et al., 2014; Chung et al., 2015). The authors of Abdul-Mageed and Ungar (2017), Agrawal and Suri (2019), Baziotis et al. (2018), Huang et al. (2019), Polignano et al. (2019), and Shrivastava et al. (2019) also used deep learning models based on CNN or recurrent neural networks. We compare some works focused on emotion analysis in Tables 5.5 and 5.6.

| **Paper** | **Emotion Model** | **Approach** | **Dataset** | **Result** |
|---|---|---|---|---|
| (Strapparava & Mihalcea, 2008) | Categorical | LSA | Semeval 2007 | 18% $F_1$ Score |
| (Balabantaray et al., 2012) | Categorical | SVM | Their own | 73% Accuracy |
| (Balahur et al., 2012) | Categorical | SVM | ISEAR | 45% $F_1$ Score |
| (Roberts et al., 2012) | Categorical | SVM | EmpaTweet | 67% $F_1$ Score |
| (Buechel & Hahn, 2016) | VAD/Categorical | SVM | Semeval 2007 | 0.42 Pearson Correl. |
| (Abdul-Mageed & Ungar, 2017) | Categorical | GRU | Their own | 96% $F_1$ Score |
| (Baziotis et al., 2018) | Categorical | LSTM | SemEval 2018 | 53% $F_1$ Score |
| (Huang et al., 2019) | Categorical | CNN, LSTM | SemEval 2018 | 65% $F_1$ Score |
| (Alhuzali & Ananiadou, 2021) | Categorical | BERT | SemEval 2018 | 58% $F_1$ |
| (Polignano et al., 2019) | Categorical | CNN, LSTM | SemEval 2018 | 84% $F_1$ Micro Score |
| (Alhuzali & Ananiadou, 2021) | Categorical | BERT | SemEval 2018 | 71% $F_1$ Micro Score |
| (Polignano et al., 2019) | Categorical | CNN, LSTM | SemEval 2019 | 70% $F_1$ Micro Score |
| (Agrawal & Suri, 2019) | Categorical | LSTM | SemEval 2019 | 78% $F_1$ Micro Score |
| (Polignano et al., 2019) | Categorical | CNN, LSTM | ISEAR | 52-78% $F_1$ Score |
| (Shrivastava et al., 2019) | Categorical | CNN | TV-Charmed | 72% $F_1$ Score |

Table 5.5: Overview of papers related to the emotion detection task.

| **Paper** | **Approach** | **Dataset** | **Result** |
|---|---|---|---|
| (Köper et al., 2017) | Random Forrest, CNN, LSTM | WASSA 2017 | 0.72 Pearson Correlation |
| (Goel et al., 2017) | CNN, LSTM | WASSA 2017 | 0.74 Pearson Correlation |
| (Duppada et al., 2018) | XG Boost, Random Forrest | SemEval 2018 | 0.80 Pearson Correlation |
| (Huang et al., 2019) | CNN, LSTM | WASSA 2017 | 0.77 Pearson Correlation |

Table 5.6: Overview of papers related to the emotion intensity detection task.

To the best of our knowledge, the best results on subjectivity classification were achieved by *AdaSent* model (Zhao et al., 2015) with 95.5% of accuracy. AdaSent stands for a *self-adaptive hierarchical sentence model*, a deep learning model for representing sentence meaning. AdaSent is inspired by the gated recursive convolutional neural network (Cho, van

Merriënboer, Bahdanau, & Bengio, 2014) and forms the representation of the sentence from traditional word embeddings (e.g., word2vec, GloVe or fastText) but can incorporate the order of words in the sentence and thus improve results on downstream tasks like subjectivity classification. Cer et al. (2018) achieved 93.9% of accuracy with another model for sentence semantic representation, called *Universal Sentence Encoder*. The model produces an input sentence's fixed-length vector (sentence embeddings).

Poria et al. (2016) first used the deep learning approach to aspect extraction. They employed CNN in combination with linguistic patterns and word embeddings to tag each word in a sentence as either an aspect or non-aspect word. Shu et al. (2017) tried to tackle the problem of domain dependency. They proposed using a pre-trained CRF model for aspect extraction on different domains to improve results on a new domain. In H. Xu et al. (2018), a novel convolutional neural network model is proposed with two types of pre-trained word embeddings, i.e., general-purpose and domain-specific. Using two types of embeddings brought performance improvement and the model outperformed the other state-of-the-art methods at that time.

In the case of the aspect sentiment classification task, supervised methods for sentence-level polarity detection can also be applied. Khalil and El-Beltagy (2016) used a CNN classifier with fine-tuned word embeddings for a specific domain to detect aspect sentiment polarity of laptops and restaurant reviews. P. Chen et al. (2017) proposed a novel model that adopts a multiple-attention mechanism to capture sentiment features separated by a long distance. They combined multiple attentions with a recurrent neural network, concretely Long Short-Term Memory and Gated Recurrent Unit. Liu et al. (2018) proposed a novel recurrent neural network architecture with external memory and a delayed memory update mechanism to track entities specifically for the ABSA task. Liang et al. (2022) and C. Zhang et al. (2019) employed graph convolutional networks. Another related work can be found in Li et al. (2020).

The mentioned works showed that incorporating deep learning techniques and neural networks is beneficial, significantly outperforming the traditional supervised machine learning algorithms and pushing forward the state-of-the-art results. For other deep learning approaches and a more comprehensive survey, see L. Zhang et al. (2018).

We applied similar deep learning techniques for polarity detection, aspect-based sentiment analysis and emotion analysis tasks in Přibáň and Martínek (2018), Přibáň and Pražák (2023), Přibáň and Steinberger (2021), and Přibáň et al. (2022, 2024).

## 5.3 Transformer-based Approaches

Finally, with the advent of Transformer-based models, a sea change arrived and the *pre-train* and *fine-tune* paradigm emerged. Usually, the Transformer-based model is pre-trained as a language model to obtain general knowledge about the language. Then, this pre-trained model is utilized for a given task by introducing additional task-specific weights into the model and being fine-tuned on the annotated task data. Within this paradigm, the focus turned mainly to *objective engineering*, designing the training objectives used at both the pre-training and fine-tuning stages (P. Liu et al., 2023).

In the original paper (Devlin et al., 2019), BERT achieved 94.9% of accuracy on *SST-2* dataset. Sun, Qiu, et al. (2019) utilize BERT fine-tuning methods for text classification tasks. They were able to achieve 95.79% of accuracy on *IMDb* dataset, 28.62 of error rate[5] on *Yelp-Fine* dataset and 1.81 of error rate on *Yelp-Binary* dataset.

*XLNet* Yang et al. (2019), an improved variant of BERT, outperformed BERT on 20 NLP tasks, including four well-known datasets for sentiment polarity classification. More concretely, for polarity detection, XLNet achieved 96.2% of accuracy on *IMDb* dataset, 96.8% of accuracy on *SST-2* dataset, 27.8% of error rate on *Yelp-Fine* dataset and 1.55 of error rate on *Yelp-Binary* dataset. Jiang et al. (2020) proposed an enhanced framework for the fine-tuning phase of Transformer models and achieved an accuracy of 97.5% on the SST dataset.

In Nandi et al. (2021b), the authors compared multiple approaches for subjectivity classification. They also fine-tuned the BERT model, obtaining 96.6% accuracy on the Cornell Movie Review dataset. Q. Chen et al. (2022) used BERT and RoBERTa models and contrastive learning techniques for text classification, including subjectivity classification, achieving an accuracy of 97.3% on the same Cornell Movie Review dataset. Alhuzali and Ananiadou (2021) utilized the BERT model and introduced a new span prediction technique for emotion classification.

Sun, Huang, and Qiu (2019) simultaneously solve aspect extraction and aspect sentiment classification tasks by introducing auxiliary sentences and transforming the problem into a sentence-pair classification task. H. Xu et al. (2019) and Rietzler et al. (2020) improved results by pre-training the model on the task domain data of the ABSA task. J. Liu et al. (2021) treated the ABSA task as a text generation task outperforming the previous SotA results.

We employed the Transformer-based models for subjectivity classification and polarity detection tasks in Přibáň and Pražák (2023), Přibáň and Steinberger (2021, 2022), Přibáň et al. (2024), Sido et al. (2021), and Šmíd and Přibáň (2023)

### 5.3.1 Prompt-based Learning

The second sea change is currently in progress, in which the *pre-train* and *fine-tune* paradigm is fading from the main interest of researchers and is being replaced by the *prompting* or *prompt-based learning*. This paradigm usually utilizes the pre-trained (language) model to reformulate the solved problem into a form similar to the pre-training language modeling task by using a textual prompt. The model can then predict the output directly by using the language modeling ability that was pre-trained. See Section 4.3.7 for a more detailed description.

Gao et al. (2021) experimented with prompt-based fine-tuning for SC. With the English T5 model, they automatically generated prompts for BERT and RoBERTa models, which they

[5] The error rate is computed as $1 - accuracy$

consequently fine-tuned for the SC task. They demonstrated that their few-shot approach leads to better results than traditional fine-tuning. Hosseini-Asl et al. (2022) leverage the text generation ability of the GPT-2 model and apply it to the ABSA and polarity detection tasks. Mao et al. (2022) conducted an empirical study of prompt-based polarity and emotion detection tasks. They utilized RoBERTa and BERT models and analyzed the biases of these models for the evaluated tasks.

W. Zhang, Li, et al. (2021) formulate the ABSA tasks as a text generation problem. They propose two paradigms to deal with the ABSA tasks, namely annotation-style and extraction-style modeling, both generating textual output in a desired format. They utilize the English T5 text-to-text Transformer-based model and evaluate their approach on various ABSA tasks on datasets from the SemEval competitions (Pontiki et al., 2014, 2015, 2016). They showed the effectiveness of their approach by establishing new SotA results. Similarly, the work of W. Zhang, Deng, et al. (2021) used the same English T5 model to solve a newly introduced ABSA task called *aspect sentiment quad prediction* by generating textual output. Another approach proposed by Gao et al. (2022) aims to solve multiple ABSA tasks simultaneously. The authors applied the English T5 model to a prompt created from the individual ABSA tasks. They evaluated their model on the same datasets as W. Zhang, Li, et al. (2021), outperforming the previously mentioned approach and achieving new SotA results.

#### 5.3.1.1 Sentiment Analysis with LLMs

Currently, for prompting the large language models (LLMs) such as GPT-3 (Brown et al., 2020), GPT-4 (OpenAI, 2023), Llama (Touvron, Lavril, et al., 2023), Llama 2 (Touvron, Martin, et al., 2023), PaLM 2 (Anil et al., 2023) or Orca (Mukherjee et al., 2023) are used.

Qin et al. (2023) evaluated the ChatGPT model on seven downstream NLP tasks, including sentiment classification of the SST-2 dataset. They use a simple prompt to predict the sentiment of movie reviews, achieving 87.6% accuracy on 876 test examples. In Zhong et al. (2023), the authors evaluated ChatGPT on the popular GLUE (A. Wang et al., 2018) benchmark that contains the SST-2 dataset. They achieved 92.0% of accuracy on 50 examples and showed that ChatGPT achieves lower performance compared to fine-tuned BERT-like models. Further, they improved the results with a technique called *chain-of-thought* (CoT) proposed by Wei et al. (2022) and improved the accuracy to 96%.

Fei et al. (2023) utilized ChatGPT and the CoT prompt technique to detect implicit sentiment. Han et al. (2023) evaluated ChatGPT on a wide range set of aspect-based sentiment tasks, showing a significant performance gap between ChatGPT and other SotA methods. The authors claim that as the complexity of the task increases, the decline in ChatGPT's performance becomes more pronounced when compared to leading SotA techniques. W. Zhang et al. (2023) comprehensively explored the performance of ChatGPT on various sentiment classification and aspect-based sentiment analysis tasks. They also state that the performance of LLM models decreases with more complex tasks requiring deeper understanding or structured sentiment information compared to supervised fine-tuning.

We applied the prompt-based learning for the sentiment classification and ABSA tasks in Přibáň et al. (2024) and Šmíd and Přibáň (2023).

# Multilingual Sentiment Analysis

# 6

This chapter focuses on the approaches for multilingual SA, the underlying techniques and their details. Additionally, we present related work for cross-lingual sentiment analysis (CLSA).

The CLSA is a challenging task, aiming to enable SA in other languages[1] with limited or no labeled data (in these, i.e., *target languages*) by transferring knowledge from a language (called *source language*), typically English, where the annotated data are available, see Figure 6.1 for visualization. A more advanced and difficult version of CLSA is the so-called zero-shot CLSA, which relies exclusively on data from the source language (e.g., English) to build the system.

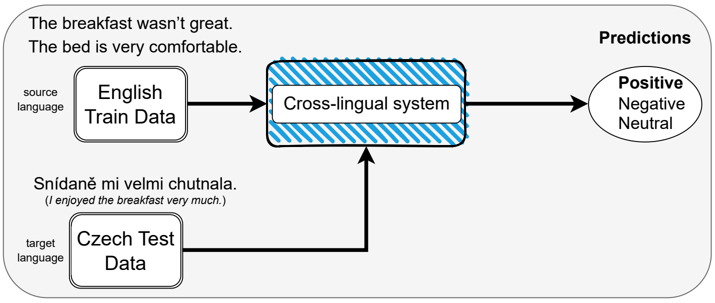

The breakfast wasn't great.
The bed is very comfortable.
source language
English Train Data
Cross-lingual system
Predictions
Positive
Negative
Neutral
Snídaně mi velmi chutnala.
(I enjoyed the breakfast very much.)
target language
Czech Test Data

Figure 6.1: Visualization of a model for cross-lingual sentiment analysis.

## 6.1 Cross-lingual Sentiment Analysis

At the beginning of the SA research, the predominant focus was almost exclusively on English. However, in the following years, the attention has moved and research has been done even for other languages. Consequently, multilingual and cross-lingual methods have been developed in recent years. Nonetheless, the development of effective multilingual techniques for a majority of NLP tasks remains an ongoing and intricate challenge. Additionally, a persistent issue concerns the prevalence of English-centric datasets. In other words, there is

[1] These languages are often called the low-resource languages.

very little (or any) of SA datasets for other languages, so-called *low-resource languages* (Balabantaray et al., 2012; X. Chen et al., 2018; Dashtipour et al., 2016; B. Liu, 2012; Ruder et al., 2019) compared to the English language.

### 6.1.1 The Multilingual and Cross-lingual Concepts

The *multilingual* and *cross-lingual* concepts are very closely related and there is a significant overlap between them, yet they are not equal. Generally, in NLP, a *multilingual* system or approach is designed to process text (perform some NLP task) in more than one language. Such systems often encompass components or parts shared across all languages (e.g., some common preprocessing steps) and language-specific parts, for example, training sentiment classifiers for each language separately. On the other hand, the *cross-lingual* system transfers or adapts knowledge of other languages to perform the task. The approach (or its parts) for a particular language depends on other languages' approaches, data, or tools. The *multilingual* and *cross-lingual* concepts are often used interchangeably, even though they are not equal.

### 6.1.2 Cross-lingual Approaches

The primary motivation for developing cross-lingual methods is to enable *transfer learning* across languages, in most cases between resource-rich language (e.g., English) and low-resource language. The goal is to develop methods that will allow us to use resources (data, methods etc.) of resource-rich languages in a certain NLP task for low-resource languages (Ruder et al., 2019). A resource-rich language is a language that has enough available resources for a specific NLP task. Let us explain the concept of *target* and *source* language. The *source* language denotes the language used for obtaining some knowledge or training data, usually, it is the resource-rich language. The *target* language is usually the low-resource language and aims to solve the task in the target language.

We divide the existing CLSA approaches into three main categories – *machine translation* approaches, *methods for explicit transfer of knowledge between languages* and *multilingual Transformers-based models*.

Machine translation can be utilized to translate annotated data from the source language to the target language and employ these translations to train a model as shown in Balahur and Turchi (2012).

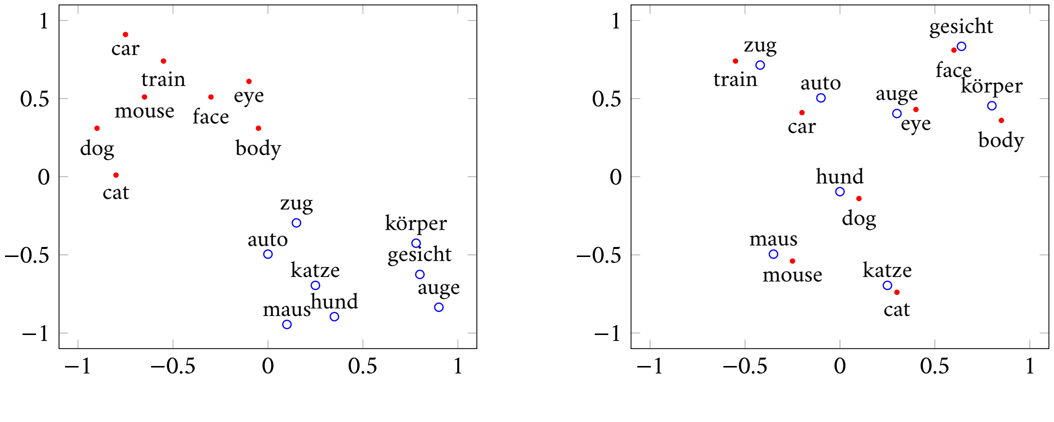
Second, methods that transfer knowledge between languages explicitly. For example, use linear transformation to align semantic spaces in different languages into one common space. The shared cross-lingual space (word embeddings) represents the input used to train neural networks, such as CNN or LSTM, as used in Abdalla and Hirst (2017) and Přibáň et al. (2022).

Lastly, multilingual Transformer-based models like mBERT or XLM-R can be fine-tuned with annotated source language data. The fine-tuned model can classify text in the target languages due to the model's cross-lingual properties. The cross-lingual ability of these models to transfer knowledge between languages is obtained during their pre-training phase.

The cross-lingual approaches are based on the monolingual techniques and principles described previously. The main idea of the cross-lingual approach is to add a method that transfers the knowledge necessary for a given task between languages. As mentioned, we divide the approaches according to the type of method that does the transformation. Based on the thesis' goals, this thesis focuses solely on the multilingual Transformer-based models and methods that explicitly transfer knowledge between languages (i.e., linear transformations in our case).

## 6.2 Cross-lingual Word Embeddings

As described in Section 4.1, word embeddings, allow us to capture the meaning of words in a vector representation. Recently, they have become extremely useful and important in building NLP systems. Cross-lingual word embeddings (CLWE) project multiple monolingual word embeddings spaces into one shared space where vectors are in different languages. CLWE allows representing words multilingually. It means that vectors representing semantically close words in different languages are similar, see Figure 6.2.



(a) Embeddings before the projection.

(b) Embeddings after the projection.

Figure 6.2: Sample visualisation of monolingual embeddings for English and German before and after their projections into a shared cross-lingual space.

In SA (and in NLP generally), CLWE allows transferring knowledge between languages, which is very useful, especially for low-resource languages. For example, with CLWE, approaches using monolingual embeddings that were already developed can be trained with the training data from resource-rich language and CLWE. Thanks to the cross-lingual embeddings and their properties, the trained model can now predict samples from the low-resource language.

Next, we focus on mapping only two monolingual embeddings. These CLWE are also called bilingual word embeddings (BWE). The ultimate, more complex goal of CLWE is to learn a shared embedding space between words in all languages (Ruder et al., 2019).

Ruder et al. (2019) categorize cross-lingual embedding methods mainly by the parallel data required by the methods. The parallel data represent the bilingual supervision signal that allows us to learn to align two monolingual spaces into one cross-lingual space. The main differences between the models usually come from the required data. The other differences are not so important since they are just implementation details for the specific architecture. To support this claim, they showed that the methods usually optimize the same or similar learning objectives (it is just written in different forms).

The parallel data utilized in these methods have two key properties that distinguish them: (1) *type of alignment* and (2) *comparability* of the parallel data. The parallel data alignment defines whether the data are aligned at the level of words, sentences or documents. The comparability means how much are the parallel data similar, i.e., whether it is a literal translation (data are *parallel*) or whether the parallel data are just similar (*comparable*). Here, we do not distinguish between them. Finally, we can define the fundamental categorization according to the type of data alignment:

1. **Word-level alignment**: Most methods use data aligned at the word level, typically in bilingual or multilingual dictionaries containing pairs of translated words. Such dictionaries are easy to obtain for most languages. Most of these approaches utilize pre-trained monolingual word embeddings, a bilingual dictionary and linear transformation.

2. **Sentence-level alignment**: A parallel corpus aligned at a sentence level is another type of data used by cross-lingual methods. An example of a commonly used sentence-level aligned dataset is Europarl corpus (Koehn, 2005).

3. **Document-level alignment**: A parallel corpus that contains translated documents in different languages. An example of such a corpus is Wikipedia, where many pages (about the same topic) are in multiple languages.

Further, we aim for word-level alignment methods since we use them in our research contribution of this thesis. We summarize the multi-level categorization of methods for CLWE in Figure 6.3. We do not describe all the mentioned types of methods. A detailed description of the other methods not mentioned, including the original papers, can be found in Ruder et al. (2019).

### 6.2.1 Methods for Word-level Alignment Data

Mapping-based approaches transform pre-trained monolingual word embeddings using linear transformation and bilingual dictionaries into one joint space (Brychcín, 2020). Linear transformation allows transformation between two vector spaces[2] using affine transformations, e.g., scaling, rotation, translation or reflection.

[2] By space, we mean word embeddings, also called semantic space, expressed by matrix $\mathbf{X}$.

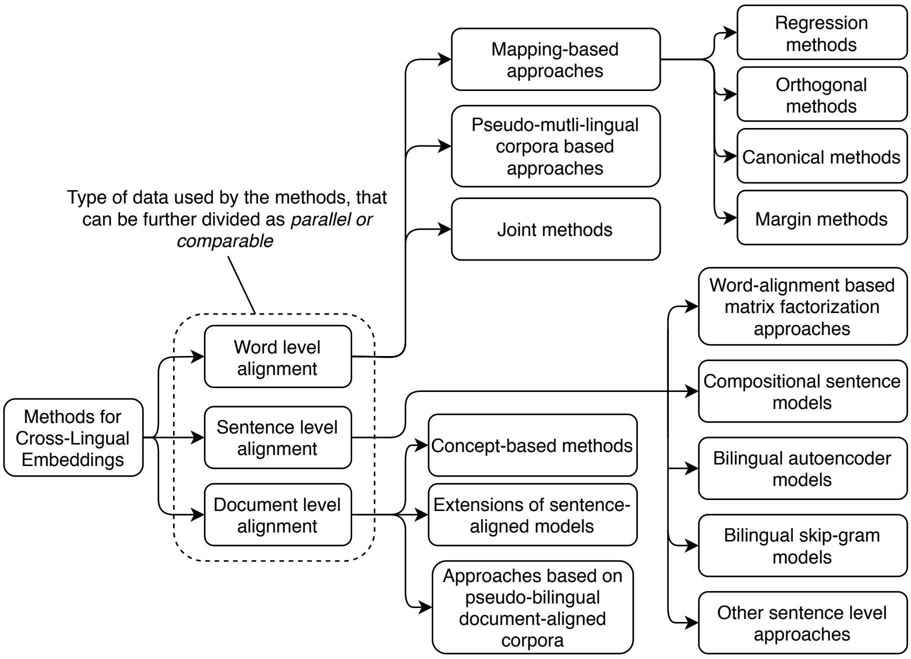

Type of data used by the methods, that can be further divided as *parallel or comparable*
Methods for Cross-Lingual Embeddings
Word level alignment
Sentence level alignment
Document level alignment
Mapping-based approaches
Pseudo-mutli-lingual corpora based approaches
Joint methods
Regression methods
Orthogonal methods
Canonical methods
Margin methods
Word-alignment based matrix factorization approaches
Compositional sentence models
Bilingual autoencoder models
Bilingual skip-gram models
Other sentence level approaches
Concept-based methods
Extensions of sentence-aligned models
Approaches based on pseudo-bilingual document-aligned corpora

Figure 6.3: An overview of categorization of methods for creating CLWE according to Ruder et al. (2019).

Let us define monolingual vector spaces and a dictionary of translated pairs of words. The dictionary $D$ contains $n$ translated pairs of words (called *seed words* or *seed dictionary*) $((w_1^s, w_1^t), (w_2^s, w_2^t), \ldots, (w_n^s, w_n^t))$. Vector space of the source language $s$ is represented by a matrix $\mathbf{X}^s \in \mathbb{R}^{d \times n}$ and vector space of the target language $t$ is represented by a matrix $\mathbf{X}^t \in \mathbb{R}^{d \times n}$ where $n$ is the size of the seed dictionary and $d$ is a dimension of the vector spaces. Each word $w_i$ from the dictionary $D$ is in matrices $\mathbf{X}^s$ and $\mathbf{X}^t$ represented by vectors $\mathbf{x}_i^s$, $\mathbf{x}_i^t$, respectively.

The goal of the linear transformation is to find a transformation matrix $\mathbf{W}^{s \to t} \in \mathbb{R}^{d \times d}$ that projects the semantic space $\mathbf{X}^s$ of the *source language* into the semantic space $\mathbf{X}^t$ of the *target language*. The transformed source space is represented by $\widehat{\mathbf{X}}^s$ and is obtained through matrix multiplication as shown below:

$$\widehat{\mathbf{X}}^s = \mathbf{W}^{s \to t} \mathbf{X}^s \tag{6.1}$$

Any word vector from the source space that is not present in $\mathbf{X}^s$ can be transformed into the target space by multiplication with the transformation matrix $\mathbf{W}^{s \to t}$.

### 6.2.1.1 Mean Squared Error Transformation

The transformation method described in Mikolov, Le, and Sutskever (2013) estimates the transformation matrix by minimizing the mean squared error (**MSE**) between the pairs of vectors $(\mathbf{x}_i^s, \mathbf{x}_i^t)$ for the corresponding word from the dictionary $D$. This method minimizes the MSE by finding the transformation matrix that produces the smallest error between the source and target language word vectors. This method belongs under the *regression methods*. The MSE is calculated as follows:

$$MSE = \sum_{i=1}^{n} \left\| \mathbf{W}^{s \to t} \mathbf{x}_i^s - \mathbf{x}_i^t \right\|^2 \tag{6.2}$$

#### 6.2.1.2 Orthogonal Transformation

The orthogonal transformation method (**Orto**) constrains the transformation matrix $\mathbf{W}^{s \to t}$ to be orthogonal, meaning that it is a square matrix with columns and rows that are orthonormal vectors ($\mathbf{W}^\top \mathbf{W} = \mathbf{W}\mathbf{W}^\top = I$, where $I$ is the identity matrix). This method has the same objective as the MSE transformation but with the added orthogonality restriction. The optimal transformation matrix $\mathbf{W}^{s \to t}$ can be computed using Singular Value Decomposition (SVD) as follows:

$$\mathbf{W}^{s \to t} = \mathbf{V}\mathbf{U}^\top \tag{6.3}$$

where matrices $\mathbf{V}$ and $\mathbf{U}$ are computed with SVD as $\mathbf{X}^{t^\top} \mathbf{X}^s = \mathbf{U}\Sigma\mathbf{V}^\top$ as described in Artetxe et al. (2016). The orthogonality constraint ensures that the transformation does not squeeze or re-scale the transformed space but rather only rotates it. This helps to preserve the relationships between words (vectors) in the space, particularly the angles between words and, thus, the similarity between words in the transformed space.

#### 6.2.1.3 Canonical Transformation

The Canonical methods are based on *Canonical Correlation Analysis* (**CCA**), which provides a way of measuring a linear relationship between two multivariate variables (i.e., vectors) (Brychcín, 2020). The method aligns monolingual vector spaces $\mathbf{X}^s$ and $\mathbf{X}^t$ to a shared space represented by matrix $\mathbf{Y}^o$ (Ruder et al., 2019). To achieve this, CCA computes transformation matrices $\mathbf{W}^{s \to o}$ for the source language and $\mathbf{W}^{t \to o}$ for the target language that maps the spaces into the shared space $\mathbf{Y}^o$. These transformation matrices can be computed analytically using SVD (Hardoon et al., 2004)[3] or by minimizing the negative correlation (denoted as $Ncor$) between the source language vectors $\mathbf{x}_i^s$ and target language vectors $\mathbf{x}_i^t$ projected into the shared space $\mathbf{Y}^o$. The negative correlation is given by:

$$Ncor = -\sum_{i=1}^{n} \rho(\mathbf{W}^{s \to o} \mathbf{x}_i^s, \mathbf{W}^{t \to o} \mathbf{x}_i^t) \tag{6.4}$$

The correlation between the transformed source language vectors $\mathbf{W}^{s \to o} \mathbf{x}_i^s$ and the transformed target language vectors $\mathbf{W}^{t \to o} \mathbf{x}_i^t$ is computed using the following equation:

[3] We use this approach in our experiments.

$$\rho(\mathbf{W}^{s\to o}\mathbf{x}_i^s, \mathbf{W}^{t\to o}\mathbf{x}_i^t) = \frac{cov(\mathbf{W}^{s\to o}\mathbf{x}_i^s, \mathbf{W}^{t\to o}\mathbf{x}_i^t)}{\sqrt{var(\mathbf{W}^{s\to o}\mathbf{x}_i^s) \times var(\mathbf{W}^{t\to o}\mathbf{x}_i^t)}} \tag{6.5}$$

where $cov$ is the covariance and $var$ is the variance. The CCA method was first used by Faruqui and Dyer (2014) to map two monolingual word embedding spaces into a cross-lingual space and was later extended to multiple languages by Ammar et al. (2016). Using the approach from Ammar et al. (2016), we can modify the CCA method to transform only the source space into the target space without using the shared space. The transformation matrix $\mathbf{W}^{s\to t}$ can be then computed as follows:

$$\mathbf{W}^{s\to t} = \mathbf{W}^{s\to o}(\mathbf{W}^{t\to o})^{-1} \tag{6.6}$$

### 6.2.1.4 Ranking Transformation

The *Ranking Transformation* (**Rank**) method (Lazaridou et al., 2015) uses the *max-margin hinge loss* (MML) instead of MSE to address and reduce the *hubness* problem (Radovanović et al., 2010). The goal of this method is to rank the correct translations of a word $w_i$ (vectors $\mathbf{x}_i^s$ and $\mathbf{x}_i^t$) higher than random translations (negative examples) of the same word $w_i$ (vectors $\mathbf{x}_i^s$ and $\mathbf{x}_j^t$). The optimization goal is to minimize the following function:

$$MML = \sum_{i=1}^{n}\sum_{j\neq i}^{k} \max\{0, \gamma - \cos(\mathbf{W}^{s\to t}\mathbf{x}_i^s, \mathbf{x}_i^t) + \cos(\mathbf{W}^{s\to t}\mathbf{x}_i^s, \mathbf{x}_j^t)\} \tag{6.7}$$

where $\gamma$ and $k$ are hyper-parameters representing the margin and the number of negative examples, respectively.

### 6.2.1.5 Orthogonal Ranking Transformation

Brychcín (2020) combined the orthogonal and ranking transformations to create a new method called the *Orthogonal Ranking Transformation* (**Or-Ra**). This method aims to both keep the transformation matrix $\mathbf{W}^{s\to t}$ orthogonal and reduce hubness. The objective function and details of this method can be found in Brychcín (2020).

### 6.2.1.6 Other Transformations

We are aware that there are other methods (Adams et al., 2017; Lample et al., 2018; Xiao & Guo, 2014; Zou et al., 2013) to align semantic spaces, but we decided to use the five named methods for our experiments. For example, in the VecMap method (Artetxe et al., 2018), the authors proposed an unsupervised approach to automatically induct the dictionary $D$ based on the observation that two equivalent words in different languages should have a similar distribution. Using the observation, they induct an initial dictionary, which they then iteratively improve. They use the orthogonal transformation to map the semantic spaces.

We decided to incorporate the first four methods (MSE, Orto, CCA, Rank) for our experiments in this thesis because Ruder et al. (2019) divided the methods into the same four categories[4]. Based on Ruder et al. (2019), we consider these four methods to be the principal methods for cross-lingual word embeddings. The other methods, such as VecMap (Artetxe et al., 2018), are based or built on top of these basic methods and, as shown in Ruder et al. (2019), although they are different methods, they optimize very similar objectives with differences in used data and regularization. Additionally, we introduced a fifth method (Or-Ra), that serves as a complement to these essential methods. The inclusion of this method was motivated by our favorable prior experiences with it and our intention to subject it to comparative analysis alongside the initial four methods in the context of CLSA.

## 6.3 Multilingual Transformer-based Models

The cross-lingual capability of multilingual Transfomer-based models like XLM-R (Conneau et al., 2020) is learned during the pre-training phase thanks to the similarity of words between languages. Some words are universal or have quite similar surface forms across languages, e.g., numbers, words like *coffee, metro, football* and others. These words usually appear in the same contexts across all languages. Thus, the model learns to align the information or meaning between different languages. It helps the model develop a shared understanding of transferable linguistic patterns and representations across languages.

In other words, the model learns to project words or tokens from different languages into a shared embedding space. This shared space allows similar or semantically related words in different languages to have similar representations. Consequently, the model can leverage the similarities and transfer knowledge across languages, enabling cross-lingual capabilities. The alignment is done implicitly during the pre-training compared to cross-lingual embeddings, where the alignment is performed explicitly, i.e., by learning some additional function that can project or align different word embeddings into one shared cross-lingual space.

The usage of the multilingual model for cross-lingual sentiment classification does not differ from the monolingual approach. Unlike cross-lingual embeddings, here, the model is trained on data in the source language (e.g., English) and can be directly used or evaluated on data from any other languages that the multilingual model supports.

## 6.4 Related Work for Cross-lingual Sentiment Analysis

In recent years, much less research has been devoted to CLSA compared to the monolingual task. As we mentioned in Section 6.1.2, the CLSA approaches can be roughly divided into three groups: (1) *machine translation* (MT), (2) *explicit methods for knowledge transfer between*

[4] We use slightly different names than Ruder et al. (2019).

*languages, including linear transformations* (EM/LT) and (3) *multilingual Transformer-based models* (EMB)

Table 6.1 provides an overview of selected works devoted to CLSA. The table compares various aspects of these works, including employed machine learning models, methods for knowledge transfer between languages[5], data domains and involved languages. Earlier approaches rely on SVM or logistic regression classifiers, usually using machine translation. Along with the evolution of neural networks, we saw the integration of CNN or LSTM models in combination with explicit methods for transferring knowledge between languages, such as linear transformations. With the advent of Transformer-based models, multilingual versions with inherent and embedded cross-lingual capabilities gained prominence in CLSA. Most recently, the LLMs[6] have been introduced and used for SA, although for now, almost exclusively for English.

The works are mostly focused on the movie, product or restaurant review domains with two or three classes, infrequently with more fine-grained class partition. Furthermore, English is the dominant resource-rich source language across most cases, while the target languages are predominantly French, Chinese or Spanish. Occasionally, other languages are incorporated into the analysis, reflecting a limited but growing exploration of diverse linguistic contexts within CLSA research.

The early works in CLSA relied on machine translation (Balahur & Turchi, 2012, 2014; Can et al., 2018; Sharma, 2020; X. Wan, 2008, 2009; P. Zhou et al., 2016). They usually translated or exploited data from English to obtain training data for another language. Ghorbel (2012) translated the lexical resource of SenitWordNet from English to French to improve the performance of SA in French. Sazzed (2020) compiled a large manually annotated dataset of Bengali reviews, translated it into English and compared different methods, including supervised machine learning classifier, unsupervised approach and transfer learning for CLSA.

Barriere and Balahur (2020) used multilingual BERT-like models and machine translation to augment a dataset to improve results of Twitter SA in French, Spanish, German and Italian. In W. Zhang, He, et al. (2021), the authors focused on cross-lingual ABSA by using multilingual BERT-like models and data augmentation. The closely related work to ours can be found in Thakkar et al. (2021), where the authors use the neural machine translation encoder-based model and English data to perform zero-shot cross-lingual sentiment classification on French. Eriguchi et al. (2018) performed the zero-shot classification from Slovene to Croatian.

Jain and Batra (2015) employed a recursive autoencoder architecture and sentence-aligned corpora of Hindi and English and evaluated the system on the Hindi movie reviews dataset. H. Zhou et al. (2015) proposed a method for creating cross-lingual word embeddings specifically for SA. They used an SVM classifier based on these embeddings for the

[5] Although linear transformations belong under the explicit transfer methods due to their importance in our work, we list this approach separately in Table 6.1 with the *LT* abbreviation.

[6] Despite the fact that the listed works use LLMs for English only, we include them in our cross-lingual overview as they also have cross-lingual capabilities and we use them for Czech and French in our experiments.

| Source | Approach | Method (Model) | Transfer Method | Data Domain | #Classes | Languages |
|---|---|---|---|---|---|---|
| X. Wan (2009) | ML | SVM | MT | Product reviews | 2 | EN, CN |
| Balahur and Turchi (2014) | ML | SVM | MT | News articles | 2 | EN, FR, DE, ES |
| H. Zhou et al. (2015) | ML | SVM | EM | Product reviews | 2 | EN, CN |
| Barnes et al. (2016) | ML | SVM | EM, MT, LT | Hotel reviews | 2 | EN, ES |
| Abdalla and Hirst (2017) | ML | Log. regression | LT | Restaurant reviews | 5 | EN, CN |
| Can et al. (2018) | NN | GRU | MT | Restaurant reviews | 2 | EN, ES, RU, NL, TR |
| Eriguchi et al. (2018) | NN | LSTM | MT | Product reviews | 2 | EN, FR |
| Barnes et al. (2018) | NN | Feedforward neural net. | EM, LT | Hotel reviews | 4 | ES, CA, EU |
| X. Dong and de Melo (2018) | NN | CNN | EM, LT | Movie reviews | 2 | EN, FR |
| X. Chen et al. (2018) | NN | Adversial neural net. | EM | Hotel reviews | 5 | EN, CN |
| Sharma (2020) | ML | Log. regression | MT | Tweets | 2 | EN, FR |
| Sazzed (2020) | ML | SVM | MT | Youtube reviews | 2 | EN, BN |
| Barriere and Balahur (2020) | T | multilingual BERT-like | EMB | Tweets | 2 | EN, FR, ES, DE, IT |
| Aliramezani et al. (2020) | NN | GRU | EM, LT | Food reviews | 2 | EN, FA |
| Kuriyozov et al. (2020) | NN | GRU | EM, LT | Hotel reviews | 2 | TR, UZ |
| Přibáň and Steinberger (2021) | T | multilingual BERT-like | EMB | Movie reviews,<br>Facebook posts,<br>Product reviews | 2-3 | EN, CS |
| W. Zhang, He, et al. (2021) | T | multilingual BERT-like | EMB | Restaurant reviews | 2 | EN, FR, ES, NL, RU |
| Thakkar et al. (2021) | T | multilingual BERT-like | EMB | News articles | 3 | EN, SL, HR |
| C. Wang and Banko (2021) | T | multilingual BERT-like | EMB | Product reviews | 2 | EN, FR, DE, JA |
| Přibáň et al. (2022) | NN | CNN, LSTM | LT | Movie reviews | 2-3 | EN, CS, FR |
| Catelli et al. (2022) | T | multilingual BERT-like | EMB | Hotel reviews | 2 | EN, IT |
| Qin et al. (2023) | T | LLMs | EMB | Movie reviews | 2 | EN |
| Zhong et al. (2023) | T | LLMs | EMB | Movie reviews | 2 | EN |
| Han et al. (2023) | T | LLMs | EMB | Hotel reviews | 3 | EN |
| W. Zhang et al. (2023) | T | LLMs | EMB | Hotel reviews | 3 | EN |

Table 6.1: Overview of selected works focused on CLSA. We classify different machine learning approaches in the **Approach** column as classical machine learning (ML), neural networks (NN) and Transformers (T). The column **Method (Model)** denotes the type of machine learning model or method for classification. The type of method for knowledge transfer between languages is presented under the column **Transfer Method** as machine translation (MT), explicit method (EM), linear transformation (LT) and embedded (EMB). The **Languages** column enumerates the involved languages using ISO 639-1 codes.

task of polarity classification. Barnes et al. (2016) compared multiple techniques for cross-lingual aspect-based sentiment classification, including the technique from Mikolov, Le, and Sutskever (2013). Abdalla and Hirst (2017) used the same least square linear transformation method from Mikolov, Le, and Sutskever (2013) to conduct experiments on English, Spanish and Chinese. Barnes et al. (2018) introduced an approach for creating bilingual sentiment word embeddings. These embeddings are optimized to represent semantic information in the source and target languages using a small bilingual dictionary and sentiment information extracted only from the source language. The effectiveness of this approach was demonstrated through comparison with other cross-lingual methods. X. Dong and de Melo (2018) presented an algorithm to cross-lingually project word vector information to other languages and transfer sentiment information across languages. They used a CNN for classification and they evaluated their approach on nine languages, including French. X. Chen et al. (2018) trained an adversarial neural network with bilingual embeddings for polarity classification in Arabic and Chinese only with English training data. Similarly to our work, Aliramezani et al. (2020) and Kuriyozov et al. (2020) used linear transformations for Persian and Turkish, respectively.

C. Wang and Banko (2021) compared multiple Transfomer-based models for monolingual and cross-lingual text classification tasks in an industry setting for various languages, including Japanese, German or Spanish. In Přibáň and Steinberger (2021), we introduced the Czech dataset for subjectivity classification, which was later used to test zero-shot subjectivity classification between English and Czech with multilingual Transformer-based models. The authors in Winata et al. (2022) studied possibilities for cross-lingual classification for languages unseen during pre-training of Transformer-based models. They analyze the effectiveness of several few-shot learning strategies for the zero-shot classification of unseen languages. Catelli et al. (2022) utilized the mBERT to perform cross-lingual sentiment classification of TripAdvisor reviews between English and Italian. A more detailed overview of cross-lingual and multilingual SA can be found in Agüero-Torales et al. (2021), Farra (2019), and Y. Xu et al. (2022).

In Přibáň and Steinberger (2021), we demonstrated the effectiveness of huge multilingual BERT-like models for cross-lingual SAw between Czech and English. In the study from Přibáň et al. (2022, 2024), we evaluated the usage of linear transformations for zero-shot CLSA between Czech, French and English. In Chapter 7, we describe these publications and our contributions in detail.

# PART III

# Thesis Contributions

# 7 Cross-lingual Sentiment Classification

The main contribution of this thesis is focused on cross-lingual sentiment analysis (CLSA), more specifically on *zero-shot cross-lingual sentiment classification*. We have already described the theory necessary to understand the problem of CLSA in Chapter 6, but it is important to remember that in the zero-shot settings, we use annotated data only from one language (source) while performing the sentiment classification on data from the second language (target).

We use two approaches, namely modern multilingual Transformer-based models and linear transformations in conjunction with CNN and LSTM neural networks and we evaluate their performance on Czech, French, and English datasets. We aim to compare and assess the models' ability to transfer knowledge across languages and discuss the trade-off between their performance and training/inference speed. To establish robust benchmarks, we build strong monolingual baselines for all languages comparable with the current SotA approaches, achieving state-of-the-art results in Czech (96.0% accuracy) and French (97.6% accuracy). Further, we compare these models with the cross-lingual models and the latest large language models (LLMs), such as Llama 2 and ChatGPT.

We show that the large multilingual Transformer-based XLM-R model consistently outperforms all other cross-lingual approaches in zero-shot cross-lingual sentiment classification. It surpasses them by at least 3%, but a difference larger than 5% is not unusual. The large XLM-R model also performs close to monolingual results, proving its great capability to transfer knowledge between languages for the SA task. Next, we show that the smaller Transformer-based models are comparable in performance to older but much faster cross-lingual approaches with linear transformations. For example, the best-performing cross-lingual LSTM model with linear transformation trained on English data achieved an accuracy of 92.1% on the French dataset, compared to the smaller XLM-R model's accuracy of 90.3%. Remarkably, this performance is achieved with just approximately 0.01 of the training time required for the smaller XLM-R model. This underscores the potential of linear transformations as a pragmatic alternative to resource-intensive and slower Transformer-based models in real-world applications. The LLMs achieved remarkable results compared to the large XLM-R model. The results are on par or better, at least by 1% – 3%, but with significant additional hardware requirements and limitations.

To the best of our knowledge, there is no prior work that simultaneously and adequately compared in detail these two crucial aspects: task performance and training and inference times of Transformer-based models and their older counterparts. While some studies such as (Park et al., 2022) focus solely on comparing the relative speed-ups of Transformer-based models, and others like (Karita et al., 2019; Lakew et al., 2018; Zeyer et al., 2019) analyze task performance differences between Transformer-based models and older models based on CNN and LSTM neural networks, none of these investigations is focused on cross-lingual sentiment and does not offer the detailed, dedicated and thorough examination that we provide.

Overall, we contribute to the understanding of CLSA and provide valuable insights into the strengths and limitations of the cross-lingual approaches for SA. We see and highlight our key contributions as follows:

1. *We propose and evaluate approaches that deal with CLSA in Czech, French and English, showing their great ability to transfer knowledge between the languages.*

2. *We compare the performance and speed of these models and place these two aspects in a common context. We examine and compare the important aspects of training and inference speed between the two approaches, as well as their potential limitations when applied in real-world scenarios.*

3. *Based on the extensive experiments, we propose a set of recommendations for the configuration and usage of linear transformations for the CLSA task.*

The work described in this chapter is mainly based on three publications: "Are the Multilingual Models Better? Improving Czech Sentiment with Transformers" (Přibáň & Steinberger, 2021), "Linear Transformations for Cross-lingual Sentiment Analysis" (Přibáň et al., 2022) and "A comparative study of cross-lingual sentiment analysis" (Přibáň et al., 2024). We released all our resources and source codes[1] freely for research purposes.

Additionally, our study encompassed an assessment of multilingual systems designed for sentiment classification across various languages employed in practical applications. This evaluation is presented in the paper titled "Comparative Analyses of Multilingual Sentiment Analysis Systems for News and Social Media" (Přibáň & Balahur, 2023). Furthermore, we conducted a series of cross-lingual experiments targeting subjectivity classification, as documented in the publication "Czech Dataset for Cross-lingual Subjectivity Classification" (Přibáň & Steinberger, 2022). Given the nuanced distinctions between these two works and the cross-lingual sentiment classification, we decided to include them in Chapter 8.

This chapter is organized as follows. The limitations and challenges of CLSA are discussed in Section 7.1. We introduce the data and datasets used in our experiments in Section 7.2. In Section 7.3, we describe the models for classification. Section 7.4 is focused on methodology and experimental setup. We put monolingual results in a separate Section 7.5. The core

[1] The resources and source codes are available at https://github.com/pauli31/linear-transformation-4-cs-sa.

cross-lingual results, findings and comparisons are presented in Section 7.6. Supplementary experiments are placed in Section 7.7. The discussion and the implications of our findings are included in Section 7.8. Finally, we provide the conclusion in Section reflabel:conclusion. The related work for CLSA was already discussed in Section 6.4, thus, we do not focus on it here.

## 7.1 Challenges and Limitations

As previously outlined, we categorize cross-lingual approaches for SA into three key categories: *machine translation*, *explicit methods for knowledge transfer between languages* and *multilingual Transformer-based models*.

The earlier cross-lingual approaches, often reliant on machine translation, exhibit inferior results compared to monolingual counterparts. The weakness of the approach utilizing machine translation is the required system for machine translation itself since it can be slow, expensive, unsatisfactory or, in some cases, even unavailable. As we show, explicit methods, like linear transformations, are much faster and require only a fraction of computational resources compared to Transformer-based models at the cost of not achieving SotA results. The newest LLMs in zero-shot settings provide similar results for the polarity detection task compared to the SotA results obtained by fine-tuned Transformer-based models. However, these SotA results are redeemed by the tremendous computational resources that are required. It should also be noted that, as shown in W. Zhang et al. (2023), the LLMs are significantly outperformed in more complex tasks such as ABSA by the fine-tuned models.

The results of existing cross-lingual methods can hardly be compared with each other because each work usually uses a different dataset or pairs of languages. In contrast, our study compares different cross-lingual methods in three languages both in terms of accuracy and their training and inference speed. The existing cross-lingual works are usually restricted only to English, French, Spanish or Chinese and are merely dedicated to accuracy while completely ignoring other aspects, such as training or inference speed, which are crucial in real-world deployment.

Another limitation of the very recent works with LLMs is their exclusive focus on SA evaluations conducted nearly solely for English. To the best of our knowledge, we provide the first SA results for LLMs in French and Czech, expanding the scope of cross-lingual evaluations.

## 7.2 Data for Experiments

This section describes the polarity detection datasets we used in our experiments. We also provide information about pre-trained word embeddings required for the CLSA with a linear transformation approach. Additionally, we cover building bilingual dictionaries needed for linear transformations. At last, we introduce a dataset for the word analogies task, specifically designed to evaluate the linear transformations, see Section 7.2.5.

### 7.2.1 Polarity Detection Datasets

For our experiments, we utilize four publicly available datasets with binary polarity labels (*positive* and *negative*) from the movie review domain in English, Czech and French. We also include the third *neutral* label for the Czech and one English dataset. Table 7.1 shows details about the datasets.

- `IMDB` (Maas et al., 2011): This English dataset consists of 50k movie reviews obtained from the Internet Movie Database[2] with *positive* and *negative* classes. The reviews were split into training and testing sets of equal size. We selected a random subset of 2.5k examples from the training set as development data.

- `SST-2` (Socher et al., 2013): This English dataset contains around 12k manually annotated movie reviews split into two categories (*positive* and *negative*) with training, testing, and development sets. It also has a fine-grained version (SST-5) with five labels (*very positive/negative, positive, negative, neutral*). To create the `SST-3` dataset, we merged[3] the labels *positive* and *very positive* into one class, analogously the *negative* and *very negative* labels into one class, resulting in a dataset with three classes (*positive, negative, neutral*).

- `CSFD` (Habernal et al., 2013): The Czech CSFD dataset consists of 90k movie reviews from the Czech movie database[4] that were annotated according to their star rating (0–1 stars as *negative*, 2-3 stars as *neutral*, 4–5 stars as *positive*). We use both versions of the dataset: a) `CSFD-2` – only the examples labeled as *positive* or *negative*, and b) `CSFD-3` – all examples with *positive*, *negative*, and *neutral* labels. The data was split according to the scheme used in Přibáň and Steinberger (2021).

- `Allocine` (Théophile, 2020): This dataset consists of 100k positive and 100k negative movie reviews scraped from the Allociné[5] website and annotated in the same way as the CSFD dataset. The reviews were divided into three balanced training, testing, and development sets.

### 7.2.2 Word Embeddings

In our experiments with linear transformations, we employed two types of pre-trained fastText (Bojanowski et al., 2017) word embeddings: a) existing *fastText* embeddings trained on a corpus of Common Crawl and Wikipedia texts[6] and b) *in-domain* embeddings that we trained on the text from the training parts of the sentiment datasets. We trained separate

[2] https://www.imdb.com

[3] The SST-2 dataset was created in the same way, but the examples with the *neutral* label are omitted.

[4] https://www.csfd.cz

[5] https://www.allocine.fr

[6] https://fasttext.cc/docs/en/crawl-vectors.html

| | **IMDB** (English) | | | | **SST** (English) | | | |
|---|---|---|---|---|---|---|---|---|
| | train | dev | test | total | train | dev | test | total |
| Positive | 11,242 | 1,258 | 12,500 | 25,000 | 3,610 | 444 | 909 | 4,963 |
| Negative | 11,258 | 1,242 | 12,500 | 25,000 | 3,310 | 428 | 912 | 4,650 |
| Neutral | - | - | - | - | 1,624 | 229 | 389 | 2,242 |
| Total | 22,500 | 2,500 | 25,000 | 50,000 | 8,544 | 1,101 | 2,210 | 11,855 |
| | **CSFD** (Czech) | | | | **Allocine** (French) | | | |
| | train | dev | test | total | train | dev | test | total |
| Positive | 22,117 | 2,456 | 6,324 | 30,897 | 79,413 | 9,796 | 9,592 | 98,801 |
| Negative | 21,441 | 2,399 | 5,876 | 29,716 | 80,587 | 10,204 | 10,408 | 101,199 |
| Neutral | 22,235 | 2,456 | 6,077 | 30,768 | - | - | - | - |
| Total | 65,793 | 7,311 | 18,277 | 91,381 | 160,000 | 20,000 | 20,000 | 200,000 |

Table 7.1: Polarity detection datasets statistics.

embeddings for each language, with the English embeddings being created by concatenating the texts from the SST and IMDB datasets. We used the skip-gram algorithm and the gensim library (Řehůřek & Sojka, 2010) to train the embeddings, applying lowercasing and filtering out words with a frequency below 5. The training was conducted for 15 epochs. In all experiments, we used word embeddings with a dimension of 300.

An important note is that we used only a small portion of plain text (approximately 50MB for English and Czech and 100MB for French) to pre-train the in-domain embeddings, compared to the gigabytes of plain text from Wikipedia used to create the existing general fastText embeddings. This means that our in-domain embeddings were pre-trained on a significantly smaller dataset than the general embeddings. As shown in the results in Section 7.6.1, our in-domain embeddings still yielded favorable results in the CLSA task despite this limitation.

#### 7.2.2.1 Normalization of Vectors

As shown in Artetxe et al. (2016) and Brychcín et al. (2019), normalizing the word vectors usually leads to improved results for the linear transformations. We follow the approaches from Artetxe et al. (2016) and Brychcín et al. (2019) by using dimension-wise mean centering of the semantic space (centering the space around the mean for each dimension). The mean centering of a word vector $\mathbf{x}$ from space a $\mathbf{X}$ (word embeddings) results in a vector $\widetilde{\mathbf{x}}$ that is obtained by the following equation:

$$\widetilde{\mathbf{x}} = \mathbf{x} - \bar{\mathbf{X}} \tag{7.1}$$

where $\bar{\mathbf{X}}$ is the mean of $\mathbf{X}$ along each dimension.

Next, we normalize each word vector $\mathbf{x}$ to be a unit vector (to have a unit length), so all training instances contribute equally to the optimization goal (Artetxe et al., 2016). The unit vector $\hat{\mathbf{x}}$ can be computed as follows:

$$\hat{\mathbf{x}} = \frac{\mathbf{x}}{||\mathbf{x}||} \tag{7.2}$$

where $||\mathbf{x}||$ is the Euclidean norm.

### 7.2.3 Bilingual Dictionaries

The linear transformations require bilingual dictionaries to align the semantic spaces, see Section 6.2. We obtained these dictionaries by translating the 40k most common words from the CSFD dataset into English and French using Google Translate[7]. We repeated this process for the IMDB and Allocine datasets. Some translation errors were identified and manually corrected. The dictionaries are available at our GitHub repository[1].

### 7.2.4 Word Analogy Dataset

We use word analogies task for intrinsic evaluation, described in Section 7.2.5. The word analogies dataset presented in Brychcín et al. (2019) consists of two types of analogies: *semantic* and *syntactic* and includes nine categories for English and Czech. The semantic part of the dataset focuses on the meaning of the words and includes three categories: *capital-common-countries*, *family*, and *state-currency*. The syntactic part of the dataset is divided into six categories: *adjective-comparative*, *adjective-opposite*, *adjective-superlative*, *noun-plural*, *state-adjective* and *verb-past-tense*. The performance of the evaluated linear transformation is calculated as the average accuracy score across all nine categories.

### 7.2.5 Evaluation of Linear Transformations

There are two main types of tasks to evaluate the quality of cross-lingual embeddings: *intrinsic* and *extrinsic* tasks (Brychcín et al., 2019; Ruder et al., 2019). Extrinsic tasks evaluate the CLWE on downstream tasks or other NLP tasks where the CLWE can be applied. Cross-lingual sentiment classification is an example of extrinsic evaluation.

The intrinsic evaluation aims to evaluate certain abilities of the CLWE, for example, semantic or syntactic relationships between words. The good performance of a method on the intrinsic task does not directly imply a good performance for a downstream task. Common examples of an intrinsic task are the word similarity or word analogy tasks (Brychcín, 2020; Brychcín et al., 2019; Ruder et al., 2019). We use the word analogy task to evaluate the performance of the five selected linear transformations, i.e., *Mean Squared Error Transformation* (MSE), *Orthogonal Transformation* (Orto), *Canonical Transformation* (CCA), *Ranking Transformation* (Rank) and *Orthogonal Ranking Transformation* (Or-Ra), introduced in Section 6.2.1.

[7] https://translate.google.com

The word analogy task involves questions of the form: word $w_1$ is in a relationship to $w_2$ as word $w_3$ is in the same relationship to $w_4$, where the goal is to predict $w_4$ (Brychcín et al., 2019). For instance, the word pair in Czech *Paříž* (*Paris*) and *Francie* (*France*) has the same relationship as the word pair *Madrid* and *Spain* in English. In this case, the relationship is between the capital city and the corresponding country and the goal is to predict the word *Spain*.

Formally, we can label analogy word pair from the source space $\mathbf{X}^s$ as $(w_1^s, w_2^s)$ and word (vector) from the target language vector space $\mathbf{X}^t$ as $w_3^t$. To find the searched word (vector) $w_4^t$, we first approximate the searched vector $\mathbf{v}$ using equation 7.3, where $\hat{\mathbf{X}}^s$ is the source space $\mathbf{X}^s$ transformed into the target space $\mathbf{X}^t$.

$$\mathbf{v} = \hat{\mathbf{X}}^s(w_2^s) - \hat{\mathbf{X}}^s(w_1^s) + \mathbf{X}^t(w_3^t) \tag{7.3}$$

Then, we search in the target space $\mathbf{X}^t$ for the most similar word (vector) to the vector $\mathbf{v}$. The similarity between the vector $\mathbf{v}$ and the word $w^t$ from semantic space is computed using the cosine similarity. The target vector $\hat{w}_4^t$ is then estimated using equation 7.4.

$$\hat{w}_4^t = \arg\max_{w^t} \frac{\mathbf{X}^t(w^t) \cdot \mathbf{v}}{\|\mathbf{X}^t(w^t)\|_2 \cdot \|\mathbf{v}\|_2}. \tag{7.4}$$

If the found vector $\hat{w}_4^t$ is equal to the vector $w_4^t$, then the answer is recorded as correct. The accuracy metric is used.

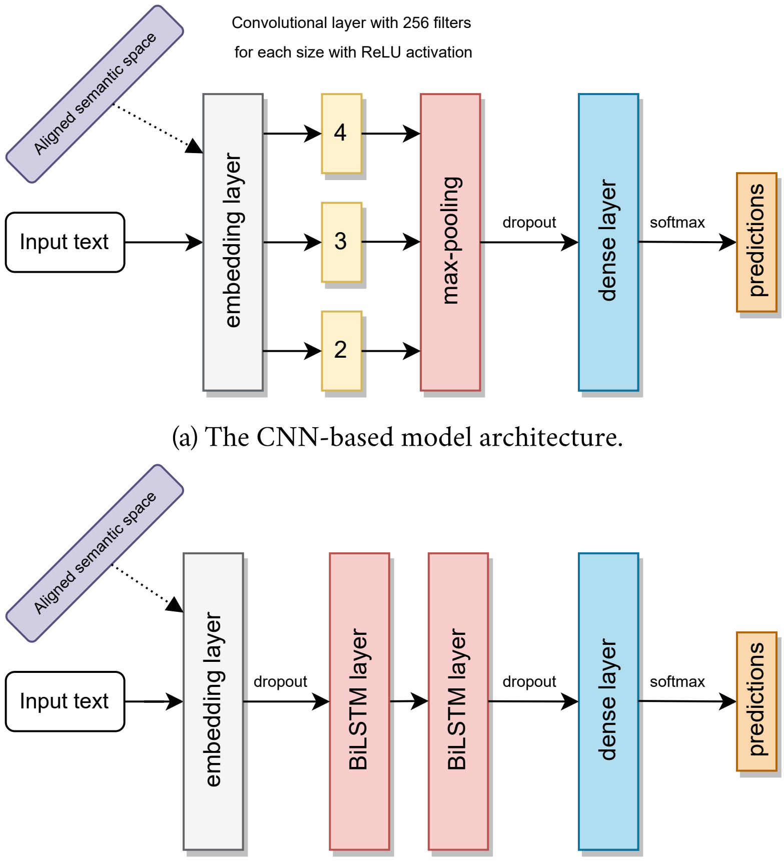
## 7.3 Classification Models

Hereafter, we introduce classification models for our experiments: a) neural network classification models used with linear transformations and b) the models based on the Transformer architecture.

### 7.3.1 Models for Linear Transformations

We utilize two neural network models, a CNN-based model and an LSTM-based model, to perform cross-lingual polarity detection using linear transformations. During the experiments, the transformations are applied to obtain cross-lingual embeddings, which are then used to represent the input text for the mentioned models.

The `CNN` model is based on the architecture proposed by Y. Kim (2014), and includes a single convolutional layer applied to the word embeddings. See Figure 7.1a for an illustration of the model's architecture. The input text is transformed into a matrix with dimensions $n \times d$, where $n$ is the length of the text and $d = 300$ is the dimensionality of the word embeddings. This matrix is then processed using 1-dimensional convolution with filter sizes of 2, 3, and 4 (256 filters for each size), followed by ReLU activation and max-over-time pooling. The output is concatenated and passed through a fully-connected layer that produces prediction scores for each class. The class with the highest score is selected as the final class prediction. To prevent overfitting, a dropout (Srivastava et al., 2014) of 0.5 is applied before the fully-connected layer. The model has around 700k parameters.



(a) The CNN-based model architecture.

(b) The LSTM-based model architecture.

Figure 7.1: The figure illustrates the architecture neural network models for the CLSA task with linear transformations.

The `LSTM` model has the same input embedding layer that converts the input text into a matrix with dimensions identical to the CNN model input matrix. This matrix is then processed through two BiLSTM layers, each with 512 units (hidden size). The output is then passed into a fully-connected layer to predict the polarity classes. A dropout rate of 0.5 is also applied before the fully-connected layer. The model has around 1.6M parameters.

#### 7.3.1.1 Training Details

During the training of the models, the embeddings layer is kept frozen, meaning that the embeddings are not fine-tuned. For out-of-vocabulary words, we use the ability of fastText embeddings to generate vectors for unknown words. We train our model using the Adam (Kingma & Ba, 2015) optimizer with a constant learning rate or linear learning rate decay.

The learning rates are set to 1e-3 or 1e-4. We use a batch size of 32. Training is conducted for a maximum of 10 epochs. We randomly shuffle training data before each epoch.

We tokenize the English and Czech text with the *MorphoDiTa* (Straková et al., 2014) tool from the *CorPy*[8] library. For French text, we use the NLTK (Bird & Klein, 2009) tokenizer[9]. We lowercase the input text after the tokenization.

## 7.3.2 Transformer-based Models

The second type of models we use for polarity detection are BERT-like models based on the Transformer (Vaswani et al., 2017) architecture, namely BERT (Devlin et al., 2019). These models are pre-trained on various modified language modeling tasks, usually *Masked Language Modeling* (MLM). The exact pre-train procedure may differ, but in general, it is always some language modeling objective. The pre-train model is then fine-tuned on the target downstream task, which, in our case, is polarity detection. For our purposes, we can divide the models into two groups: *monolingual* and *multilingual*. A monolingual model is pre-trained on a single language and can only be used for that language, while a multilingual model is pre-trained on multiple languages and can be used for all of them. The multilingual property allows us to train the model with data in the source language and evaluate it on data in the target language and thus obtain a cross-lingual model. The models differ in the number of parameters, vocabulary size and number of supported languages, as shown in Table 7.2. In the table, we also include the sizes of the LSTM and CNN models we use.

| Type | Model | Parameters | Vocab | #Langs |
|---|---|---|---|---|
| Czech | Czech Electra | 13M | 30k | 1 |
| | Czert-B | 110M | 30k | 1 |
| | RobeCzech | 125M | 52k | 1 |
| French | CamemBERT | 110M | 32k | 1 |
| English | BERT$_{\text{Base-Cased}}$ | 110M | 29k | 1 |
| Multilingual | mBERT | 177M | 120k | 104 |
| | XLM | 570M | 200k | 100 |
| | XLM-R$_{\text{Base}}$ | 270M | 250k | 100 |
| | XLM-R$_{\text{Large}}$ | 559M | 250k | 100 |
| Other | LSTM | 1.6M | - | - |
| | CNN | 0.7M | - | - |

Table 7.2: Models statistics with a number of parameters, vocabulary size and a number of supported languages.

`Czech Electra` (Kocián et al., 2022) is a Czech model based on the Electra-small model (Clark et al., 2020) pre-trained on 253GB of text documents. `Czert-B` (Sido et al., 2021)

[8] https://pypi.org/project/corpy/

[9] We use the *TreebankWordTokenizer* class https://www.nltk.org/_modules/nltk/tokenize/treebank.html.

is a Czech cased version of the original $\text{BERT}_{\text{BASE}}$ model (Devlin et al., 2019). Unlike the original BERT model, the authors adjusted the batch size to 2048 and slightly modified the pre-training objective. `RobeCzech` (Straka et al., 2021) is a Czech variant of the RoBERTa model (Y. Liu et al., 2019). `CamemBERT` (Martin et al., 2020) is a French model that follows the architecture and pre-training approach of the RoBERTa model. $\texttt{BERT}_{\texttt{Base-Cased}}$ (Devlin et al., 2019) is the original BERT model. `mBERT` (Devlin et al., 2019) is a multilingual model with the same architecture as the $\text{BERT}_{\text{Base-Cased}}$ but pre-trained on the top of 104 languages, including English, French and Czech. $\texttt{XLM-R}_{\texttt{Base}}$ and $\texttt{XLM-R}_{\texttt{Large}}$ (Conneau et al., 2020) are multilingual versions of the RoBERTa for 100 languages. `XLM` (Conneau & Lample, 2019) is a multilingual model that modifies the training procedure of the original BERT model for multilingual settings mainly by using the Byte-Pair Encoding (BPE) and increasing the shared vocabulary between languages.

#### 7.3.2.1 Transformers Fine-Tuning

Here, we address the task of polarity detection as a text classification problem. We fine-tune our models for binary classification (*positive* and *negative* labels) or three-class classification (*positive, negative* and *neutral*). The architecture we use for text classification follows the approaches in the original papers for our models. Specifically, for models based on the BERT model, we use the hidden vector $\mathbf{h} \in \mathbb{R}^H$ of the classification token `[CLS]` as a representation of the entire input sequence where $H$ is the hidden size of the model. The vector is obtained from the pooling layer, which is a fully-connected layer of size $H$ and a hyperbolic tangent activation function. A dropout of 0.1 is applied and it is then passed through a task-specific linear layer represented by the matrix $\mathbf{W} \in \mathbb{R}^{|C| \times H}$, where C is a set of classes. The output class $c \in C$ is computed as $c = \text{argmax}\ (\mathbf{hW}^T)$.

For the RoBERTa-based models, we use the same[10] approach is used and in addition, an extra dropout of 0.1 is applied before the pooling layer (as in the original RoBERTa implementation). For the XLM model, we use the last hidden state of the first input token (without any pooling layer) and apply the same linear layer ($\mathbf{W} \in \mathbb{R}^{|C| \times H}$) and the same approach to obtain the classification output.

We employ the Adam (Kingma & Ba, 2015) optimizer with default parameters ($\beta_1 = 0.9, \beta_2 = 0.999$) and the cross-entropy loss function. We fine-tune all the parameters of the models. The training data is shuffled randomly before each epoch, and the number of epochs is determined by the performance of the development data, with the best-performing epoch being selected. Due to the Transformer-based models' restriction on the input length, the maximum input sequence is 512. The batch size is set to 32. The input text is tokenized using the HuggingFace library's tokenizer[11] for the corresponding model. We use either a constant learning rate or a linear learning rate decay (without learning rate warm-up) with initial learning rates of 2e-6 and 2e-5. We selected these learning rate values based on our

[10] The first artificial token `<s>` of the input sequence is used instead of the `[CLS]` token.

[11] https://github.com/huggingface/tokenizers

previous experiment observation and the values used in Sun, Qiu, et al. (2019). The details of the used hyper-parameters can be found in Appendix A.

## 7.3.3 Large Language Models

To compare CNN-based, LSTM-based and BERT-like models, we include results with large language models (LLMs), specifically **ChatGPT** (OpenAI, 2022) and **Llama 2** (Touvron, Martin, et al., 2023). LLMs are autoregressive generative transformers pre-trained on massive datasets, often containing billions of parameters (Carlini et al., 2021).

After the initial pre-training phase, which largely aligns with the methodology employed by BERT-like models, the LLMs undergo a distinctive training approach known as reinforcement learning with human feedback (RLHF) (Ouyang et al., 2022), which is a model training approach that enhances the alignment of a fine-tuned language model's behavior with human preferences and its ability to follow instructions. One of the advantages of LLMs is their ability to perform well on new tasks in various scenarios, including zero-shot settings (where no examples are provided), few-shot settings with only a limited number of examples, as demonstrated by Brown et al. (2020) and tasks guided by textual instructions, known as **prompts**. In the case of zero-shot settings and prompting, there is no additional fine-tuning on training data.

### 7.3.3.1 Sentiment Classification with Prompting

Prompting is a relatively new paradigm in NLP (P. Liu et al., 2023) that encourages a pre-trained model to make specific predictions by providing a prompt specifying the task to be done. This technique introduces the need for *prompt engineering*, which is the process of searching for the most suitable prompt that enables a language model to effectively solve the given task. In the era of LLMs, prompt engineering has gained even greater significance, leading to a growing body of research in this area. For example, White et al. (2023) provide a catalog of prompt patterns to solve various problems.

For our experiments, we use the large Llama 2 model with 70B parameters. Llama 2 is based on Llama (Touvron, Lavril, et al., 2023), trained on 40% more data and has twice the context length compared to its predecessor. This model is open source, unlike ChatGPT. Due to the substantial size of the Llama 2 model, deploying it on our own hardware infrastructure proved unfeasible and given the fact that ChatGPT is a closed proprietary model, we were forced to rely on the already deployed versions of these models and access them through the available application programming interface (API). More concretely, we use the fine-tuned version, denoted as *Llama-2-70b-chat*, tailored for dialogue interactions and applications. We utilize the deployed version at HuggingChat[12] and we use the *hugging-chat-api* library to perform the API calls. Additionally, we employ the GPT-3.5 Turbo version of ChatGPT for our experiments, which was developed on top of GPT-3 (Brown et al., 2020). We call the

[12] https://huggingface.co/chat/

official paid OpenAI API[13]. Both of these models have a maximum input length set to 4096 tokens, if the input is longer, we trim the classified example[14].

Influenced, in part, by the work of White et al. (2023), we construct three types of prompts – *basic*, *advanced* and *in-context* which we use for our experiments. Since both of our selected models exhibit multilingual capabilities, we exploit this feature to classify reviews across all three evaluated languages. We employ identical English prompts for both Czech and French datasets while preserving the original language of the review in each respective dataset.

The **basic** prompt simply instructs the model to act as a sentiment classifier for the three-class classification of reviews. The example for the following review *"The movie was fantastic!!!"* is shown in Figure 7.2.

**Basic prompt**

You are a sentiment classifier, classify the following review as "positive","negative" or "neutral".
Answer in one word only.

The review:

The movie was fantastic!!!

Figure 7.2: Example of the basic prompt.

The **advanced** prompt is designed to give the model more details about the task. It encompasses a comprehensive task description, delving into the nuances of the individual classes, and provides explicit directives on approaching the task methodically, breaking it down into distinct steps. The intention was to provide more context about the task and thus achieve better results. The advanced prompt is inspired by the chain-of-thought (CoT) prompting (Wei et al., 2022), which provides a series of intermediate reasoning steps. These steps lead the model to output the final answer step by step and, as shown in Wei et al. (2022) and improve the ability of LLMs to perform complex reasoning and performance. The advanced prompt is shown in Figure 7.3.

Finally, the third **in-context** prompt is based on a technique called *in-context learning* (ICL), which enables pre-trained language models to perform a previously unseen task without any fine-tuning by feeding a small number of training examples as part of the input (Brown et al., 2020; H. Liu et al., 2022; Min et al., 2022). In our case, the in-context prompt is similar to the basic one, but in addition, it contains four randomly sampled examples from the training dataset. The example can be seen in Figure 7.4.

The examples for the in-context prompt are always the same for each currently evaluated sample and the examples are in the same language as the evaluated dataset. The examples are sampled so that each class is represented at least once, i.e., for three-class classification,

[13] https://platform.openai.com/

[14] Such cases are very rare and can occur only for a few examples, due to the lengths of the relatively short text in the datasets.

**Advanced prompt**

You are a Movie and TV Show Review Sentiment Analyzer. You will be given a text of a movie or TV show review, please analyze its content and determine the most appropriate category from the following list. The categories are divided based on the type of sentiment:

Category 1 - positive: Includes reviews that are satisfied with the movie or TV show.
Category 2 - neutral: Includes reviews that are mixed or do not significantly express any sentiment.
Category 3 - negative: Includes reviews that are dissatisfied with the movie or TV show.

The text for analysis will be marked with four slashes, i.e., ////.

Step 1:#### Judge the overall mood of the text and determine which category the text most likely belongs to.
Step 2:#### Focus more closely on the keywords used in the text. Check if the keywords suggest a specific category. For instance, if the text extensively praises the movie or TV show, you should choose "Positive". If the review is mixed or ambiguous, choose "Neutral". If the text criticizes the movie or TV show, choose "Negative".
Step 3:#### Determine the final category based on the highest probability.

Use the following format:
Step 1:#### <rationale for Step 1>
Step 2:#### <rationale for Step 2>
Step 3:#### <rationale for Step 3>

User's answer:#### <the evaluated sentiment itself>

Ensure that you are inserting #### to separate each step.

////The movie was fantastic!!!////

Figure 7.3: Example of the advanced prompt.

*positive, negative* and *neutral* classes are sampled at least once. The prompts for binary classification maintain a similar structure but exclude instructions for the *neutral* class, see Figures A.4, A.5 and A.6 in Appendix A.5.

## 7.4 Methodology & Experimental Setup

This section describes the different types of experiments we carried out and additional details about them. Our study includes several types of experiments. The core cross-lingual experiments are performed either by the Transformer-based models or the CNN or LSTM models in conjunction with the linear transformations. Additionally, we experiment with LLMs (Llama 2 and ChatGPT). We provide the main results for cross-lingual experiments in Section 7.6. To establish the upper-performance threshold for cross-lingual experiments, we conduct a series of monolingual experiments, see Section 7.5 for results along with a comparison to the state-of-the-art models.

**In-context prompt**

You are a sentiment classifier. You will be given a review, please classify the review as “positive”,“negative” or “neutral”. Answer in one word only. As an example, you will obtain examples of the reviews and the desired output.

The examples:
Review:“An opportunity missed.” sentiment:negative
Review:“The most consistently funny of the Austin Powers films.” sentiment:positive
Review:“Kurys never shows why, of all the period’s volatile romantic lives, Sand and Musset are worth particular attention.” sentiment:neutral
Review:“Even as I valiantly struggled to remain interested, or at least conscious” sentiment:negative

Ensure that the output is only one word, i.e., one of the sentiment classes.

The review:

The movie was fantastic!!!

Figure 7.4: Example of the in-context prompt.

### 7.4.1 Evaluation of the Polarity Detection Task

To evaluate the performance of our models on the polarity detection task, we perform multiple experiments for both monolingual and cross-lingual settings. For each experiment, we report the average accuracy score with a 95% confidence interval for the test data obtained by repeating the experiment at least six times. We repeat the experiments to account for the random initialization of neural network weights and to provide more reliable results. We always selected the model with hyper-parameters that achieved the best results on the development data.

We experimented with various combinations of hyper-parameters (e.g., learning rate, learning rate scheduler, optimizer, dropout) for both the linear transformation-based approach and the Transformer-based approach, and we converged to the combinations that turned out to be optimal for our task. The optimal hyper-parameters for monolingual results are reported in Tables A.1, A.2, A.3 and A.4 in Appendix A.

### 7.4.2 Zero-Shot Cross-lingual Sentiment Classification

Our cross-lingual experiments are performed in a zero-shot setting. The idea behind zero-shot cross-lingual experiments for a pair of languages is to use data only from the *source language* to train a cross-lingual model and evaluate the model on data from the *target language*, as is indicated in Figure 6.1. This allows us to test the capacity of the model to transfer knowledge between languages without any labeled data in the target language.

Such ability to transfer knowledge between languages can be very valuable and useful in real-world scenarios because one may have enough labeled data only in one language and creating data in other languages can be prohibitively expensive or even impossible.

The binary classification experiments are done for all pairs of languages. The three-class classification is performed only for the Czech-English pair, and to the best of our knowledge, we are unaware of a suitable French dataset in the movie review domain with three classes that we could use.

We slightly modified the training data for the source language compared to the monolingual experiments. Still, more importantly, the data for testing remains unchanged to allow comparison of the cross-lingual results with the monolingual. For model training, where the source language is French, we use the same split as in the monolingual experiments. In the case of Czech as the source language, we use the CSFD training and testing parts for training and the dev part as development data. The same is valid for the English IMDB and SST datasets, where we use the testing and training parts of the dataset for training models. We use the same hyper-parameter settings for the cross-lingual experiments as for the corresponding models in the monolingual experiments.

We decided to extend the training data for English and Czech because otherwise, the testing parts of the datasets would not be used in the cross-lingual experiments. We assume that in real-world use, all available data would be used for training. For example, using the IMDB dataset without extending the training data would discard half (25k examples) of the dataset. Unfortunately, to the best of our knowledge, there are no other cross-lingual studies available that we could use and directly compare with our results. Consequently, we primarily compare our results with our findings and additionally with the existing monolingual state-of-the-art results.

### 7.4.3 Cross-lingual Sentiment Analysis with Linear Transformations

In Section 7.2.5, we discussed the two tasks to evaluate linear transformations: intrinsic and extrinsic. In this thesis, our main focus is on the extrinsic task of CLSA. These main extrinsic results for CLSA based on linear transformations are outlined in Section 7.6.1. Our recommendations for using linear transformations are discussed in Section 7.8.1. For additional intrinsic experiment with linear transformations, see Section 7.7.1.

Each cross-lingual experiment for a pair of languages based on the linear transformation starts by aligning two semantic spaces of the *source* and *target* languages using one of the five linear transformations described in Section 6.2.1. To recall, the source language is the one whose data is used to train the model, while the target language is the one used to evaluate the model. We explore two options (directions) for transformation: a) transforming the source word embeddings into the target space while leaving the target word embeddings unchanged, and b) transforming the target word embeddings into the source space while leaving the source word embeddings unchanged. In the case of the first option, the source word embeddings are transformed into the target space, while the target word embeddings remain unchanged. Analogously, for transformation from the target space into the source one, the word embeddings of the target space are transformed into the source space, which is not changed. We use the resulting cross-lingual word embeddings space to represent the input text for the CNN or LSTM neural network.

In our cross-lingual experiments based on linear transformations, we investigated two modifications that could potentially impact the performance. Firstly, we examined the effect of using in-domain (movie review) word embeddings in comparison to word embeddings trained on general text, see Section 7.2.2 for their description. Secondly, as discussed in Section 7.2.2.1, we noted that normalization of the word embeddings had been shown to improve performance on intrinsic tasks. Therefore, we explored whether this modification also enhances performance in the downstream task of polarity detection. We considered three options in our experiments. First, we did not perform any normalization on the word embeddings. Second, we normalized the embeddings only before the linear transformation. Finally, we also normalized the embeddings before and after the linear transformation.

We perform the normalization after the linear transformation because the transformation may damage the transformed embeddings. If any normalization is performed, we apply it to both the source and target word embeddings.

Prior to conducting all cross-lingual experiments using linear transformations, we sought to determine the optimal dictionary size for aligning word embeddings and its impact on accuracy. To achieve this, we selected various dictionary sizes, ranging from the first 20 most common words to up to 20,000 of the most common words in our dictionary. We evaluated the impact of the different dictionary sizes on the accuracy of zero-shot cross-lingual polarity detection using a CNN model trained on the IMDB–CSFD-2 dataset pair. The chart with the results and further details about the experiments with dictionary sizes are provided in Section 7.7.1.2.

### 7.4.4 Cross-lingual Sentiment Analysis with Transformer-based Models

The approach for cross-lingual experiments with multilingual Transformer-based models is much more straightforward compared to those with linear transformations. The multilingual model is firstly fine-tuned on data from the source language and then evaluated on data from the target language without the need for any explicit alignment between the two languages. Notably, the model has no prior exposure to labeled data from the target language. In this work, we experiment with four multilingual models: mBERT, XLM-R$_{\text{Base}}$, XLM-R$_{\text{Large}}$ and XLM, mentioned in Section 7.3.2.

Additionally, we wanted to investigate how the performance of the Transformer-based models is affected by limited training data. We started the fine-tuning of the models with only 10% of the original training dataset and we evaluated them on the complete non-reduced test part. We repeated this process, adding an additional 10% of the training data each time until we used the full dataset. Since each model was fine-tuned ten times and testing each cross-lingual dataset pair would be too expensive, we restricted these experiments to the IMDB–CSFD-2 dataset pair only.

### 7.4.5 Experiments with LLMs

As we mentioned, we do not fine-tune the Llama 2 and ChatGPT LLMs, but rather, we leverage the technique called prompting with three types of prompts described in Section 7.3.3.1. In each prompt, the models are instructed to respond in a pre-defined single-word format, signifying the predicted class. However, the output of these models is text in general and it may happen that the model produces output in a different format than it was instructed. In such scenarios, we consider the output as incorrect. We present these original results in Table 7.10.

We observed that the incorrect outputs often contain the correct prediction, but the output includes some additional text. For example, instead of the desired single-word response like *"positive"*, the output was *"overall, the movie has positive sentiment"*. For these predictions, we decided to manually fix the output and report the results in Table 7.11[15]. Despite the manual correction, for some outputs, we were not able to assign the correct predictions because the model returned text as follows *"cannot determine sentiment"*, *"unclear sentiment undetermined"*, *"n/a"*, *"mixed"* etc. In these situations, we consider the output as incorrect. Nonetheless, we have to note that such cases were mostly rare depending on the prompt–dataset–model experiment configuration. The consistency or diversity of generated textual outputs of both models can be controlled by the *temperature* hyper-parameter. Lower values for temperature result in more consistent outputs, while higher values generate more diverse and creative results (OpenAI, 2022). Based on our experiences and recommendations from the model's documentation, we use values similar to the default values, i.e., 1 for ChatGPT and 0.2 for Llama 2.

Inspired by similar works focused on LLMs (Qin et al., 2023; W. Zhang et al., 2023; Zhong et al., 2023), and guided by practical constraints, our experimental evaluations were conducted exclusively on randomly selected subsets of each dataset[16]. Notably, the size of these subsets consisted of 5,000 examples, with one exception being the SST dataset, which, due to its more limited data volume, was evaluated in its entirety. The reason behind this decision is that the ChatGPT API is a paid service and our budget was limited. These limitations prohibited us from executing experiments on the full test sets of the datasets. Furthermore, the openly deployed Llama 2 model, while freely accessible, imposed significant restrictions on the number of requests it could accommodate per minute, allowing only about 6 requests per minute. Given these constraints and to maintain equitable evaluation conditions for both large language models (LLMs), we opted to conduct evaluations on a subset of 5,000 examples. It is worth noting that, in comparison, other relevant studies typically employ much smaller evaluation sets, often consisting of only tens or hundreds of examples.

[15] Sometimes, the output of the model is always in the correct format, in such cases, the results in Tables 7.10 and 7.11 are identical for the same experiment configuration.

[16] The sampling is done only once for each dataset, so in every experiment, we use the same subset of the given dataset.

To potentially mitigate the aforementioned challenges with incorrect predictions, a more sophisticated approach to prompt design may be warranted with different values of the temperature parameter, albeit demanding further experimental endeavors. Regrettably, constraints related to a limited budget and time compelled us to undertake only a restricted set of experiments to yield results that are amenable to comparison with the older approaches. We are aware that further experiments, analyses, and investigations are required to fully understand the effect of individual prompts and models' outputs. Consequently, we defer these endeavors to future research.

## 7.5 Monolingual Results

We report the monolingual results as the accuracy score for all models in Tables 7.3, 7.4, 7.5 and 7.6. Our best results are highlighted in bold, while results from other papers that outperform our models are underlined. The models denoted by *CNN* and *LSTM* were trained with in-domain embeddings, while the models with the suffix *-F*, i.e., *CNN-F* and *LSTM-F*, were trained with the original fastText embeddings. For the LSTM and CNN models, there are two results separated by a slash, where the first number represents the accuracy score for the unnormalized embeddings, and the second number represents the score for the normalized version of the word embeddings.

| **Model** | **CSFD** (Czech) | |
|---|---|---|
| | **2 Classes** | **3 Classes** |
| CNN | $93.9^{\pm0.1}/93.4^{\pm0.1}$ | $83.7^{\pm0.1}/82.9^{\pm0.2}$ |
| CNN-F | $91.5^{\pm0.2}/92.6^{\pm0.1}$ | $80.3^{\pm0.1}/81.7^{\pm0.2}$ |
| LSTM | $94.4^{\pm0.2}/93.9^{\pm0.1}$ | $84.8^{\pm0.2}/84.2^{\pm0.1}$ |
| LSTM-F | $92.1^{\pm0.3}/92.6^{\pm0.3}$ | $81.8^{\pm0.3}/82.8^{\pm0.2}$ |
| Czert-B | $94.4^{\pm0.1}$ | $84.9^{\pm0.1}$ |
| RobeCzech | $95.1^{\pm0.9}$ | $86.0^{\pm0.2}$ |
| Czech Electra | $93.2^{\pm0.4}$ | $81.8^{\pm0.1}$ |
| mBERT | $93.1^{\pm0.3}$ | $82.9^{\pm0.1}$ |
| XLM | $93.9^{\pm0.2}$ | $83.8^{\pm0.1}$ |
| XLM-R$_{\text{Base}}$ | $94.3^{\pm0.3}$ | $85.0^{\pm0.1}$ |
| XLM-R$_{\text{Large}}$ | $\mathbf{96.0}^{\pm0.0}$ | $\mathbf{87.2}^{\pm0.1}$ |
| Habernal et al. (2013)† ME | - | $79.0^{\pm0.3}$ |
| Brychcín and Habernal (2013)† ME | - | $81.5^{\pm0.3}$ |
| Libovický et al. (2018)* LSTM | - | $80.8^{\pm0.1}$ |
| Lehečka et al. (2020)* Distill-BERT | 93.8 | - |

Table 7.3: Monolingual accuracy results for the Czech CSFD dataset. The models from papers marked with † were evaluated with 10-fold cross-validation and the ones marked with * were evaluated on a custom data split. The ME stands for Maximum entropy classifier.

| **Model** | **Allocine** (French) |
|---|---|
| CNN | $95.0^{\pm0.1}/95.1^{\pm0.1}$ |
| CNN-F | $94.3^{\pm0.1}/94.7^{\pm0.2}$ |
| LSTM | $96.4^{\pm0.1}/96.4^{\pm0.1}$ |
| LSTM-F | $95.7^{\pm0.1}/95.9^{\pm0.1}$ |
| CamemBERT | $97.5^{\pm0.0}$ |
| mBERT | $96.2^{\pm0.1}$ |
| XLM | $96.3^{\pm0.0}$ |
| XLM-R$_{\text{Base}}$ | $96.9^{\pm0.0}$ |
| XLM-R$_{\text{Large}}$ | $\mathbf{97.6}^{\pm0.0}$ |
| Théophile (2020) CamemBERT | 97.4 |
| Théophile (2020) CNN | 94.1 |
| Soleymani et al. (2021) Reformer | 95.1 |

Table 7.4: Monolingual accuracy results for the French Allocine dataset (2 classes).

The normalization of the embeddings occasionally improves the monolingual results, particularly for the English SST dataset, but we cannot state that it has some significant effect on the monolingual performance. On the other hand, in-domain embeddings (CNN and LSTM) generally yield better performance compared to models with the original fastText embeddings (CNN-F and LSTM-F), although the difference is typically small. Additionally, the LSTM model tends to outperform the CNN model, likely due to its greater number of parameters (1.6M versus 0.7M for the CNN model). The performance of Transformer-based models varies across languages, with the multilingual XLM-R$_{\text{Large}}$ consistently outperforming other models due to its larger number of parameters. However, monolingual models generally tend to outperform comparable-sized multilingual models across all languages.

While Transformer-based models generally outperform the older CNN and LSTM models, the latter remain competitive for the Czech and French datasets. However, we observe a significantly larger performance gap (around 5%) for the English SST dataset between Transformer-based models and CNN or LSTM models. Another observation is that our results for English are, in general, less competitive compared to other state-of-the-art results.

Regarding our monolingual results for English, we were unable to surpass the performance of XLNet (Yang et al., 2019) and SMART$_{\text{RoBERTa}}$ (Jiang et al., 2020), which are English monolingual models that have undergone improved pre-training phases, unlike the XLM-R$_{\text{Large}}$. The possible clear explanation is that XLM-R$_{\text{Large}}$ is a cross-lingual model that has not been optimized exclusively for English, while the other two models were, thus achieving better results. For the French CamemBERT model fine-tuned by us, we received almost an identical number as Théophile (2020) did. We observed that the French dataset produces the best results in absolute numbers compared to the other datasets. We attribute this to the larger dataset size, as it contains more examples that can help the model achieve better results.

| **Model** | **SST** (English) | |
|---|---|---|
| | **2 Classes** | **3 Classes** |
| CNN | $84.4^{\pm0.6}/84.6^{\pm0.3}$ | $66.4^{\pm1.1}/68.5^{\pm0.6}$ |
| CNN-F | $83.7^{\pm0.2}/85.4^{\pm0.4}$ | $66.1^{\pm1.0}/68.6^{\pm0.8}$ |
| LSTM | $85.3^{\pm0.4}/84.5^{\pm1.2}$ | $69.7^{\pm1.1}/68.2^{\pm1.7}$ |
| LSTM-F | $84.3^{\pm0.6}/85.9^{\pm0.9}$ | $70.4^{\pm0.7}/71.3^{\pm1.2}$ |
| BERT$_{\text{Base-Cased}}$ | $91.0^{\pm0.1}$ | $71.9^{\pm0.1}$ |
| mBERT | $85.2^{\pm0.9}$ | $65.1^{\pm0.4}$ |
| XLM | $89.6^{\pm0.2}$ | $70.5^{\pm0.4}$ |
| XLM-R$_{\text{Base}}$ | $90.9^{\pm0.2}$ | $73.5^{\pm0.2}$ |
| XLM-R$_{\text{Large}}$ | $\mathbf{94.6}^{\pm0.4}$ | $\mathbf{78.1}^{\pm0.5}$ |
| Jiang et al. (2020) SMART$_{\text{RoBERTa}}$ | $\underline{97.5}$ | - |
| Yang et al. (2019) XLNet | 97.1 | - |
| Conneau et al. (2020) XLM-R$_{\text{Large}}$ | 95.0 | - |

Table 7.5: Monolingual accuracy results for the English SST dataset.

| **Model** | **IMDB** (English) |
|---|---|
| CNN | $91.8^{\pm0.1}/91.6^{\pm0.2}$ |
| CNN-F | $89.3^{\pm0.6}/91.1^{\pm0.2}$ |
| LSTM | $92.5^{\pm0.2}/92.6^{\pm0.4}$ |
| LSTM-F | $90.7^{\pm0.7}/91.5^{\pm0.5}$ |
| BERT$_{\text{Base-Cased}}$ | $93.7^{\pm0.0}$ |
| mBERT | $92.4^{\pm0.4}$ |
| XLM | $86.4^{\pm0.2}$ |
| XLM-R$_{\text{Base}}$ | $94.5^{\pm0.2}$ |
| XLM-R$_{\text{Large}}$ | $\mathbf{96.2}^{\pm0.1}$ |
| Yang et al. (2019) XLNet | $\underline{96.8}$ |
| Sun, Qiu, et al. (2019) BERT$_{\text{Large}}$ | 95.8 |

Table 7.6: Monolingual accuracy results for the English IMDB dataset (2 classes). NB stands for NB-weighted-BON + DV-ngram model from the paper.

## 7.6 Cross-lingual Results

In this section, we present and compare the key outcomes of our cross-lingual experiments. In Section 7.6.1, we assess the performance of the five linear transformations described in Section 6.2.1. Section 7.6.2 contains results for experiments with LLMs. In Section 7.6.3, we present the results obtained by multilingual Transformer-based models and compare them with LLMs and models employing linear transformations. In Section 7.6.4, we compare the cross-lingual models with each other. We compare the training and inference speeds of the models in Section 7.6.5, as these factors are crucial for their practical usability in production.

### 7.6.1 Results for Linear Transformations

We present averaged accuracy results for cross-lingual experiments based on linear transformations in Tables 7.7, 7.8 and 7.9 for Czech-English, English-French and Czech-French language pairs, respectively. Each line in the tables provides an averaged accuracy over the five linear transformations for a given experiment configuration.

| | | Evaluated on **Czech** | | | Evaluated on **English** | | |
|---|---|---|---|---|---|---|---|
| **Model** | **Norm.** | **Monoling.** | **EN-s ⇒CS-t** | **CS-t ⇒EN-s** | **Monoling.** | **CS-s ⇒EN-t** | **EN-t ⇒CS-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| **CSFD** (Czech) – **IMDB** (English) | | | | | | | |
| **CNN** | - | 93.9/91.5 | 86.8/77.3 | **88.4**/78.7 | 91.8/89.3 | 83.1/78.4 | 78.3/74.6 |
| | B | 93.4/92.6 | 87.9/85.7 | 87.7/85.7 | 91.6/91.1 | **85.5**/82.9 | 82.5/81.6 |
| | B,A | | 88.1/85.7 | 87.2/85.6 | | 84.9/83.8 | 82.8/83.7 |
| **LSTM** | - | 94.4/92.1 | 85.9/78.5 | 86.8/81.5 | 92.5/90.7 | 79.8/74.3 | 78.6/81.7 |
| | B | 93.9/92.6 | 86.0/81.2 | **87.3**/82.0 | 92.6/91.5 | 83.1/78.7 | 78.9/81.9 |
| | B,A | | 86.5/83.2 | 86.1/82.2 | | 80.5/81.2 | 81.4/**85.2** |
| **CSFD** (Czech) – **SST** (English) | | | | | | | |
| **CNN** | - | 93.9/91.5 | 85.1/72.7 | 85.1/73.5 | 84.4/83.7 | 76.4/73.9 | 73.9/74.0 |
| | B | 93.4/92.6 | 85.0/81.3 | 84.7/81.2 | 84.6/85.4 | 77.5/77.1 | 75.9/77.0 |
| | B,A | | **85.3**/81.9 | 83.5/80.5 | | 78.1/**78.8** | 76.7/77.7 |
| **LSTM** | - | 94.4/92.1 | **83.2**/73.3 | 80.6/74.1 | 85.3/84.3 | 74.6/77.2 | 75.3/76.9 |
| | B | 93.9/92.6 | 80.5/79.0 | 81.6/79.9 | 84.5/85.9 | 76.7/76.2 | 74.7/**78.1** |
| | B,A | | 82.0/78.8 | 82.5/79.6 | | 77.1/77.4 | 76.8/77.9 |
| **CSFD** (Czech) – **SST** (English) 3 Classes | | | | | | | |
| **CNN** | - | 83.7/80.3 | 55.7/47.6 | 55.7/44.9 | 66.4/66.1 | 46.2/48.5 | 50.4/52.5 |
| | B | 82.9/81.7 | 57.2/52.6 | 57.1/53.7 | 68.5/68.6 | 48.9/50.1 | 51.4/53.0 |
| | B,A | | **57.5**/53.7 | 55.7/53.8 | | 49.7/**54.1** | 51.4/53.4 |
| **LSTM** | - | 84.8/81.8 | **53.6**/48.1 | 51.7/40.0 | 69.7/70.4 | 44.7/45.2 | 48.9/50.3 |
| | B | 84.2/82.8 | 52.9/53.5 | 51.6/53.5 | 68.2/71.3 | 51.2/45.6 | 50.5/50.9 |
| | B,A | | 52.6/**53.6** | 51.2/52.2 | | 49.5/47.9 | **52.3**/51.6 |

Table 7.7: Averaged cross-lingual accuracy results for linear transformations obtained on the Czech-English language pair. See the text in Section 7.6.1 for a full description.

In each table, we provide results for models trained with both the *in-domain* embeddings pre-trained by us and the existing *fastText* embeddings, separated by a slash character. We report the results of experiments where the semantic spaces were transformed in both directions[17]. We highlight the background of pairs where in-domain embeddings yield better results than the existing fastText embeddings with a gray color. We also underline the result when the model with any normalization achieves a better performance than the model without normalization[18]. The best results for each language and model pair are in bold.

The *Norm.* column in the tables indicates the type of normalization applied to the word embeddings during the experiments. The *B,A* combination of letters represents that normalization was used before and after the linear transformation. The *B* letter marks normalization before the transformation, while the sign - means no normalization. To facilitate comparison, we also include a *Monoling.* column for the monolingual results of our models[19]. Due to many results and for better clarity and readability, we decided to report only the averaged values here. The complete results for every linear transformation are separately placed in Appendix A.2 in Tables A.5, A.6, A.7, A.8, A.9, A.10, A.11, A.12, A.13, A.14, A.15 and A.16.

| | | Evaluated on **French** | | | Evaluated on **English** | | |
|---|---|---|---|---|---|---|---|
| **Model** | **Norm.** | **Monoling.** | **EN-s ⇒FR-t** | **FR-t ⇒EN-s** | **Monoling.** | **FR-s ⇒EN-t** | **EN-t ⇒FR-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| | | | **Allocine** (French) – **IMDB** (English) | | | | |
| **CNN** | - | 95.0/94.3 | 89.1/76.8 | 86.2/79.3 | 91.8/89.3 | 84.3/78.5 | 80.0/77.0 |
| | B | 95.1/94.7 | 90.1/85.5 | 88.4/86.1 | 91.6/91.1 | 86.5/85.1 | 83.7/83.0 |
| | B,A | | **90.5**/86.4 | 88.8/87.1 | | 83.8/86.4 | **86.7**/85.7 |
| **LSTM** | - | 96.4/95.7 | 89.7/81.2 | 88.8/82.6 | 92.5/90.7 | 79.3/83.1 | 86.7/83.5 |
| | B | 96.4/95.9 | 90.5/79.1 | 89.9/84.0 | 92.6/91.5 | 85.6/85.6 | 88.1/85.6 |
| | B,A | | **91.2**/86.0 | 88.4/87.5 | | 81.2/88.4 | **89.3**/86.0 |
| | | | **Allocine** (French) – **SST** (English) | | | | |
| **CNN** | - | 95.0/94.3 | 87.1/71.3 | 84.0/74.1 | 84.4/83.7 | 77.0/78.1 | 74.6/77.1 |
| | B | 95.1/94.7 | 89.0/84.5 | 86.2/83.9 | 84.6/85.4 | 79.9/79.8 | 79.9/78.8 |
| | B,A | | **89.2**/85.6 | 85.8/84.5 | | 79.5/80.4 | **80.9**/80.2 |
| **LSTM** | - | 96.4/95.7 | 85.4/75.7 | 84.6/76.3 | 85.3/84.3 | 76.7/79.1 | 80.9/78.6 |
| | B | 96.4/95.9 | 85.3/80.0 | 83.7/81.4 | 84.5/85.9 | 80.5/81.2 | 82.0/80.0 |
| | B,A | | **86.2**/81.4 | 83.7/80.9 | | 80.7/81.4 | **82.3**/80.9 |

Table 7.8: Averaged cross-lingual accuracy results for linear transformations obtained on the English-French language pair. See the text in Section 7.6.1 for a full description.

[17] For example, the column labeled as **EN-s** ⇒ **CS-t** in Table 7.7 means that the English space was transformed into the Czech space. English is the source language (-s suffix) and Czech is the target language (-t suffix), in other words, the English dataset is used for training (source) and Czech for the evaluation (target). Analogously, the **CS-t** ⇒ **EN-s** denotes that the Czech space was transformed into the English space, but still, the source language (English) was used for training and the target language (Czech) for evaluation.

[18] We use these formatting techniques to make it easier to compare and observe (in a global view) the effect of normalization and in-domain embeddings on the results.

[19] The slash character also separates the monolingual results pairs and has the same meaning in terms of used in-domain and fastText embeddings.

#### 7.6.1.1 Language Comparison

The tables reveal a stark contrast between the monolingual and cross-lingual results, with the former outperforming the latter by a significant margin with some minor exceptions. For example, the best result (89.3% of accuracy) in Table 7.8 for English on the Allocine-IMDB dataset pair is worse only by 3.3% than the corresponding monolingual result of 92.6%. It may seem to be a large gap, but it is worth noting that the model has never encountered any labeled English examples and thus, we consider it to be a fine result.

The difference between monolingual and cross-lingual settings for experiments with three classes (the CSFD–SST dataset pair) is much more significant, 31% - 27% and 23% - 17% for evaluation of Czech and English, respectively. Such performance drop is caused by the inability of the cross-lingual models to properly learn the *neutral* label (class). Here, we do not show the results for each label separately, but the cross-lingual models are usually able to classify only the *positive* and *negative* classes, whereas the monolingual models are partly able to perform the classification. This can be caused by the fact that the CSFD dataset was annotated in a distant supervised way (partly unsupervised), but the SST dataset was annotated manually. Hence, the annotation of the neutral class might be perceived or shifted and the sentiment label may vary. We discuss it in more detail in Section 7.8.4.

We have noticed that in general, the English-French language pair tends to achieve better accuracy results in absolute numbers compared to cases where the language is paired with the Czech language. For example, according to Table 7.8, the CNN model for French achieves around 90% of accuracy on the Allocine-IMDB pair when trained on English and evaluated on French. However, the results from Table 7.9 indicate that the same model, when trained on Czech and evaluated on French, achieves around 80%-85% accuracy. This can be explained by the fact that French and English come from language families that are much closer and more similar to each other than the Czech language, which belongs to the Slavic language family. This makes the transfer of knowledge and transformation between English and French much easier than in cases involving Czech.

| | | Evaluated on **Czech** | | | Evaluated on **French** | | |
|---|---|---|---|---|---|---|---|
| **Model** | **Norm.** | **Monoling.** | **FR-s ⇒CS-t**<br>in-domain/fastText | **CS-t ⇒FR-s**<br>in-domain/fastText | **Monoling.** | **CS-s ⇒FR-t**<br>in-domain/fastText | **FR-t ⇒CS-s**<br>in-domain/fastText |
| **CNN** | - | 93.9/91.5 | 83.9/75.5 | 79.9/72.7 | 95.0/94.3 | 82.3/75.6 | 77.9/74.8 |
| | B | 93.4/92.6 | 84.8/80.6 | 85.0/77.4 | 95.1/94.7 | 84.0/79.5 | 83.2/74.1 |
| | B,A | | **85.2**/83.3 | 85.1/82.6 | | **85.0**/82.4 | 84.1/78.5 |
| **LSTM** | - | 94.4/92.1 | 85.2/80.9 | 87.1/78.5 | 96.4/95.7 | 77.9/77.4 | 74.9/75.6 |
| | B | 93.9/92.6 | 86.8/84.0 | 87.5/81.1 | 96.4/95.9 | **81.3**/75.1 | 75.7/76.2 |
| | B,A | | 87.4/83.8 | **87.8**/84.8 | | 80.8/75.8 | 80.5/81.0 |

Table 7.9: Averaged cross-lingual accuracy results for linear transformations obtained on the Czech-French language pair. See the text in Section 7.6.1 for a full description.

#### 7.6.1.2 General Observations

The obvious improvement in performance, particularly for the original fastText embeddings, is attributed to the normalization, as indicated by the underlined numbers in the tables. For example, in Table 7.7 the normalized CNN models for the combination of CSFD-IMDB dataset pair evaluated on Czech using the fastText embeddings in the EN-s ⇒CS-t direction achieved 85.7 of accuracy, which is an 8.4% higher accuracy than the result (77.3%) for the unnormalized model. The effect of improvement caused by the normalization is not that visible for the in-domain embeddings. There is no significant difference or discernible pattern in performance between the two versions (before and after the linear transformation) of normalization that we apply. Nonetheless, the normalization consistently produces results that are at least as good as, but usually better than, the unnormalized ones.

Another observation is that the results for models with in-domain embeddings achieved mostly better results than those trained with the original fastText embeddings, even though the in-domain embeddings were pre-trained on a much smaller amount of data (see Section 7.2.2) in comparison to the fastText embeddings.

Based on the results from the mentioned Tables in Appendix A.2, we can not mark any of the five linear transformations as the best one. Although the MSE method performed worse than the other four in some cases, no linear transformation consistently outperformed the others.

There was not much difference in the accuracy performance between the LSTM and CNN models. While the CNN model performed better for the Czech-English language pair, both models yielded similar results overall. From reported confidence intervals in the Tables in Appendix A.2 and partly from our experiences during the experiments, we observed that the CNN is more stable during training compared to the LSTM model.

### 7.6.2 Results for LLMs

We present the accuracy results[20] of unmodified outputs for LLMs in Table 7.10. Table 7.11 contains results with manually fixed predictions, as described above.

From Table 7.10 with unmodified results, we can clearly observe a superiority of ChatGPT in mean performance across all datasets. Typically, this advantage falls within the range of 1% to 2%, except for the IMDB and Allocine datasets, where the difference is more distinctive. The performance of ChatGPT remains consistent irrespective of the employed prompt. Whereas, in the case of Llama 2 model, the *in-context* prompt results differ the most. We hypothesize that this variance can be attributed to random sampling of examples, potentially leading to suboptimal or non-representative selections which may confound the model. In that case, we would expect the same behaviour for ChatGPT unless ChatGPT is more capable. However, we do not have any evidence for such an assumption and more experiments

[20] We present the outcomes for LLMs prior to the Transformer-based findings in Section 7.4.4. This sequencing is deliberate as, within the tables contained in this section, we conduct comprehensive comparisons and summarize the results across all models.

and analyses would be required to validate it. As such, we leave the question open for future research.

The outcomes depicted in Table 7.11 display a higher degree of similarity between the two models in comparison to those illustrated in Table 7.10. The manual fixing of the outputs resulted in a substantial improvement in the accuracy of the Llama 2 model. This observation suggests that the model has difficulties following the instructions for the desired output format rather than with the task itself. From both tables, it is evident that we achieved comparable results as other works employing the ChatGPT model. The comparison of different prompts (from Table 7.11) for ChatGPT shows that, in most cases, all prompts provide similar performance. In the case of the Llama 2 model, the in-context prompt is significantly better for the CSFD-2, IMDB and Allocine datasets.

| **Model** | **Prompt** | **CSFD** | | **SST** | | **IMDB** | **Allocine** |
|---|---|---|---|---|---|---|---|
| | | 2 classes | 3 classes | 2 classes | 3 classes | 2 classes | 2 classes |
| Llama 2 (ours) | basic | 91.9 | **66.5** | 94.8 | 77.8 | 91.7 | 94.9 |
| | advanced | 88.7 | 62.9 | 91.1 | 76.7 | 86.7 | 92.8 |
| | in-context | 89.4 | 50.8 | 94.7 | 77.2 | 72.4 | 87.4 |
| | average | 90.0 | 60.1 | 93.5 | 77.2 | 83.6 | 91.7 |
| ChatGPT (ours) | basic | 92.0 | 62.2 | 95.3 | **79.8** | 94.2 | 94.8 |
| | advanced | 89.4 | 62.5 | 91.7 | 78.3 | 92.5 | 94.2 |
| | in-context | **93.1** | 63.5 | 95.7 | 79.2 | **95.1** | **95.5** |
| | average | 91.5 | 62.7 | 94.2 | 79.1 | 93.9 | 94.8 |
| ChatGPT (W. Zhang et al., 2023) 500 ex. | simple | - | - - | 93.6 | - | 94.2 | - |
| ChatGPT (Qin et al., 2023) 800 ex. | simple | - | - - | 87.6 | - | - | - |
| ChatGPT (Zhong et al., 2023) 50 ex. | chain-of-thought | - | - - | **96.0** | - | - | - |

Table 7.10: Accuracy results for LLMs without additional manual output fixing. Noteworthy performance metrics are highlighted: bold numbers signify the top results for each dataset, while underlined numbers indicate superior averages across the models. Information about the number of evaluated examples is given for the related works.

### 7.6.3 Results for Transformer-based Models

We report the cross-lingual accuracy results for the multilingual Transformer-based models for three languages Czech, French, and English. The results are presented in Tables 7.12, 7.13 and 7.14 for Czech, French and English, respectively[21]. Bold numbers in the tables indicate the best results in a given column (language pair). We also include the best absolute results achieved with the linear transformation approach for both the CNN and LSTM models, as well as monolingual results[22], including selected current state-of-the-art results.

From the tables, it is evident that the XLM-R$_{\text{Large}}$ model dominates in the cross-lingual experiments in all configurations (except for the SST-CSFD dataset pair for three-class clas-

[21] All results in this table were obtained by us unless otherwise indicated by a citation at the beginning of the row. Results with citations were taken from the referenced works.

[22] These results come from the tables in Appendix A.2 and not from the averaged results in Section 7.6.1.

| Model | Prompt | CSFD | | SST | | IMDB | Allocine |
|---|---|---|---|---|---|---|---|
| | | 2 classes | 3 classes | 2 classes | 3 classes | 2 classes | 2 classes |
| Llama 2 (ours) | basic | 92.1 | 66.8 | 95.9 | 77.8 | 95.2 | 94.9 |
| | advanced | 90.8 | 64.1 | 94.3 | 77.1 | 94.3 | 94.3 |
| | in-context | 90.1 | **74.1** | 95.2 | 77.2 | **97.6** | **97.1** |
| | average | 91.0 | 68.4 | 95.1 | 77.4 | 95.7 | 95.4 |
| ChatGPT (ours) | basic | 92.0 | 62.2 | 95.5 | **79.8** | 94.4 | 95.1 |
| | advanced | **94.1** | 63.8 | 95.4 | 78.5 | 93.8 | 95.6 |
| | in-context | 93.2 | 63.5 | 95.7 | 79.2 | 95.1 | 95.5 |
| | average | 93.1 | 63.2 | 95.6 | 79.2 | 94.4 | 95.4 |
| ChatGPT (W. Zhang et al., 2023) 500 ex. | simple | - | - - | 93.6 | - | 94.2 | - |
| ChatGPT (Qin et al., 2023) 800 ex. | simple | - | - - | 87.6 | - | - | - |
| ChatGPT (Zhong et al., 2023) 50 ex. | chain-of-thought | - | - - | **96.0** | - | - | - |

Table 7.11: Accuracy results for LLMs with additional manual output fixing. Noteworthy performance metrics are highlighted: bold numbers signify the top results for each dataset, while underlined numbers indicate superior averages across the models. Information about the number of evaluated examples is given for the related works.

| | Evaluated on **Czech** | | | | | |
|---|---|---|---|---|---|---|
| | CSFD (2 classes) | | | | CSFD (3 classes) | |
| **Model** | **EN → CS**<br>(IMDB – CSFD) | **EN → CS**<br>(SST-2 – CSFD) | **FR → CS**<br>(Allocine – CSFD) | **Monoling.** | **EN → CS**<br>(SST-3 – CSFD-3) | **Monoling.** |
| mBERT | 76.2$^{\pm0.5}$ | 70.0$^{\pm0.9}$ | 79.1$^{\pm0.0}$ | 93.1$^{\pm0.3}$ | 45.8$^{\pm0.8}$ | 82.9$^{\pm0.1}$ |
| XLM | 82.1$^{\pm0.5}$ | 79.6$^{\pm0.6}$ | 84.1$^{\pm0.1}$ | 93.9$^{\pm0.2}$ | 51.1$^{\pm0.6}$ | 83.8$^{\pm0.1}$ |
| XLM-R$_{\text{Base}}$ | 88.1$^{\pm0.4}$ | 85.2$^{\pm0.3}$ | 89.4$^{\pm0.4}$ | 94.3$^{\pm0.3}$ | **59.8**$^{\pm0.3}$ | 85.0$^{\pm0.1}$ |
| XLM-R$_{\text{Large}}$ | **92.1**$^{\pm0.0}$ | **91.0**$^{\pm0.3}$ | **93.4**$^{\pm0.1}$ | **96.0**$^{\pm0.0}$ | 59.4$^{\pm1.4}$ | **87.2**$^{\pm0.1}$ |
| CNN-Best | 89.2$^{\pm0.1}$ | 86.3$^{\pm0.2}$ | 87.0$^{\pm0.2}$ | 93.9$^{\pm0.1}$ | 59.7$^{\pm0.5}$ | 83.7$^{\pm0.1}$ |
| LSTM-Best | 89.1$^{\pm0.3}$ | 86.7$^{\pm0.9}$ | 88.9$^{\pm0.2}$ | 94.4$^{\pm0.2}$ | 57.2$^{\pm0.4}$ | 84.8$^{\pm0.2}$ |
| Czert-B | - | - | - | 94.4$^{\pm0.1}$ | - | 84.9$^{\pm0.1}$ |
| RobeCzech | - | - | - | 95.1$^{\pm0.9}$ | - | 86.0$^{\pm0.2}$ |
| Czech Electra | - | - | - | 93.2$^{\pm0.4}$ | - | 81.8$^{\pm0.1}$ |
| Lehečka et al. (2020)* | - | - | - | 93.8 | - | - |
| Libovický et al. (2018)* | - | - | - | - | - | 80.8$^{\pm0.1}$ |

Table 7.12: Accuracy cross-lingual results for Transformer-based models evaluated on the Czech CSFD dataset compared with the best results of models based on linear transformations. Models marked with * were evaluated on a custom data split.

sification) and outperforms other multilingual Transformer-based models by a large margin. The model size partly causes this superiority since it has many more parameters than the mBERT and XLM-R$_{\text{Base}}$ models. Despite the fact that the XLM model has roughly the same number of parameters as the XLM-R$_{\text{Large}}$ model, its cross-lingual results are worse, even by more than 10% of accuracy. On the other hand, the differences in monolingual results between these two models are not that distinctive, showing a much greater ability of the XLM-R$_{\text{Large}}$ to transfer knowledge between languages compared to the XLM model.

For the binary classifications, we can see that the results for the (zero-shot) cross-lingual

| | Evaluated on **French** – Allocine (2 classes) | | | |
|---|---|---|---|---|
| **Model** | **EN → FR**<br>(IMDB – Allocine) | **EN → FR**<br>(SST-2 – Allocine) | **CS → FR**<br>(CSFD – Allocine) | **Monoling.** |
| mBERT | $84.3^{\pm0.4}$ | $77.4^{\pm0.0}$ | $64.5^{\pm0.5}$ | $96.2^{\pm0.1}$ |
| XLM | $87.4^{\pm0.3}$ | $86.0^{\pm0.2}$ | $80.5^{\pm0.7}$ | $96.3^{\pm0.0}$ |
| XLM-$\mathrm{R}_{\mathrm{Base}}$ | $90.3^{\pm0.4}$ | $89.9^{\pm0.2}$ | $87.7^{\pm0.1}$ | $96.9^{\pm0.0}$ |
| XLM-$\mathrm{R}_{\mathrm{Large}}$ | $\mathbf{94.0}^{\pm0.5}$ | $\mathbf{93.7}^{\pm0.1}$ | $\mathbf{92.8}^{\pm0.1}$ | $\mathbf{97.6}^{\pm0.0}$ |
| CNN-Best | $91.2^{\pm0.1}$ | $89.6^{\pm0.2}$ | $86.1^{\pm0.6}$ | $95.1^{\pm0.1}$ |
| LSTM-Best | $92.1^{\pm0.3}$ | $87.6^{\pm0.5}$ | $85.8^{\pm1.0}$ | $96.4^{\pm0.1}$ |
| CamemBERT | - | - | - | $97.5^{\pm0.0}$ |
| Théophile (2020) | - | - | - | 97.4 |
| Soleymani et al. (2021) | - | - | - | 95.1 |

Table 7.13: Accuracy cross-lingual results for Transformer-based models evaluated on the French Allocine dataset compared with the best results of models based on linear transformations.

classification for multilingual Transformers-based models in all languages are worse than the monolingual results[23]. However, in some cases, the cross-lingual results are very close to the monolingual ones. For example, the XLM-$\mathrm{R}_{\mathrm{Large}}$ model achieved an accuracy of 95.1% when trained on French and evaluated on English, which is only 1.1% worse than the monolingual result (96.2% of accuracy) where the model was trained on English only. This result is outstanding, considering that the model in the cross-lingual setting has never seen any labeled data in English.

In the cases of experiments with three classes (the SST-CSFD dataset pair) when models were evaluated on the Czech language (Table 7.12), both of the XLM-R models achieved similar results, but far worse than the monolingual results. Again, as with linear transformations, the significantly worse performance of the multilingual Transformer-based models on the three-class SST-3 dataset in comparison with the monolingual results is caused by their inability to learn the *neutral* class. We believe the reason for this is the same as for the linear transformation methods, i.e., the discrepancy in the data annotation. We discuss it in more detail in Section 7.8.4.

### 7.6.4 Cross-lingual Models Comparison

Tables 7.12, 7.13 and 7.14 summarize the overall cross-lingual results for Czech, French and English, respectively. In the cross-lingual experiments, the XLM-$\mathrm{R}_{\mathrm{Large}}$ model significantly outperforms cross-lingual models based on linear transformations. The XLM-$\mathrm{R}_{\mathrm{Base}}$ model shows comparable performance to LSTM and CNN models, except for the evaluation on English. The mBERT and XLM models mostly achieve much worse results than the CNN and LSTM models with linear transformations in cross-lingual experiments when evaluated on French and Czech. The mBERT is particularly weak in transferring knowledge between

[23] The model was trained with labeled data from the target language.

| | Evaluated on **English** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | IMDB (2 classes) | | | SST-2 (2 classes) | | | SST-3 (3 classes) | |
| **Model** | **CS → EN**<br>(CSFD – IMDB) | **FR → EN**<br>(Allocine – IMDB) | **Monoling.** | **CS → EN**<br>(CSFD – SST-2) | **FR → EN**<br>(Allocine – SST-2) | **Monoling.** | **CS → EN**<br>(CSFD – SST-3) | **Monoling.** |
| mBERT | $65.7^{\pm1.5}$ | $80.1^{\pm0.3}$ | $92.4^{\pm0.4}$ | $65.2^{\pm0.6}$ | $78.9^{\pm0.3}$ | $85.2^{\pm0.9}$ | $47.9^{\pm3.2}$ | $65.1^{\pm0.4}$ |
| XLM | $85.1^{\pm0.7}$ | $88.7^{\pm0.3}$ | $93.8^{\pm0.2}$ | $78.4^{\pm1.7}$ | $85.3^{\pm0.4}$ | $89.6^{\pm0.2}$ | $47.9^{\pm0.1}$ | $70.5^{\pm0.4}$ |
| XLM-R$_{\text{Base}}$ | $89.5^{\pm0.1}$ | $92.6^{\pm0.1}$ | $94.5^{\pm0.2}$ | $82.9^{\pm0.1}$ | $87.0^{\pm0.1}$ | $90.9^{\pm0.2}$ | $51.2^{\pm3.2}$ | $73.5^{\pm0.2}$ |
| XLM-R$_{\text{Large}}$ | **94.0**$^{\pm0.1}$ | **95.1**$^{\pm0.0}$ | $96.2^{\pm0.1}$ | **87.9**$^{\pm0.1}$ | **90.6**$^{\pm0.4}$ | $94.6^{\pm0.4}$ | $57.3^{\pm0.3}$ | **78.1**$^{\pm0.5}$ |
| CNN-Best | $85.9^{\pm0.1}$ | $88.1^{\pm0.2}$ | $91.8^{\pm0.1}$ | $79.2^{\pm0.1}$ | $82.0^{\pm0.3}$ | $85.4^{\pm0.3}$ | **61.4**$^{\pm1.1}$ | $68.6^{\pm0.8}$ |
| LSTM-Best | $86.2^{\pm1.0}$ | $90.1^{\pm0.5}$ | $92.6^{\pm0.4}$ | $80.3^{\pm0.4}$ | $83.0^{\pm0.5}$ | $85.9^{\pm0.9}$ | $59.4^{\pm0.0}$ | $71.3^{\pm1.2}$ |
| BERT$_{\text{Base-Cased}}$ | - | - | $93.7^{\pm0.9}$ | - | - | $91.0^{\pm0.1}$ | - | $71.9^{\pm0.1}$ |
| Yang et al. (2019) | - | - | **96.8** | - | - | 97.1 | - | - |
| Sun, Qiu, et al. (2019) | - | - | 95.8 | - | - | - | - | - |
| Jiang et al. (2020) | - | - | - | - | - | **97.5** | - | - |

Table 7.14: Accuracy cross-lingual results for Transformer-based models evaluated on the English IMDB and SST datasets compared with the best results of models based on linear transformations.

languages and performs worse than the linear transformation methods in all cross-lingual experiments.

The XLM-R models are on par with the performance of the CNN and LSTM models that use linear transformations in the three-class classification (CSFD and SST-3). Surprisingly, the multilingual Transformer-based models are even surpassed by the linear transformation approaches when evaluated on the English SST-3 dataset (Table 7.14). For example, the XLM-R$_{\text{Large}}$ model is worse by 4.1% than the CNN model (57.3% vs 61.4%).

In the context of cross-lingual sentiment classification, LLMs, including Llama 2 and ChatGPT, consistently exhibit superior performance when compared[24] to other cross-lingual methods, encompassing multilingual Transformer-based models, CNN, and LSTM models with linear transformations. For binary classification, the difference is not very distinctive; there are instances, such as the Czech CSFD dataset, where LLMs deliver comparable results to the XLM-R$_{\text{Large}}$ model. However, the distinction becomes more apparent in three-class classification tasks, where LLMs consistently outshine other cross-lingual methods. A standout example is the SST-3 dataset, where ChatGPT achieves an impressive accuracy of 79.8%, even surpassing the monolingual XLM-R$_{\text{Large}}$ model, which achieves 78.1% accuracy. In the case of French, LLMs perform at par with other monolingual models. LLMs showed an impressive performance in absolute zero-shot settings without any fine-tuning or training data. Despite these remarkable achievements, LLMs are not without their limitations, including concerns related to security and substantial hardware requirements, which we further discuss in Section 7.8.5.

[24] We placed the results under the *Monoling.* column in the summary Tables 7.12, 7.13 and 7.14 since they do not fit under the other columns, as the numbers listed under those columns represent the outcomes achieved by models fine-tuned specifically on corresponding data. In contrast, LLMs were not fine-tuned at all for sentiment classification, making the "Monolingual" column the most suitable place for their placement in the tables.

### 7.6.5 Runtime Comparison of Models

Transformer-based models have demonstrated better performance over older approaches, as shown in our experiments and related work. However, their fine-tuning process is more time-consuming and computationally demanding than older models such as CNN. Therefore, it is important to consider training and inference speed when deploying a model in a real-world application. Here, we compare the relative training and inference speed of the models used in our experiments. We measure the runtimes of the models on a machine equipped with Intel i7-7700k processor, 64GB of RAM, and NVIDIA RTX A4000 graphics card with 16GB of memory. The Ubuntu 18.04.6 LTS operating system is used along with Python 3.7[25].

| Model | Train Time | | | | Inference Time | | | |
|---|---|---|---|---|---|---|---|---|
| | GPU | | CPU | | GPU | | CPU | |
| | relative | [sec]/1k | relative | [sec]/1k | relative | [sec]/1k | relative | [sec]/1k |
| BERT$_{\text{Base-Cased}}$ | 1.00 | 48.11 | 20.53 | 987.94 | 1.00 | 17.73 | 16.91 | 299.70 |
| Czert | 0.95 | 45.83 | 21.99 | 1058.22 | 0.98 | 17.30 | 16.95 | 300.37 |
| RobeCzech | 0.95 | 45.89 | 21.23 | 1021.50 | 0.89 | 15.71 | 17.60 | 311.97 |
| Czech Electra | 0.21 | 9.88 | 5.17 | 248.53 | 0.19 | 3.29 | 3.53 | 62.53 |
| CamemBERT | 0.98 | 46.94 | 23.18 | 1115.07 | 0.92 | 16.32 | 19.02 | 337.19 |
| mBERT | 1.06 | 50.95 | 21.11 | 1015.84 | 1.01 | 17.94 | 17.03 | 301.78 |
| XLM-M | 3.59 | 172.55 | 67.56 | 3250.45 | 3.23 | 57.23 | 53.82 | 954.01 |
| XLM-R$_{\text{Base}}$ | 1.01 | 48.77 | 22.35 | 1075.07 | 1.01 | 17.91 | 17.54 | 310.84 |
| XLM-R$_{\text{Large}}$ | 3.69 | 177.51 | 74.00 | 3560.21 | 3.38 | 59.86 | 60.26 | 1068.17 |
| CNN | 0.01 | 0.23 | 0.02 | 1.62 | 0.01 | 0.22 | 0.09 | 1.15 |
| LSTM | 0.01 | 0.71 | 2.05 | 98.43 | 0.02 | 0.34 | 0.08 | 1.41 |

Table 7.15: Runtimes for each model, relative to the BERT$_{\text{Base-Cased}}$ model trained on GPU and CPU. The numbers in the columns marked as *"relative"* are relative speedups to the underlined numbers which belong to the BERT$_{\text{Base-Cased}}$. The numbers under the *"[sec]/1k"* columns denote the number of seconds required to process 1,000 examples.

During our experiments, we measure the training and inference time in seconds for each model-dataset pair. Then, we calculate the average time it takes to process $N$ examples per $t$ seconds during the training and testing (inference) phases, which we call the *relative runtimes*. We obtain the relative runtimes for each model-dataset pair. For every Transformer-based model, the relative runtimes are identical across all datasets because the input sequence length is always 512 tokens[26]. However, for the CNN and LSTM models, the input is processed up to 512 tokens without padding the examples. The relative runtimes differ because each dataset has a different distribution of example lengths (some are shorter). We average the relative runtimes across all datasets to obtain a single relative runtime for the CNN and LSTM models.

[25] Details about the versions of the Python libraries used in our experiments are available in our GitHub repository[1].

[26] The shorter examples are padded to the 512 tokens and the longer examples are trimmed

Following the approach above, we ensure that the relative speeds of training and inference phases across all models are comparable. We measure the relative runtimes for both CPU and GPU and select the relative runtime of the $\text{BERT}_{\text{Base-Cased}}$ model as the baseline speed. We then calculate the speedup relative to $\text{BERT}_{\text{Base-Cased}}$ by dividing the relative runtimes of all other models by the relative runtime of $\text{BERT}_{\text{Base-Cased}}$. For instance, Table 7.15 shows that the $\text{XLM-R}_{\text{Large}}$ model is 3.69 times slower than $\text{BERT}_{\text{Base-Cased}}$ during training on GPU.

In addition to the relative runtimes, in Table 7.15, we also include the processing time in seconds required by each model to handle 1,000 examples. We do not include the total time required for training/inference because it depends on the number of examples in each dataset and in the case of the training, also on the number of epochs for which the model is trained. Please also note that the reported absolute times heavily depend on the hardware configuration employed in our experiments and these times may vary under different configurations. In contrast, the relative runtimes remain relatively independent[27] between runs on the same type of device (i.e., CPU or GPU), thus being less influenced by the hardware configuration, making them a more robust metric and more suitable for general comparison.

The total time, denoted as $t_{total}$, for training or inference can be computed[28] as follows $t_{total} = \frac{|D|}{1000} \times e \times t$, where $|D|$ represents the number of examples in the given dataset, $e$ is the number of epochs and $t$ stands for the number of seconds required to process 1,000 examples for the given model.[29]

Since for experiments with LLMs (Llama 2 and ChatGPT), we rely on the external infrastructure on which the models are deployed and we access them via API, we cannot provide a fair comparison in terms of speed with the two mentioned approaches. Consequently, we do not present the relative and absolute time values for the Llama 2 and ChatGPT models in Table 7.15. In recognition of this drawback, we aim to provide some context by sharing that, on average, the time to process 1,000 examples with the basic prompt over all datasets was approximately 17 seconds for Llama 2 and 3 seconds for ChatGPT.

The runtime comparison results and conclusions are also valid for cross-lingual experiments because the runtimes for Transformer-based models are the same. In the case of CNN and LSTM models combined with linear transformations, the runtimes will be increased by a small amount of time needed to compute the linear transformations, which is negligible[30].

[27] The times will also depend on the batch size parameter. In our experiments, we use a batch size of 32.

[28] In the case of training, the time required to compute results of development data is not included.

[29] For example, to compute the total training time of RobeCzech model on CPU for the CSFD dataset, the calculation would look like this: $\frac{65{,}793}{1{,}000} \times 13 \times 1{,}021.5 \cong 873{,}698$ seconds, which is almost 243 hours. 65,793 is the number of training examples, 13 is the number of epochs, taken from Table A.3 in the Appendix and 1,021.5 is the number of seconds required to process 1,000 examples during the training of the model.

[30] For the mentioned hardware, the computation time of linear transformations usually took less than a minute depending on the particular linear transformation.

## 7.7 Additional Experiments

This section presents the results of additional and supplementary experiments with Transformer models and linear transformations. In Section 7.7.1, we delve into supplementary experiments concerning linear transformations. This encompasses a thorough examination of word analogy as an intrinsic evaluation task, along with an exploration of the dictionary size essential for optimal linear transformations. Section 7.7.2 evaluates the effect of training data size on the overall performance of the Transformer-based models.

### 7.7.1 Supplementary Experiments with Linear Transformations

To comprehensively evaluate the employed linear transformations, we also incorporate an intrinsic evaluation in the form of a cross-lingual word analogy task with results presented in Section 7.7.1.1. The optimal dictionary size for linear transformations was determined experimentally with the underlying experiments described in Section 7.7.1.2.

#### 7.7.1.1 Word Analogy Evaluation

As stated in previous works (Artetxe et al., 2016; Brychcín et al., 2019), the normalization approach described in Section 7.2.2.1 can enhance results on the intrinsic evaluation tasks. Our experiments on cross-lingual analogies, as shown in Table 7.16, confirm this finding. When any normalization is used, regardless of the linear transformation or embeddings (in-domain vs fastText), the results are constantly better in almost all cases compared to cases without any normalization.

A second noteworthy finding is that the original fastText word embeddings consistently outperform the in-domain word embeddings on the word analogy task. This is caused by the fact that the word analogies evaluate the cross-lingual embeddings on a range of semantic analogies (e.g., relations between currencies or capital cities) and syntactic analogies (e.g., adjective word pairs or verb infinitive and past tense forms pairs), see (Brychcín et al., 2019) for further details and examples. As a result, the data from the movie domain used to pre-train the in-domain word embeddings does not contain as many examples of such pairs, limiting the embeddings' syntactic and semantic information relevant to the general language needed for success in the word analogy task. In contrast, the fastText embeddings were trained on a larger, more general text corpus (Wikipedia), which encompasses most of the aspects necessary for success in the word analogy task.

Further, in Section 7.8, we discuss the impact of embeddings and their normalization on performance in both intrinsic (word analogies) and extrinsic (polarity detection) tasks. Since the dataset used for word analogy evaluation does not include French data, we only evaluate the Czech-English pair. However, as shown by (Brychcín et al., 2019), normalization has the same effect across most of languages.

| Direction | Norm. | Embeddings | Linear Transformation | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | Orto | MSE | CCA | Rank | Or-Ra | Average |
| **CS-s ⇒EN-t** | - | fastText | 58.1 | 51.9 | 60.6 | 63.5 | 63.9 | 59.6 |
| | | in-domain | 33.7 | 29.1 | 34.3 | 35.7 | 34.3 | 33.4 |
| | B | fastText | **72.0** | 64.3 | **72.4** | 74.2 | **76.0** | 71.8 |
| | | in-domain | 37.7 | 31.1 | 37.9 | 39.2 | 39.5 | 37.1 |
| | B,A | fastText | **72.0** | **71.7** | 72.0 | **74.7** | 75.5 | **73.2** |
| | | in-domain | 37.7 | 39.9 | 38.0 | 39.5 | 40.0 | 39.0 |
| **EN-s ⇒CS-t** | - | fastText | 47.1 | 35.4 | 47.7 | 52.9 | 50.5 | 46.7 |
| | | in-domain | 22.1 | 20.3 | 23.1 | 25.6 | 24.4 | 23.1 |
| | B | fastText | 59.0 | 43.0 | **60.7** | **66.0** | **63.0** | 58.3 |
| | | in-domain | 23.2 | 20.1 | 24.1 | 28.0 | 27.9 | 24.7 |
| | B,A | fastText | **59.8** | **65.3** | 59.4 | 64.9 | 59.9 | **61.8** |
| | | in-domain | 23.2 | 28.2 | 24.0 | 26.2 | 25.8 | 25.5 |

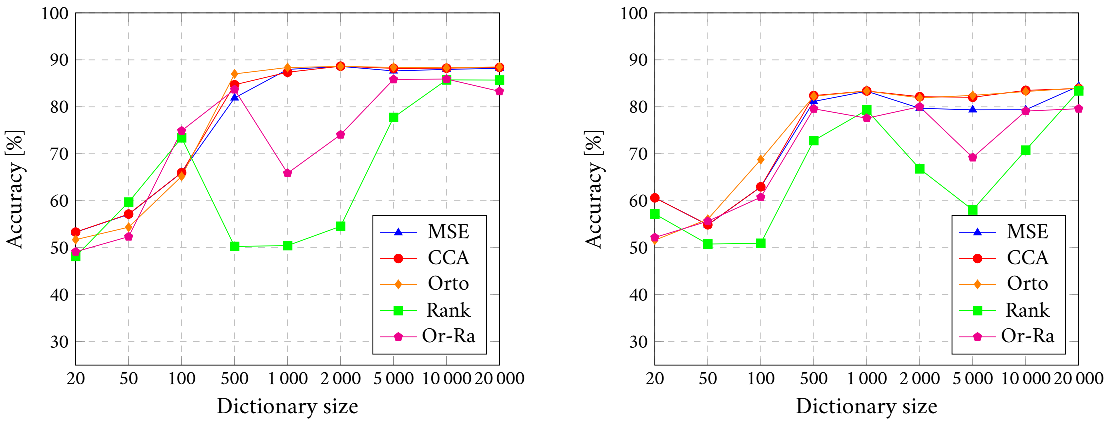
Table 7.16: This table shows the accuracy results of cross-lingual word analogies. The notation **EN-s ⇒CS-t** denotes that the transformation was performed from the English source semantic space into the Czech target semantic space. In this case, the evaluation was performed on the Czech data. Analogously, the **CS-s ⇒EN-t** denotes the transformation from Czech embeddings into English, with evaluation on English data. The column *Norm.* indicates the type of normalization applied to the word embeddings. The **B,A** letters represent that normalization was used before and after the linear transformation, the **B** letter marks normalization before the transformation and the sign - means no normalization.

### 7.7.1.2 The Impact of Dictionary Size

Figure 7.5 indicates that the optimal dictionary size for all transformation methods is 20k. Therefore, based on our experiments and previous experiences, we fixed the dictionary size to 20k for all other cross-lingual experiments with linear transformations. We only performed the complete experiments with the dictionary size for the IMDB–CSFD-2 dataset pair, but our incomplete experiments for other models and dataset pairs showed a similar trend. Interestingly, both the ranking and orthogonal ranking transformations exhibited a similar drop in performance for dictionary sizes between 500 and 5000 most common words. Although we did not further investigate this behaviour, we suspect it is due to negative sampling, which depends more on the dictionary size and requires more examples to function correctly without any adverse effects. We also observed that increasing the dictionary size beyond 20k did not lead to any performance improvements. The final observation is that a dictionary size of 500 is sufficient to achieve decent performance for the MSE, CCA, and orthogonal transformations.

## 7.7.2 The Effect of Training Data Size

Thanks to the pre-training phase, the Transformer-based models contain information that is subsequently leveraged during fine-tuning. Here, we were interested in the helpfulness of



(a) Results for the Czech CSFD-2 dataset. Trained on the English IMDB dataset with the **EN-s** ⇒**CS-t** transformations.

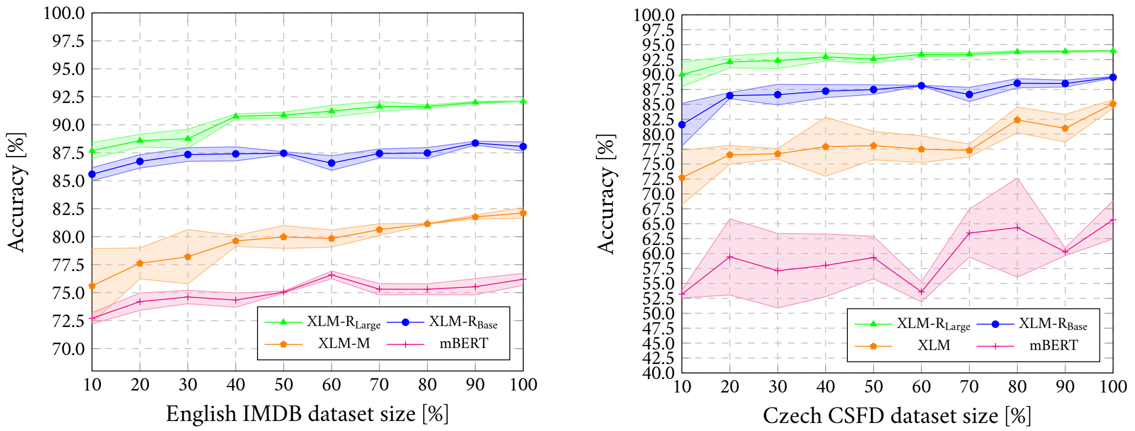
(b) Results for the English IMDB dataset. Trained on the Czech CSFD-2 dataset with the **CS-s** ⇒**EN-t** transformations.

Figure 7.5: Dependence of average accuracy for different dictionary sizes for the CNN models trained on the CSFD-2 and IMBD datasets. The chart is plotted for the in-domain embeddings for the transformation from the English source space into the Czech target space, i.e., Figure 7.5a (**EN-s** ⇒**CS-t**) and for the transformation from Czech source space into the English target space, i.e., Figure 7.5b (**CS-s** ⇒**EN-t**). The embeddings were not normalized.

the pre-trained model's information during experiments in the cross-lingual scenario with a reduced number of training examples.

Figure 7.6 shows accuracy results for the experiments with reduced training dataset sizes for multilingual Transformer-based models. As shown in the previous results, the mBERT model performs worst and when trained on Czech data, it also has a very large confidence interval compared to other models. This is consistent with our experiments on the full dataset size, where mBERT was less stable than the other models during the fine-tuning. The performance drop and lack of stability during fine-tuning are most likely caused by the number of data and vocabulary size used for the Czech language in the mBERT model during the pre-training phase, as reported in Conneau et al. (2020). In contrast, the XLM-R models use a much larger vocabulary (see Table 7.2) and more pre-training data for languages other than English compared to the mBERT model.

As expected, the accuracy generally increases with larger training data sizes. The XLM-R models perform exceptionally well with only 40% and 20% of the English and Czech training datasets, respectively. These results are very close to the accuracy obtained with the full training datasets, showing a significant capacity to transfer knowledge between languages for the polarity detection task with a reasonable amount of training examples. On the other hand, the performance of the other models typically grows steadily with the increasing size of training data.



(a) Results for the Czech CSFD-2 dataset. Trained on the English IMDB dataset.

(b) Results for the English IMDB dataset. Trained on the Czech CSFD-2 dataset.

Figure 7.6: Accuracy results for experiments with multilingual Transformer-based models. The experiment for each dataset part was repeated ten times. The highlighted area represents the confidence interval.

## 7.8 Discussion & Recommendations

In this section, we discuss our observations and compare performance, properties and other aspects of the two types of cross-lingual models for zero-shot polarity detection: multilingual Transformer-based models and approaches using linear transformations. We discuss certain aspects of the results, including the options and limitations of the deployment of CLSA models. Furthermore, we offer practical recommendations for the usage of linear transformations.

### 7.8.1 Recommendations for Linear Transformation

We realize and understand that there is a large number of settings and combinations required to perform cross-lingual sentiment classification with linear transformations. Selecting the optimal settings may be tricky or even challenging. To help researchers with this task, we provide a set of recommendations for the cross-lingual sentiment analysis that we derived from our results and experiences. These hints and suggestions can facilitate the use of cross-lingual transformations in real-world applications or guide future research endeavours.

Firstly, we recommend using the canonical (CCA) transformation due to its stability in results across different datasets. The appropriate alternative method could be the orthogonal (Orto) transformation due to its easy-to-implement analytical solution. The orthogonal ranking (Or-Ra) method may also be an option. However, based on our findings, we primarily recommend the CCA method. The reason is that in some rare cases (see Tables in Appendix A.2), the Orto and Or-Ra methods performed significantly worse than the other methods.

Secondly, normalization proved to be an effective way to improve results, especially for fastText embeddings that are trained on a general (non-domain specific) text. The normalization in combination with in-domain embeddings also usually brings an improvement, albeit not as distinctive as with fastText embeddings. Therefore, we suggest using any of the normalization techniques, as it, in most cases, improves results or at least does not harm them.

Next, in-domain embeddings generally provide better results than general embeddings. Our recommendation is to use or train custom in-domain embeddings if possible. If in-domain embeddings are unavailable, using the general embeddings combined with any normalization technique may be sufficient.

Lastly, as we stated, we did not observe a significant difference between the LSTM and CNN models. In this case, we suggest using the CNN model because our configuration of that model has fewer parameters and proved to be more stable (usually, it has a smaller confidence interval) during training.

### 7.8.2 The Impact of Normalization and In-domain Word Embeddings

Here, we discuss the effect of different configurations of word embeddings on the performance of linear transformations, specifically examining the impact of the normalization of word embeddings and the use of in-domain versus general word embeddings.

Firstly, our experiments, detailed in Section 7.7.1.1, confirm the findings of previous studies (Artetxe et al., 2016; Brychcín et al., 2019), that normalization of the word embeddings enhances the performance of linear transformations on intrinsic tasks, e.g., word analogies. In our study, we were interested in whether this property is also valid for the extrinsic task of cross-lingual polarity detection. Our experiments revealed that normalization usually improves performance, particularly for the general fastText word embeddings. For the in-domain embeddings, the improvement is not that significant. Still, generally, we can state that normalization mostly improves the results in cross-lingual polarity detection when linear transformations are used.

Another intriguing observation is that despite the poor results of in-domain embeddings on the cross-lingual word analogy evaluation task (intrinsic), as seen in Table 7.16, they perform better than the general fastText embeddings on the cross-lingual polarity detection task, as shown in Tables 7.7, 7.8, and 7.9. However, the difference is not as significant as in the case of cross-lingual word analogies.

Furthermore, our in-domain word embeddings, trained on a much smaller amount[31] of text than the general fastText embeddings, can improve the performance of the polarity detection task (extrinsic), despite their poor performance on the cross-lingual word analogy evaluation task (intrinsic). We conclude that good performance on intrinsic evaluation tasks

[31] We used only the training data of the sentiment datasets (see Section 7.2.2), which are many times smaller than the Wikipedia dataset used for the general fastText embeddings.

does not necessarily imply good results on extrinsic tasks and vice versa. Therefore, intrinsic and extrinsic evaluations should be conducted together to obtain a reliable overview of performance when evaluating linear transformations.

### 7.8.3 Training Speed Versus Performance

Various requirements must be considered when evaluating a system in a research environment or deploying it in a real-world application. In academia or a research environment, the goal is often to achieve the best results without any restrictions on computational resources. Meanwhile, in a real-world application or a production environment, there can be restrictions on the available resources and training and inference speed requirements must be satisfied. When some cross-lingual solution is to be used, it is important to set the goal and restrictions that define the used model. If the goal is only to achieve the best performance without any resource limitations, the XLM-$R_{\text{Large}}$ is the obvious choice. The model achieves the best results in the cross-lingual experiments, outperforming other models (both the linear transformations and multilingual Transformer-based models) by a large gap in almost all cases.

Unfortunately, the XLM-$R_{\text{Large}}$ model is also the largest (in a number of parameters) and slowest one in terms of training and inference speed, as indicated in Table 7.15. Thanks to its large number of parameters, it may require significant resources, namely a GPU card with sufficient memory. Such GPU card may not be available in a production environment for inference. One may fine-tune the model on GPU and then run the inference on CPU, but as shown in Table 7.15, the average inference time on CPU for the XLM-$R_{\text{Large}}$ model is 60.26 times slower than the inference time of the BERT model on GPU. Of course, it always depends on a specific use case, but in situations where resources are limited, it may be more practical to consider linear transformation-based approaches for deployment.

The cross-lingual models (CNN and LSTM) that use linear transformations usually achieve comparable or even better results than the smaller multilingual Transformer-based models (XLM-$R_{\text{Base}}$ and mBERT), as shown in Tables 7.12, 7.13, and 7.14. For example, the accuracy of the XLM-$R_{\text{Base}}$ model trained on French (Allocine) and evaluated on Czech (CSFD), see Table 7.12, is 89.4%, while the accuracy of the cross-lingual LSTM model with the same configuration is 88.9%. The main difference lies in their training and inference times, as the LSTM model can be trained in only a fraction (0.01)[32] of the training time of the XLM-$R_{\text{Base}}$ model. So, in the cases where the Transformer-based models do not meet the required limits, the linear transformation approaches can be an option because they are much faster in training and inference than the Transformer-based models. Another secondary advantage is that the smaller models that do not require the GPU will have much smaller electricity consumption.

[32] The runtimes are relative to the BERT model, but the training and inference runtimes for XLM-$R_{\text{Base}}$ are roughly 1 (the runtime is almost the same). Thus, we can consider the runtimes of other models to be also relative to the XLM-$R_{\text{Base}}$ model.

The disadvantage of models that use linear transformations is that they require the relatively difficult configuration of hyper-parameters, which we attempted to mitigate by providing tips and recommendations in Section 7.8.1. On the other hand, the application of multilingual Transformer-based models for cross-lingual usage is more straightforward and it does not differ from the monolingual application.

### 7.8.4 Three-class Classification

As we mentioned, the results for the cross-lingual three-class classification on the SST-CSFD dataset pair are notably worse for both Transformer-based models and linear transformation compared to the monolingual results. The models were not able to successfully learn the *neutral* class in a cross-lingual setting. For example, when the model was trained on Czech data, it could not correctly classify the neutral class in English. Since we observed this behaviour for both types of models (linear transformations and Transformers-based models), we suspected that it was rather caused by the annotations of the datasets than by the inability of the models to learn to classify the *neutral* class. We investigated several examples with the neutral label in both datasets to confirm our suggestion.

Upon further investigation, we discovered that some examples labeled as neutral in the CSFD dataset were actually negative. For example, the review "*Špatně zpracované. Špatné.*" ("*Badly filmed. Bad.*") is clearly negative but labeled as neutral. These mismatched annotations are caused by the distant supervised annotation used for CSFD dataset, as described in Section 7.2.1. Consequently, the models were confused and in a cross-lingual setting, some neutral examples in a different language could not be classified correctly, such as being classified as negative instead of neutral. We can conclude that the CSFD dataset's neutral examples are not entirely suitable for use in conjunction with the SST dataset. A manual correction or using a different dataset would be more appropriate for cross-lingual classification in pairs with the SST dataset. However, given the size of our study and the enormous number of experiments, such activities and experiments are beyond the scope of this thesis part. The "incorrect" neutral labels in the monolingual experiments with the CSFD dataset do not have a significant effect because the models are evaluated on test data labeled in the same way as the train data. The "incorrect" neutral labels also do not impact the binary classification since the positive and negative examples are correctly annotated.

### 7.8.5 LLMs Consideration and Discussion

While the results obtained from ChatGPT and Llama 2 in our experiments are undoubtedly impressive, it is important to acknowledge that their implementation comes with certain drawbacks and considerations. Despite their remarkable performance, LLMs may not always be the universal choice due to specific constraints. For example, employing these models in a secure banking environment, where sensitive data cannot be sent over the Internet to the APIs, might prove challenging or even impossible. The practical use of LLMs in real-world scenarios will depend on the specific conditions and trade-offs. As we have shown, the LLMs

**Review**: *The story... is moldy and obvious.*
**Label**: negative

**Model output:**

Step 1: The text expresses a negative sentiment towards the movie or TV show.
Step 2: The keywords used in the text, such as "moldy" and "obvious," suggest criticism and dissatisfaction.
Step 3: Based on the negative sentiment expressed in the text and the use of negative keywords, the final category is "Negative".

User's answer: Negative

(a) ChatGPT output.

**Review**: *The story... is moldy and obvious.*
**Label**: negative

**Model output:**

Step 1:#### The overall mood of the text is negative.
Step 2:#### The reviewer uses words such as "moldy" and "obvious," indicating disappointment and unoriginality.
Step 3:#### Based on the negative language used, it is highly probable that this review falls under the Negative category.

User's answer:#### Negative

(b) Llama 2 output.

Figure 7.7: Example of LLMs outputs for the advanced prompt for English review annotated as *negative*.

achieve results similar to the older approaches for certain datasets and configurations. However, LLMs are associated with additional expenses, either in terms of financial investment (paid API) or hardware that is significantly more expensive than hardware required by the older neural networks and smaller Transformer-based models. Furthermore, in cases where the performance improvement is marginal, such as a mere 1%, thoughtful consideration must be given to whether the investment in the required hardware is justified. In contrast, unlike LLMs, the older supervised approaches require annotated data and additional fine-tuning for the specific task but operate more efficiently on less expensive hardware.

While LLMs undoubtedly offer significant advantages, they are not without minor drawbacks. Occasionally, these models may produce output that is either undesirable or in an incorrect format. On the other hand, LLMs have the great ability to provide valuable explanations, insight or reasoning behind their decisions. For example, when our advanced prompt based on the *chain-of-thought* (Wei et al., 2022) prompting technique was used for the following review: *"The story ... is moldy and obvious ."* annotated as *negative*, the output of LLMs contains detailed reasoning, see Figure 7.7. We provide more examples in Appendix A.6.

In conclusion, while LLMs like ChatGPT showcase remarkable performance and offer enhanced interpretability through advanced prompts, traditional methods, such as classic Transformer-based models and CNNs or LSTMs with linear transformations, still find practical applications. The choice between these models ultimately depends on the specific requirements, constraints, and priorities of a given task or application.

## 7.9 Conclusion

In this part of the thesis, we have presented a comprehensive study on zero-shot cross-lingual sentiment classification (polarity detection) using multilingual Transformer-based models and neural networks such as CNN and LSTM that use linear transformation to transfer knowledge between languages. Our experiments involved four polarity datasets in the Czech, French and English languages. We performed cross-lingual experiments on all pairs of the three languages. We prepared competitive monolingual baselines that are almost on par with the current SotA models in SA. We compared our zero-shot cross-lingual results with the monolingual ones. We showed that the large XLM-R model (without any labeled training data in the evaluated language) can achieve results that are close to the monolingual ones and outperform all other cross-lingual models. The smaller Transformer-based models and linear transformation-based models provide relatively good performance proving their ability to transfer knowledge between languages.

Our supplementary experiments with LLMs like Llama 2 and ChatGPT and the subsequent comparisons with the aforementioned approaches have shown the remarkable potential of LLMs in zero-shot settings. Regarding performance, LLMs are on par or better with the large XLM-R model. However, these outstanding results come at the price of additional issues. Namely, the required hardware has much higher requirements than the older models since LLMs have many times more parameters. Furthermore, issues pertaining to data privacy and security loom large, particularly in cases involving models like ChatGPT, where data must traverse the internet to access the model's private API. This approach can be problematic, especially when handling highly sensitive data.

In addition to the models' performance, we considered both the performance and speed (training and inference) of the models to assess their practical usability. Considering the large computational resources required by the Transformer-based models, we suggest that in certain situations with restricted resources in real-world applications, using cross-lingual methods based on linear transformations can be an appropriate alternative, as they are much faster in training and inference while providing sufficient performance. Our study highlights the importance of considering both accuracy and efficiency when selecting models for real-world applications and provides valuable insights into the trade-offs between these two factors in the context of cross-lingual sentiment analysis.

Based on our evaluation, experiments and experiences, we proposed a set of recommendations and tips to enhance the usage of the linear transformations due to their challenging hyper-parameters configuration. Overall, our findings can help facilitate the usage of cross-lingual transformations in real-world applications and guide future research in this area. Our work contributes to understanding how to leverage multilingual models for cross-lingual sentiment analysis effectively.

### 7.9.1 Future Work

In the potential future work, we would extend the work by verifying the effectiveness of our cross-lingual results and approaches on different domains. This would involve creating datasets that are specifically designed for cross-lingual sentiment analysis in various domains. Additionally, we would explore the possibility of transferring knowledge across different domains and languages simultaneously. Such a task is very challenging. This would require developing methods to effectively capture and transfer sentiment-related knowledge across multiple domains and languages. Such efforts could potentially result in significant improvements in cross-lingual sentiment analysis, with practical applications in various domains. Another extension of the work would be to fix incorrect neutral classification in the CSFD dataset.

# 8 Sentiment Analysis and Related Tasks

In this chapter, we present the additional work related to SA that complements this thesis's main contribution (cross-lingual sentiment analysis), presented in Chapter 7.

In addition to the task of CLSA, our research contribution to SA lies in monolingual Czech sentiment classification in which we achieved new SotA results at the time of publication of the paper called "Are the Multilingual Models Better? Improving Czech Sentiment with Transformers" (Přibáň & Steinberger, 2021), see Section 8.1. In Section 8.2, we compare multilingual systems actively deployed and used daily to perform sentiment classification in multiple languages. The comparison is part of the "Comparative Analyses of Multilingual Sentiment Analysis Systems for News and Social Media" (Přibáň & Balahur, 2023) publication.

In "Czech Dataset for Cross-lingual Subjectivity Classification" (Přibáň & Steinberger, 2022), we created a new Czech dataset for subjectivity classification and performed cross-lingual experiments, which we describe in Section 8.3.

Regarding the ABSA task, in "Improving Aspect-Based Sentiment with End-to-End Semantic Role Labeling Model" (Přibáň & Pražák, 2023) we proposed a new method to improve the performance of ABSA and achieved new SotA for the Czech language, we describe the results in Section 8.4.

We provide a brief reference to "Prompt-Based Approach for Czech Sentiment Analysis" (Šmíd & Přibáň, 2023), where we applied the prompt-based learning to the ABSA and the sentiment classification tasks. In Section 8.6, we shortly describe results for emotion analysis tasks, sourced from two publications: "UWB at SemEval-2018 Task 1: Emotion Intensity Detection in Tweets" (Přibáň et al., 2018) and "UWB at IEST 2018: Emotion Prediction in Tweets with Bidirectional Long Short-Term Memory Neural Network" (Přibáň & Martínek, 2018). These three publications are described with modest detail, as the author of this thesis is not the first author or the work does not constitute the core part of the thesis.

At last, in Section 8.7 we highlight our research contributions for other NLP tasks. Our main research contributions in this chapter include:

1. *Advancement in Czech sentiment classification: We achieved new state-of-the-art results and conducted a comparison of available monolingual and multilingual Transformer-based models tailored for the Czech language.*

2. *Creation of a new Czech dataset for subjectivity classification: We developed a valuable resource for cross-lingual evaluation by constructing a novel Czech dataset. With the dataset, we carried out cross-lingual experiments between Czech and English.*

3. *Innovative enhancement to Czech ABSA: We proposed a novel approach to improve the results of the Czech ABSA task by incorporating information from semantic role labeling*

## 8.1 Czech Monolingual Sentiment Classification

This section is based on the paper titled "Are the Multilingual Models Better? Improving Czech Sentiment with Transformers" (Přibáň & Steinberger, 2021). The paper aims to enhance Czech sentiment classification with Transformer-based models and their multilingual versions. The experiments conducted involve five multilingual and three monolingual models, all assessed on three distinct Czech sentiment classification datasets. We compare the monolingual and multilingual models' performance, including comparison with the older approach based on recurrent neural networks. Our experiments reveal that the large multilingual models exhibit the capability to surpass the performance of their monolingual counterparts, resulting in the establishment of new state-of-the-art results across all three datasets.

Furthermore, we performed limited cross-lingual experiments to test the multilingual models and their ability to transfer knowledge from English to Czech. Since these experiments were later significantly extended and evaluated in Přibáň et al. (2024) and described in Chapter 7, we do not include them here and focus only on Czech monolingual polarity detection.

### 8.1.1 Data

To the best of our knowledge, there are three Czech publicly available datasets for the polarity detection task: (1) movie review dataset (CSFD), (2) Facebook dataset (FB) and (3) product review dataset (Mallcz), all of them come from (Habernal et al., 2013) and each text sample is annotated with one of three[1] labels, i.e., *positive, neutral* and *negative*, see Table 8.1 for the class distribution of the FB and Mallcz datasets. The statistics and description of the CSFD dataset are present in Table 7.1 in Section 7.2.1.

The `FB` dataset contains 10k random posts from nine different Facebook pages that were manually annotated by two annotators. The `Mallcz` dataset consists of 145k users' reviews of products from Czech e-shop[2], the labels are assigned according to the review star rating on the scale 0-5, where the reviews with 0-3 stars are labeled as *negative*, four stars as *neutral* and five stars as *positive*.

[1] The FB dataset also contains 248 samples with a fourth class called *bipolar*, but we ignore this label.

[2] https://www.mall.cz

| | FB | | | | Mallcz | | | |
|---|---|---|---|---|---|---|---|---|
| | train | dev | test | total | train | dev | test | total |
| Positive | 1,605 | 171 | 811 | 2,587 | 74,100 | 8,253 | 20,624 | 102,977 |
| Negative | 1,227 | 151 | 613 | 1,991 | 7,498 | 848 | 2,041 | 10,387 |
| Neutral | 3,311 | 361 | 1,502 | 5,174 | 23,022 | 2,524 | 6,397 | 31,943 |
| Total | 6,143 | 683 | 2,926 | 9,752 | 104,620 | 11,625 | 29,062 | 145,307 |

Table 8.1: Polarity detection datasets statistics.

## 8.1.2 Models

We performed exhaustive experiments with Transformed-based models and to compare them with the previous works, we also implemented the older models (baseline models) that include the logistic regression classifier and the BiLSTM neural network.

For the description of logistic regression (`lrc`) and `LSTM` baseline models, please see Section 4.1 in the (Přibáň & Steinberger, 2021) paper.

In total, we use eight different Transformer-based models (five of them are multilingual). The evaluated models differ in the number of parameters (see Table 8.2). Consequently, their performance varies significantly, see Section 8.1.3.

| Model | #Params | Vocab | #Langs |
|---|---|---|---|
| Czert-B | 110M | 30k | 1 |
| Czert-A | 12M | 30k | 1 |
| RandomALBERT | 12M | 30k | 1 |
| mBERT | 177M | 120k | 104 |
| SlavicBERT | 177M | 120k | 4 |
| XLM | 570M | 200k | 100 |
| XLM-R$_{\text{Base}}$ | 270M | 250k | 100 |
| XLM-R$_{\text{Large}}$ | 559M | 250k | 100 |

Table 8.2: Models statistics with a number of parameters, vocabulary size and a number of supported languages.

`Czert-A` is the Czech version of the ALBERT model (Lan et al., 2020), also with the same modification as `Czert-B`, i.e., batch size was set to 2048 and the modified NSP prediction task is used instead of the SOP task (Sido et al., 2021). `RandomALBERT` is a randomly initialized ALBERT model without pre-training to show the importance of pre-training of such models and its performance influence on the polarity detection task. `SlavicBERT` (Arkhipov et al., 2019) is initialized from the `mBERT` checkpoint and further pre-trained with a modified vocabulary only for four Slavic languages (Bulgarian, Czech, Polish and Russian). The `Czert-B`, `mBERT`, `XLM`, `XLM-R`$_{\texttt{Base}}$ and `XLM-R`$_{\texttt{Large}}$, models are already described in Section 7.3.2. We fine-tune the Transformer-based models in the same way as is described in Section 7.3.2.1.

### 8.1.3 Experiments & Results

We undertake two distinct sets of experiments, namely *monolingual* and *cross-lingual*. In the *monolingual* experiments, we engage in the fine-tuning and evaluation of Transformer models separately for each dataset. We fine-tune and evaluate the Transformer models for each dataset separately on three-class (*positive, negative* and *neutral*) and two-class (*positive* and *negative*) sentiment classification tasks. As we already mentioned, the *cross-lingual* experiments are part of the previous Chapter 7 and we do not describe them here. Each individual experiment[3] was repeated at least five times and we reported the results using the macro $F_1$ score.

The goal of the monolingual experiments is to reveal the current SotA performance for the Czech polarity datasets, namely CSFD, FB and Mallcz, and compare the available models and their settings. All the previous works typically employed either 10-fold cross-validation or split[4] the datasets on their own (the † and * symbols in Table 8.3, respectively) causing the comparison to be challenging.

| **Model** | 3 Classes | | | 2 Classes | | |
|---|---|---|---|---|---|---|
| | **CSFD** | **FB** | **Mallcz** | **CSFD** | **FB** | **Mallcz** |
| lrc (ours) | 79.6 | 67.9 | 76.7 | 91.4 | 88.1 | 89.0 |
| LSTM (ours) | $79.9^{\pm0.2}$ | $72.9^{\pm0.5}$ | $73.4^{\pm0.1}$ | $91.8^{\pm0.1}$ | $90.1^{\pm0.2}$ | $88.0^{\pm0.2}$ |
| Czert-A | $79.9^{\pm0.6}$ | $73.1^{\pm0.6}$ | $76.8^{\pm0.4}$ | $91.8^{\pm0.8}$ | $91.3^{\pm0.2}$ | $91.2^{\pm0.3}$ |
| Czert-B | $84.9^{\pm0.1}$ | $76.9^{\pm0.4}$ | $79.4^{\pm0.2}$ | $94.4^{\pm0.1}$ | $94.0^{\pm0.3}$ | $92.9^{\pm0.2}$ |
| mBERT | $82.9^{\pm0.1}$ | $71.6^{\pm0.1}$ | $70.8^{\pm5.7}$ | $93.1^{\pm0.3}$ | $88.8^{\pm0.4}$ | $72.8^{\pm3.1}$ |
| SlavicBERT | $82.6^{\pm0.1}$ | $73.9^{\pm0.5}$ | $75.3^{\pm2.5}$ | $93.5^{\pm0.3}$ | $89.8^{\pm0.4}$ | $91.0^{\pm0.2}$ |
| RandomALBERT | $75.8^{\pm0.2}$ | $62.5^{\pm0.5}$ | $64.8^{\pm0.3}$ | $90.0^{\pm0.2}$ | $81.7^{\pm0.6}$ | $85.4^{\pm0.1}$ |
| XLM-R$_{\text{Base}}$ | $85.0^{\pm0.1}$ | $77.8^{\pm0.5}$ | $75.4^{\pm0.1}$ | $94.3^{\pm0.3}$ | $93.3^{\pm0.7}$ | $92.6^{\pm0.1}$ |
| XLM-R$_{\text{Large}}$ | $\mathbf{87.2}^{\pm0.1}$ | $\mathbf{81.7}^{\pm0.6}$ | $\mathbf{79.8}^{\pm0.2}$ | $\mathbf{96.0}^{\pm0.0}$ | $\mathbf{96.1}^{\pm0.0}$ | $\mathbf{94.4}^{\pm0.0}$ |
| XLM | $83.8^{\pm0.1}$ | $71.5^{\pm1.6}$ | $77.6^{\pm0.1}$ | $93.9^{\pm0.2}$ | $89.9^{\pm0.3}$ | $92.0^{\pm0.2}$ |
| (Habernal et al., 2013)† | $79.0^{\pm0.3}$ | $69.0^{\pm0.1}$ | $75.0^{\pm0.2}$ | - | $90.0^{\pm0.1}$ | - |
| (Brychcín & Habernal, 2013)† | $81.5^{\pm0.3}$ | - | - | - | - | - |
| (Libovický et al., 2018)* | $80.8^{\pm0.1}$ | - | - | - | - | - |
| (Lehečka et al., 2020)* | - | - | - | 93.8 | - | - |

Table 8.3: The final monolingual results as macro $F_1$ score for all three Czech polarity datasets on two and three classes. For experiments with neural networks performed by us, we present the results with a 95% confidence interval. The models from papers marked with † were evaluated with 10-fold cross-validation and the ones marked with * were evaluated on custom data split.

We fine-tune all models on training data and we select the model with the best performance on the development data. We report the results in Table 8.3 on the testing data with 95% confidence intervals.

Firstly, we re-implemented the logistic regression classifier (`lrc`) with the best feature combination from (Habernal et al., 2013) and we report the results on our data split. We

[3] Except for the experiments with the `lrc` model.

[4] The authors do not provide any recipe to reproduce the results.

can see that we obtained very similar results to the ones stated in (Habernal et al., 2013). We also tried to improve this baseline with tf-idf weighting, but it did not lead to any significant improvements, so we decided to keep the settings the same as in (Habernal et al., 2013).

For the LSTM model, we tried different combinations of hyper-parameters (learning rate, optimizer, dropout, etc.). We report the used hyper-parameters for the results from Table 8.3 in Appendix A.3 in Table A.17. Our implementation is only about 1% worse for the CSFD dataset than LSTM with the self-attention model from (Libovický et al., 2018), but they used a different data split. For the Mallcz dataset, we could not outperform the lrc baseline with the LSTM model.

We fine-tune all parameters of the seven pre-trained BERT-based models and one randomly initialized ALBERT model. In our experiments, we use constant learning rate and also linear learning rate decay with the following initial learning rates: 2e-6, 2e-5 and 2.5e-5. We got inspired by the ones used in (Sun, Qiu, et al., 2019). Based on the average number of tokens for each dataset and models' tokenizer (see Table 8.4 and Figures A.1, A.2, A.3)[5], we use a max sequence length of 64 and a batch size of 32 for the FB dataset. We restrict the max sequence length for the CSFD and Mallcz datasets to 512 and use a batch size of 32. All other hyper-parameters of the models are set to the pre-trained models' defaults. See Table A.17 in Appendix A.3 for the reported results' hyper-parameters.

| Model | CSFD | | FB | | Mallcz | |
|---|---|---|---|---|---|---|
| | Avg. | Max. | Avg. | Max. | Avg. | Max. |
| Czert-B | 84.5 | 1000 | 20.3 | 64 | 34.3 | 1471 |
| mBERT | 111.6 | 1206 | 25.6 | 66 | 46.6 | 2038 |
| SlavicBERT | 83.6 | 983 | 20.7 | 62 | 34.3 | 1412 |
| XLM | 100.5 | 1058 | 22.6 | 64 | 41.0 | 1812 |
| Czert-A | 81.7 | 993 | 19.7 | 62 | 32.6 | 1435 |
| RandomALBERT | | | | | | |
| XLM-R$_{\text{Base}}$ | 93.9 | 952 | 20.4 | 53 | 37.5 | 1670 |
| XLM-R$_{\text{Lase}}$ | | | | | | |

Table 8.4: The average and maximum number of sub-word tokens for each model's tokenizer and dataset.

If we compare the BERT model from (Lehečka et al., 2020) with the Czert-B, mBERT and SlavicBERT models[6], we can see that on the binary task, they also perform very similarly, i.e., around 93 %, but they used different test data (the entire CSFD dataset[7]). The obvious observation is that the XLM-R$_{\text{Large}}$ model is superior to all others by a significant margin for any dataset. Only for the three-class Mallcz dataset the Czert-B model is competitive (the confidence intervals almost overlap). From the results for the RandomALBERT

[5] The distributions of the other models were similar to those shown in the mentioned Figures.

[6] All of them should have the same or almost the same architecture and a similar number of parameters.

[7] The examples with positive and negative classes.

model, we can see how important is the pre-training phase for Transformers since the model is even worse than the logistic regression classifier[8].

### 8.1.4 Discussion & Remarks

We can see from the results that the recent pre-trained Transformer-based models beat the older approaches (`lrc` and `LSTM`) by a large margin. The monolingual `Czert-B` model is, in general, outperformed only by the XLM-R$_{\texttt{Large}}$ and XLM-R$_{\texttt{Base}}$ models, but these models have five times/three times more parameters, and eight times larger vocabulary. Considering these facts, the `Czert-B` model is still very competitive.

During the fine-tuning, we observed that in most cases, the lower learning rate 2e-6 (see Table A.17 in Appendix A.3) leads to better results. Thus, we recommend using the same one or a similar order. The higher learning rates tend to provide worse results and the model does not converge.

According to the generally higher confidence interval, the fine-tuning of a smaller dataset like FB that has only about 6k training examples is generally less stable and more prone to overfitting than training a model on datasets with tens of thousands of examples. We also noticed that fine-tuning of the `mBERT` and `SlavicBERT` on the Mallcz dataset is very unstable (see the confidence interval in Table 8.3). Unfortunately, we did not find out the reason. A more detailed error analysis could reveal the reason.

### 8.1.5 Conclusion

We evaluated the performance of available Transformer-based models for the Czech language on the task of polarity detection. We compared the performance of the monolingual and multilingual models and we showed that the large XLM-R$_{\texttt{Large}}$ model can outperform the monolingual `Czert-B` model. The older approach based on recurrent neural networks is surpassed by the Transformers by a very large margin. Moreover, we achieved new SotA results for all three Czech polarity detection datasets.

## 8.2 Comparison of Multilingual Systems for SA

This section presents the outcomes of a comparative analysis conducted on three operational real-world systems[9] employed in the Joint Research Center[10].

The evaluation was published in Přibáň and Balahur (2023). We evaluated three in-house SA systems originally designed for three distinct SA tasks, operating within a highly multilingual context. At the time of the evaluation, these systems processed a tremendous volume

[8] The model was trained for a maximum of 15 epochs and it would probably get better with a higher number of epochs, but the other models were trained for the same or lower number of epochs.

[9] At the time of writing the corresponding paper, i.e. in 2018.

[10] Particulary in unit I.3. It is the research centre of the European Commission https://joint-research-centre.ec.europa.eu/jrc-sites-across-europe/jrc-ispra-italy_en

of text on a daily basis. Therefore, it was essential to know their quality and also be able to evaluate these applications correctly. Due to the lack of correct evaluation, we prepared appropriate resources and tools for the evaluation, assessed these applications and summarised the results.

For the evaluation, we collected a large number of available gold standard datasets in different languages and varied text types. The aim of using different domain datasets was to achieve a clear snapshot of the overall performance of the systems and thus obtain a better evaluation quality. We compared the results obtained with the best-performing systems evaluated on their basis and performed an in-depth error analysis to gain deeper insights.

Our findings reveal interesting observations, including instances where certain systems demonstrate superior performance for datasets and tasks beyond their original design specifications. This suggests the potential for substituting one system with another to achieve performance enhancements. Our results are hardly comparable with the original dataset results because the datasets often contain a different number of polarity classes than we used, and for some datasets, there are even no basic results. In cases where comparisons are feasible, our results show that our systems perform very well in view of multilingualism.

It is important to note that the systems under evaluation utilized older approaches and machine learning methods such as SVM or logistic regression. This historical choice can be attributed to the legacy and established practices of these systems, reflecting the state-of-the-art at the time of their inception. The decision to maintain these approaches may be influenced by factors such as system stability or the operational demands of processing large volumes of text on a daily basis. While these systems may deploy older methods, our evaluation demonstrates their continued effectiveness, showcasing their usability for their purpose in multilingual contexts.

### 8.2.1 Tasks Description

The evaluated systems are intended for solving three sentiment-related tasks – *Twitter Sentiment Analysis* (*TSA*) task, *Tonality in News* (*TON*) task and the *Targeted Sentiment Analysis* (*ESA*) task that can also be called Entity-Centered Sentiment Analysis.

In the *Twitter Sentiment Analysis* and *Tonality* tasks, the systems have to assign a polarity which determines the overall sentiment of a given tweet or a news article. *Targeted Sentiment Analysis* (*ESA*) task is a task of a sentiment polarity classification towards an entity mentioned in a given text. For all mentioned tasks, the sentiment polarity can be one of the *positive, negative* or *neutral* labels or a number from $-100$ to 100, where a negative value indicates negative sentiment, a positive value indicates positive sentiment and zero (or values close to zero) means neutral sentiment. In our evaluation experiments, we used the 3-point scale (*positive, negative, neutral*).

### 8.2.2 Systems Overview

`TwitOMedia` system (Balahur et al., 2014) for the *TSA* task uses a hybrid approach, which employs supervised learning with a Support Vector Machines Sequential Minimal Optimization (Platt, 1999), on unigram and bigram features.

`EMMTonality` system for the *TON* task counts occurrences of language-specific sentiment terms from our in-house language-specific dictionaries. Each sentiment term has a sentiment value assigned. The system sums up values for all words (which are present in the mentioned dictionary) in a given text. The resulting number is normalized and scaled to a range from $-100$ to 100 where the negative value indicates negative tonality, the positive value indicates positive tonality and the neutral tonality is expressed with zero.

`EMMTonality` system also contains a module for the *ESA* task, which computes sentiment towards an entity in a given text. This approach is the same as for the tonality in news articles, with the difference that only a certain number of words surrounding the entity are used to compute the sentiment value towards the entity.

`EMMSenti` system is intended to solve only the *ESA* task. This system uses a similar approach to the `EMMTonality` system, see (Steinberger et al., 2011) for the detailed description.

The evaluated systems require different types of datasets or at least different domains to carry out a proper evaluation. We collected mostly publicly available datasets, but we also used our in-house non-public datasets. The polarity labels for all collected Twitter and news datasets are *positive, neutral or negative*. If the original dataset contained other polarity labels than the three mentioned, we either discarded them or mapped them to *positive, neutral* or *negative* polarity labels.

### 8.2.3 Twitter Datasets

In this section, we introduce the sentiment datasets specific to the Twitter domain. Our collection comprises a total of 2.8 million labeled tweets obtained from multiple datasets, with detailed statistics provided in Table 8.5. We refer to the corresponding paper (Přibáň & Balahur, 2023) for detailed information about the datasets, including their descriptions.

### 8.2.4 Targeted Entity Sentiment Datasets

For the *ESA* task, we were able to collect three labeled datasets. Datasets from L. Dong et al. (2014) and Mitchell et al. (2013) are created from tweets, and our *InHouse Entity* dataset (Steinberger et al., 2011) contains sentences from news articles, see Table 8.6 for the statistics and Přibáň and Balahur (2023) for their description.

| Dataset | Total | Positive | Negative | Neutral |
|---|---|---|---|---|
| Sentiment140 Test | 498 | 182 | 177 | 139 |
| Sentiment140 Train | 1,600,000 | 800,000 | 800,000 | - |
| Health Care Reform | 2,394 | 543 | 1,381 | 470 |
| Obama-McCain Debate | 1,904 | 709 | 1,195 | - |
| Sanders | 3,424 | 519 | 572 | 2,333 |
| T4SA | 1,179,957 | 371,341 | 179,050 | 629,566 |
| SemEval 2017 Train | 52,806 | 20,555 | 8,430 | 23,821 |
| SemEval 2017 Test | 12,284 | 2,375 | 3,972 | 5,937 |
| InHouse Tweets Test | 3,813 | 1,572 | 601 | 1,640 |
| InHouse Tweets Train | 4,569 | 2,446 | 955 | 1,168 |
| Total | 2,861,649 | 1,200,242 | 996,333 | 665,074 |

Table 8.5: Twitter datasets statistics.

| Dataset | Total | Positive | Negative | Neutral |
|---|---|---|---|---|
| Dong | 6,940 | 1,734 | 1,733 | 3,473 |
| Mitchel | 3,288 | 707 | 275 | 2,306 |
| InHouse Entity | 1,281 | 169 | 189 | 923 |
| Total | 11,509 | 2,610 | 2,197 | 6,702 |

Table 8.6: Targeted Entity Sentiment Analysis datasets statistics.

### 8.2.5 News Tonality Datasets

For the *TON*[11] task, we used our two non-public multilingual datasets. Firstly, our **InHouse News** dataset consists of 1,830 manually labeled texts from news articles about the Macedonian Referendum in 23 languages, but the majority is formed by Macedonian, Bulgarian, English, Italian and Russian, see Table 8.7. For the evaluation of our systems, we used only Bulgarian, English, Italian and Russian because other languages are either not supported by the evaluated systems or the number of examples is less than 60 samples.

| InHouse News | Total | Positive | Negative | Neutral |
|---|---|---|---|---|
| Macedonian | 974 | 516 | 234 | 224 |
| Bulgarian | 215 | 118 | 26 | 71 |
| English | 339 | 198 | 35 | 106 |
| Italian | 62 | 41 | 3 | 18 |
| Russian | 65 | 17 | 34 | 14 |
| Other Languages | 175 | 60 | 44 | 71 |
| Total | 1,830 | 950 | 376 | 504 |

Table 8.7: InHouse News dataset statistics.

| EP News | Total | Positive | Negative | Neutral |
|---|---|---|---|---|
| English | 2,193 | 263 | 172 | 1,758 |
| German | 5,122 | 389 | 179 | 4,554 |
| French | 2,964 | 574 | 308 | 2,082 |
| Italian | 1,544 | 291 | 152 | 1,101 |
| Spanish | 3,594 | 324 | 135 | 3,135 |
| Total | 15,417 | 1,841 | 946 | 12,630 |

Table 8.8: EP Tonality News dataset statistics.

[11] For this task, we also used tweets described in subsection 8.2.3

**EP News** dataset contains more than 50K manually labeled news articles with tonality about the European Parliament and European Union in 25 European languages. We selected five main European languages (English, German, French, Italian and Spanish) for the evaluation, see Table 8.8 for details.

### 8.2.6 Evaluation & Results

In this section, we present a summary of all the evaluation results for all three systems. Each system undergoes evaluation with a carefully chosen collection of datasets, where examples from each selected dataset are classified individually. Subsequently, we merge all selected datasets and conduct a unified classification.

We carry out experiments on the `EMMTonality` system with the *InHouse News* dataset on Bulgarian, English, Italian and Russian. Experiments with the *EP News* dataset are performed on the `TwitOMedia` and `EMMTonality` system with English, German, French, Italian and Spanish[12].

Each sample is classified as `positive, negative` or `neutral` and for all named systems, we did not apply any additional preprocessing steps. We used `Accuracy` and `Macro` $F_1$ as evaluation metrics.

#### 8.2.6.1 Baseline Results

We created baseline models for the *TSA* and *TON* tasks for basic comparison. These baseline models are based on unigram-bigram features. Results are shown in tables 8.9, 8.10, and 8.11. For the baseline models, we apply minimal preprocessing steps like lowercasing and word normalization, which include the conversion of URLs, emails, money, phone numbers, usernames, dates and number expressions to one common token These steps lead to a reduction of feature space.

| **Model** | **$F_1$** | **Acc.** |
|---|---|---|
| Log. regression | **55.3** | **58.4** |
| SVM | 53.1 | 56.4 |
| Naive Bayes | 42.3 | 49.9 |

Table 8.9: Baseline results for the InHouse Tweets Test dataset with (trained on InHouse Tweets Train dataset).

To train the baseline models, we use an implementation of Support Vector Machines (SVM) – concretely Support Vector Classification (SVC) with linear kernel, Logistic Regression with lbfgs solver and Naive Bayes algorithms from the scikit-learn library (Pedregosa et al., 2011), default values are used for other parameters of the mentioned classifiers. Our

[12] While experiments with the `EMMTonality` system covered all available languages, results are reported solely for English, German, French, Italian, and Spanish.

| **Model** | All langs | | | | English | | | |
|---|---|---|---|---|---|---|---|---|
| | **InHouse News** | | **EP News** | | **InHouse News** | | **EP News** | |
| | **$F_1$** | **Acc.** | **$F_1$** | **Acc.** | **$F_1$** | **Acc.** | **$F_1$** | **Acc.** |
| Log. regression | 70.4 | 73.8 | 57.8 | **87.0** | **68.5** | **76.9** | 53.4 | 82.6 |
| SVM | **71.7** | **74.7** | **59.1** | 86.6 | 67.4 | 76.0 | **54.6** | **82.7** |
| Naive Bayes | 64.6 | 70.2 | 55.2 | 85.2 | 54.7 | 71.3 | 44.6 | 81.9 |

Table 8.10: Baseline results for the `InHouse News` and the `EP News` datasets for examples in all languages (all langs) and only for English examples. We used 10-fold cross-validation (results in table are averages of individual folds). **Bold** values denote the best results.

*InHouse News* dataset does not contain a large number of examples. Therefore, we perform experiments with 10-fold cross-validation; the same approach is applied for the *EP News* dataset.

For the News datasets (*InHouse News* and *EP News*) we train baseline models with various combinations of data. Table 8.10 shows results for models which are trained on a concatenation of examples in different languages, along with results achieved solely on the English examples. In experiments involving all languages, we include all untranslated examples (texts in their original languages) and we train the model regardless of the language. The model gains the capability to classify texts in all these languages. This approach should lead to performance improvement, as is shown in Balahur et al. (2014).

If we compare baseline results from Table 8.10 with results from Table 8.12 (specifically, the last three lines of the table), a noticeable trend emerges – the baselines consistently outperform our current system, as indicated by the $F_1$ scores in the tables. The `TwitOMedia` system was initially trained on tweet messages, so it is evident that its performance on news articles will be lower.

We collected a large manually labeled dataset of tweets and wanted to study the possibility of using this dataset to train a model. The envisioned outcome was the development of a model capable of classifying news articles, a domain distinct from the training data. After comparing results from Table 8.11 with results from Table 8.12 (specifically, the last three lines of the table), we can see that our simple baseline is not outperformed on the *InHouse News* dataset by the other two systems. These findings underscore the viability of utilizing data from disparate domains for training, demonstrating the potential for performance improvement.

Additionally, we observed that incorporating the title (concatenating the title and the text) of a news article contributes to a consistent performance boost across all datasets and various combinations of training data. These results show that the title is an essential part of the news and contains significant sentiment and semantic information despite its short length.

| Model | InHouse News | | EP News | |
|---|---|---|---|---|
| | $F_1$ | Acc. | $F_1$ | Acc. |
| Log. regression | **40.8** | **46.2** | 31.0 | 49.5 |
| SVM | 38.9 | 45.6 | 28.7 | 39.7 |
| Naive Bayes | 23.9 | 29.3 | **31.4** | **62.0** |

Table 8.11: Baseline results for models trained on *SemEval 2017 Train* and *Test* datasets with. Evaluation was performed on original English examples from our *InHouse News* and *EP News* datasets. Bold values denote best results for each dataset.

### 8.2.7 Twitter Sentiment Analysis

We used a domain-rich collection of tweets datasets to evaluate a system for the *TSA* task. We collected datasets with almost 3M labeled tweets. Detailed statistics of used datasets can be seen in Table 8.5. Table 8.12 shows obtained results for *Accuracy* and *Macro* $F_1$ measures.

From Table 8.12 is evident that the `TwitOMedia` system (Balahur et al., 2014) performs best for the *InHouse Tweets Test* dataset (bold values in the table). This dataset is based on data from Nakov et al. (2013) and was used to develop (train and test) this system.

The superior performance of the `TwitOMedia` system on the *InHouse Tweets Test* dataset compared to the *InHouse Tweets Train* dataset (HTTr) can be attributed to the system being trained on translations of the HTTr dataset. The original training dataset (HTTr) was translated into several languages, and then the translations were merged into one training dataset used to train the model. This approach improves performance, as shown in Balahur et al. (2014).

For the other datasets, the performance is lower, especially for the domain-specific ones and datasets which does not contain instances with *neutral* classes, for example, *Health Care Reform* dataset or *Sentiment 140 Train* dataset.

### 8.2.8 Tonality in News

`EMMTonality` system for the *TON* task was evaluated on the same set of datasets as the one for the `TwitOMedia` system. Obtained results are shown in Table 8.12.

If we compare results of the `TwitOMedia` system and results of the `EMMTonality` system, we can see that the `EMMTonality` system achieves better results for these datasets: *Sentiment140 Test, Health Care Reform, Obama-McCain Debate, Sanders, SemEval 2017 Train,* and *SemEval 2017 Test*. The overall results are better for the `TwitOMedia` system. Both evaluated systems have comparable results for the *InHouse News* and *EP News* datasets.

The `EMMTonality` system slightly outperformed the `TwitOMedia` system in Macro $F_1$ score, see Table 8.13. Table 8.13 contains results for the *EP News* dataset for five languages. The `Config` column denotes whether only the text of an example is used or if a title of the example is concatenated with the text and is used as well.

| **Dataset** | **TwitOMedia** | | **EMMTonality** | |
|---|---|---|---|---|
| | **F$_1$** | **Acc.** | **F$_1$** | **Acc.** |
| Sentiment140 Test | 56.6 | 53.0 | 66.6 | 63.9 |
| Health Care Reform | 41.0 | 32.6 | 45.6 | 40.3 |
| Obama-McCain Debate (OMD) | 27.0 | 29.0 | 33.1 | 35.7 |
| Sanders | 46.8 | 59.1 | 52.6 | 61.8 |
| Sentiment140 Train (S140T) | 31.2 | 35.8 | 25.0 | 37.5 |
| SemEval 2017 Train | 50.1 | 52.9 | 53.8 | 56.1 |
| SemEval 2017 Test | 46.0 | 50.0 | 55.2 | 56.4 |
| T4SA | 60.3 | 66.9 | 41.0 | 39.2 |
| InHouse Tweets Test (HTT) | **71.0** | **70.8** | 58.3 | 61.0 |
| InHouse Tweets Train (HTTr) | 62.9 | 59.9 | 58.0 | 57.4 |
| All Tweets w/o S140T, OMD, T4SA | **59.7** | **66.0** | 54.5 | 56.3 |
| All Tweets w/o S140T, T4SA | 50.7 | 52.8 | 54.2 | 55.8 |
| InHouse News en | 39.7 | 42.5 | 39.8 | 42.5 |
| EP News en, text | 36.8 | **69.8** | 42.2 | 67.8 |
| EP News en, title + text | 37.2 | 69.0 | **42.5** | 67.5 |

Table 8.12: Macro $F_1$ score and Accuracy results of the evaluated `TwitOMedia` and `EMMTonality` systems. **Bold** values denote the best results in specific dataset category (Individual Twitter datasets, joined Twitter datasets and News datasets), and underlined values denote best results for specific dataset category and for each system separetely.

### 8.2.9 Targeted Sentiment Analysis

We evaluated the `EMMSenti` and `EMMTonality` systems for the *ESA* task on the *Dong, Mitchel* and *InHouse Entity* datasets with results shown in Table 8.14.

We obtained the best results for the *InHouse Entity* dataset in terms of *Accuracy* measure and also for the *Macro* $F_1$ score. The best results across all datasets and systems are obtained for the *neutral* class (not reported in the table) and for other classes, our systems work more poorly. The classification algorithm (for both systems) is based on counting subjective terms (words) around entity mentions (no machine learning algorithm or approach is involved). It is obvious that the quality of dictionaries used, as well as their adaptation to the domain, is crucial. If no subjective term from the text is found in the dictionary, the example is assigned the neutral label.

The best performance of our systems for the neutral class can be explained by the fact that most of the neutral instances do not contain any subjective term.

### 8.2.10 Error Analysis

To understand the causes of erroneous classification, we analyze the misclassified examples from Twitter and the News datasets for the `EMMTonality` and `TwitOMedia` systems. We

| Lang. | Config | TwitOMedia | | EMMTonality | |
|---|---|---|---|---|---|
| | | $F_1$ | Acc. | $F_1$ | Acc. |
| EN | Text | 36.8 | 69.8 | 42.2 | 67.8 |
| | Text+Title | 37.2 | 69.0 | 42.5 | 67.5 |
| DE | Text | 33.3 | 71.1 | 34.8 | 84.6 |
| | Text+Title | 34.4 | 68.7 | 36.0 | 73.0 |
| FR | Text | 35.4 | 61.4 | 38.9 | 54.9 |
| | Text+Title | 35.6 | 60.2 | 38.3 | 47.2 |
| IT | Text | 31.4 | 69.2 | 39.7 | 34.7 |
| | Text+Title | 35.1 | 69.0 | 40.5 | 33.0 |
| ES | Text | 33.7 | 82.8 | 39.2 | 38.6 |
| | Text+Title | 33.2 | 82.3 | 39.2 | 33.3 |

Table 8.13: Macro $F_1$ score and Accuracy results for the *EP News* dataset for English, German, French, Italian and Spanish examples.

| Dataset | EMMSenti | | EMMTonality | |
|---|---|---|---|---|
| | $F_1$ | Acc. | $F_1$ | Acc. |
| Dong | 49.1 | 51.2 | 49.6 | 50.1 |
| Mitchel | 48.3 | 66.0 | 49.0 | 64.0 |
| InHouse Entity | **51.7** | **66.3** | 50.7 | 65.9 |
| All | 50.5 | 57.1 | 51.2 | 55.7 |

Table 8.14: Macro $F_1$ score and Accuracy results for the `EMMSenti` and `EMMTonality` systems evaluation. Bold values denote best results for each dataset.

categorize the errors into four groups[13]. We randomly selected 40 incorrectly classified examples for each class and each system across all datasets used for evaluating these systems, resulting in 240 manually evaluated examples. We found the four major groups of errors:

**1. Implicit sentiment/external knowledge:** Sentiment is often expressed implicitly, or external knowledge is needed for a correct classification. The evaluated text does not contain any explicit attributes (words, phrases, emoji/emoticons) that would clearly indicate the sentiment. Because our systems are based on surface-level features (unigrams/bigrams or counting occurrences of sentiment words), they will fail in these examples. For example, text like *"We went to Stanford University today. Got a tour. Made me want to go back to college."* indicates positive sentiment, but for this decision, we have to know that *Stanford University* is a prestigious university (which is positive) and according to the sentence *"Made me want to go back to college."* author probably has a positive relation to universities or his previous studies. This group of errors is the most common in our set of error analysis examples. We observed it in 94 cases and only for examples labeled as positive or negative.

**2. Slang expression:** Misclassified examples in this group contain domain-specific words, slang expressions, emojis, unconventional linguistic means, misspelt or uppercased words like *"4life", "YEAH BOII", "yessss", "grrrl", "yummmmmy"*. We observe this type of error in 29 examples and most of them were caused by the `EMMTonality` system, which is reasonable because this system is intended for news with correct grammar and formal language.

**3. Negation:** Negation of terms is an essential aspect of sentiment classification (Reitan et al., 2015). Negations can easily change or reverse the sentimental orientation. This error appeared in 35 cases in our set of error analysis examples.

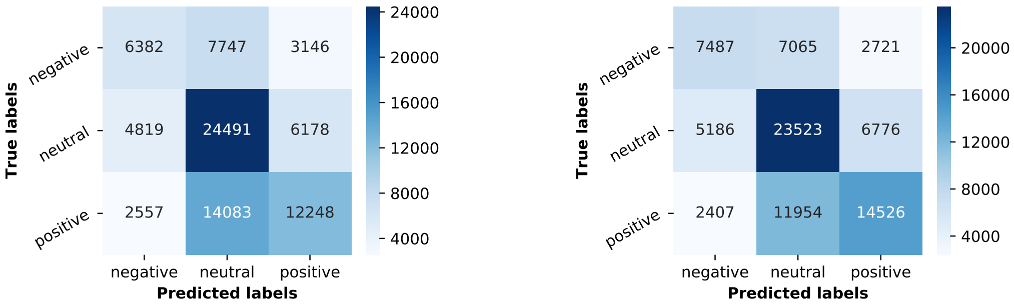
**4. Opposite sentiment words:** The last type of error is caused by sentiment words

[13] Each incorrectly classified example may be contained in more than one error group. Some examples were also (in our view) annotated incorrectly. For some cases, we could not discover the reason for misclassification.

which express the opposite or different sentiment than the entire text. This type of error was typical for examples annotated with a neutral label. For example, tweet *"#Yezidi #Peshmerga forces playing volleyball and crushing #ISIS in the frontline."* is annotated as neutral, but contains words like *"crushing, #ISIS"* or *"frontline"* which can indicate negative sentiment. We observed this type of error in 20 examples.

The first group of errors (*Implicit sentiment/external knowledge*) was the most common among the evaluated examples and is also the hardest one to solve because the system would have to have access to world knowledge or be able to detect implicit sentiment in order to be able of correct classification. This error was observed only for examples annotated with positive or negative labels; there, the explicit sentiment markers are missing. The majority of these examples were misclassified as a neutral class.

Lastly, we have to note that we could not decide the reason for misclassification in 35 cases. According to us, in seven cases was the annotated label incorrect. Figure 8.1 shows confusion matrices for the `EMMTonality` and `TwitOMedia` systems. We can see that a noticeable amount of misclassified examples was predicted as a neutral class.



(a) TwitOMedia system

(b) EMMTonality system

Figure 8.1: Confusion matrices for the `TwitOMedia` and `EMMTonality` systems on all tweets without *S140T* and *T4SA* datasets.

### 8.2.11 Conclusion

We conducted a comprehensive performance assessment of three sentiment classification systems commonly employed in practical, real-world applications. We collected and described a rich collection of publicly available datasets. Our evaluation unveiled the limitations of the systems through an error analysis. Furthermore, in supplementary experiments, we underscored the significance of document titles in system performance, underscoring their role in providing critical and valuable information for effective classification.

During our experimentation, contemporary Transformer-based models, such as BERT, were either in their nascent stages or had yet to be introduced, rendering their integration into the real-world application under evaluation impractical. The main goal and important aspect of these experiments was to obtain a clear performance snapshot of the employed multilingual sentiment classification systems. The findings from this study, serving as a foundational reference, could subsequently guide enhancements to the systems.

## 8.3 Cross-lingual Subjectivity Classification

This part of the thesis is built on a paper called “Czech Dataset for Cross-lingual Subjectivity Classification” (Přibáň & Steinberger, 2022), where we introduce a new Czech subjectivity dataset of 10k manually annotated subjective and objective sentences from movie reviews and descriptions. Our prime motivation is to provide a reliable dataset that can be used with the existing English dataset as a benchmark to test the ability of pre-trained multilingual models to transfer knowledge between Czech and English and vice versa.

### 8.3.1 Introduction

Subjectivity classification (J. M. Wiebe et al., 1999) is one of the integral parts of SA. Its basic purpose is to determine if a sentence or phrase is subjective or objective. Subjective text expresses personal feelings, views, beliefs or opinions and objective sentences hold or describe some factual information (B. Liu, 2012). It can be further used to improve other tasks such as polarity detection or information extraction (Pang & Lee, 2004; J. M. Wiebe et al., 1999). Nowadays, the subjectivity classification is often used as a benchmark test (Bragg et al., 2021; Reimers & Gurevych, 2019; S. Wang et al., 2021; Zhao et al., 2015) in transfer learning to test abilities and language understanding of pre-trained BERT-like language models based on the Transformer architecture (Vaswani et al., 2017).

Evaluation of the pre-trained models for transfer learning is a crucial part of their development. The well-known GLUE (A. Wang et al., 2018) and SuperGLUE (A. Wang et al., 2019) benchmarks are available for English. These benchmarks contain a set of diverse tasks that allow a thorough evaluation of English pre-trained models.

Our main motivation is to partly fill this gap and contribute a bit by introducing a reliable Czech dataset that can be used for cross-lingual evaluation. We intend to use the dataset to test the cross-lingual abilities of pre-trained multilingual models in pair with the existing English dataset (Pang & Lee, 2004) as a benchmark for zero-shot cross-lingual subjectivity classification. Thus, it partly tests the ability of pre-trained multilingual models to transfer knowledge between Czech and English. We are aware that to properly evaluate any pre-trained model, a diverse set of tasks is needed, but we believe that even one task can be helpful in the evaluation process. To the best of our knowledge, there is no subjectivity dataset for the Czech language, therefore, our secondary goal is to extend the available dataset resources for Czech.

In summary, we present the first Czech dataset for the subjectivity classification task that consists of 10k manually annotated sentences from movie reviews and movie descriptions. Secondly, we provide an additional dataset of 200k sentences labeled in a distant supervised way (automatically). The automatic labeling is based on the idea from Pang and Lee (2004) that movie reviews contain mostly subjective sentences and the movie descriptions usually consist of objective sentences. We describe the process of building and annotating the dataset. The dataset is annotated by two annotators and the Cohen's $\varkappa$ (Cohen, 1960) inter-annotator agreement between them reaches 0.83. We perform experiments with two multilingual mBERT (Devlin et al., 2019) and XLM-$\mathrm{R_{Large}}$ (Conneau et al., 2020) and three monolingual Transformer-based models on the new Czech dataset and provide a competitive baseline of 93.56% of accuracy. Next, we conduct experiments with the same two multilingual models on the English dataset to be able to compare our cross-lingual experiments. Our results for the monolingual experiments with English are on par with the current state-of-the-art results. Finally, we evaluate the multilingual models and their ability to transfer knowledge between English and Czech on the zero-shot cross-lingual classification task. The cross-lingual experiments show that using only English data for fine-tuning the XLM-$\mathrm{R_{Large}}$, the model can achieve worse results only by 2.8% on the Czech dataset compared to the model trained on Czech data. When the model is trained using only the Czech data, the result on the English dataset is roughly 4.4% worse than the current state-of-the-art results.

Our main contributions of this thesis part are the following: 1) we introduce the first Czech subjectivity dataset that allows cross-lingual evaluation in pair with the existing English dataset. 2) We perform a series of monolingual and cross-lingual experiments. We set a competitive baseline for the new Czech dataset. We compare the abilities of two multilingual models to transfer knowledge between Czech and English in the subjectivity classification task. 3) We release[14] the dataset and code freely for research purposes, including the dataset splits for easier comparison and reproducibility of our results. Please see our paper (Přibáň & Steinberger, 2022) for related work.

## 8.3.2 Subjectivity Dataset

We provide two datasets[14] of subjective and objective Czech sentences from movie reviews and movie descriptions (plot summaries), respectively. We use the mentioned idea from Pang and Lee (2004), in which the authors automatically created an English dataset (`Subj-EN`) of 10k subjective and objective sentences. They assume that the descriptions are mostly objective and the reviews are subjective. This assumption is valid in most cases, but there can also be objective sentences in reviews and subjective sentences in descriptions. The number of these noisy samples differs in both cases, as you can see in Table 8.15.

For this reason, we decided to create a manually annotated dataset (`Subj-CS`) of 10k examples that should eliminate the incorrect occurrences as much as possible. Secondly, we automatically built an additional dataset (`Subj-CS-L`) of 200k sentences using almost the same approach[15] as in Pang and Lee (2004).

[14] The datasets and code are freely available for research purposes at https://github.com/pauli31/czech-subjectivity-dataset

[15] Based on our observations in the dataset, we decided to use sentences or phrases with at least six tokens,

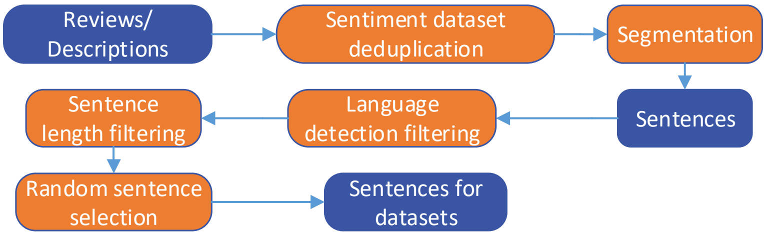

Reviews/ Descriptions
Sentiment dataset deduplication
Segmentation
Sentences
Language detection filtering
Sentence length filtering
Random sentence selection
Sentences for datasets

Figure 8.2: Data cleaning pipeline visualization.

#### 8.3.2.1 Cleaning and Obtaining Data

We acquired roughly 4M reviews and 735k descriptions from Czech Movie Database[16] (CSFD) during October 2021. The Czech sentiment movie review dataset (Habernal et al., 2013) also consists of reviews from CSFD. We assume that in the future, our dataset can be used in combination with the sentiment dataset; therefore, we decided to remove the sentiment reviews from the data we downloaded. We matched and removed about 74k reviews out of a total of 91k from the sentiment dataset. The remaining 17k reviews were most likely changed or removed from the CSFD website since the authors of the sentiment dataset originally downloaded the data in 2013. Next, we split the reviews and descriptions into sentences by UDPipe 2 (Straka, 2018)[17].

Some of the texts (mostly reviews) were written in other languages (most often Slovak and English). We filter out these out[18] and we keep only Czech sentences. Finally, we filter out sentences with less than six tokens. See Figure 8.2 for the cleaning pipeline visualization.

The entire cleaning process resulted in 884k and 19M sentences (phrases) from descriptions and reviews, respectively. We randomly selected 40k sentences from the obtained reviews and descriptions for manual annotation and 200k sentences (100k from reviews and 100k from descriptions) for the automatically created dataset. The remaining sentences are not utilized.

### 8.3.3 Annotation Procedure

Two native Czech speakers performed the annotation. Even though the subjectivity classification may seem like an easy task, it proved to be rather difficult for some sentences to assign a subjectivity label.

Firstly, the task of subjectivity classification was explained to the annotators along with the meaning of the subjective and objective sentences according to the definition in B. Liu (2012). We summarize the annotation guidelines in Section 8.3.4.

but they used sentences longer than nine tokens.

[16] https://www.csfd.cz

[17] We use the *czech-pdt-ud-2.5-191206.udpipe* model.

[18] We use the Python package `langdetect` available at https://pypi.org/project/langdetect/ to detect the language.

During the first annotation stage, each of the annotators were asked to label a common set of 100 sentences with one of three labels: `subjective, objective` and `trash`, see Section 8.3.4 for their description. We use the trash label because, despite our best data cleaning efforts, there were still undesirable texts: e.g., short sequences of words that do not make any sense (random words), only numbers and other characters, sentences in other languages, texts that were obviously incorrectly segmented and made no sense etc.

After the first 100 annotated sentences, the annotators discussed the conflicts to clarify and improve the annotation guidelines. Based on the discussion, we decided to extend the annotation labels by two more `unsure` and `question`.

The questions appeared to be rather problematic. The subjectivity was not clear very often and thus, we decided to exclude them. In addition, the questions are only in a tiny part of the data, i.e., 1.73% and 2.41% for review and description sentences, respectively, see Table 8.15.

The unsure label was added because, for some sentences, the annotators were not able to assign the subjectivity. For example, sentences for which a context (previous sentence) is needed to decide, sentences that describe a movie or event but contain some clearly subjective adjective(s) can be perceived or interpreted as subjective or objective depending on an individual person. Other problematic sentences are commands, wishes or parts of poems and rhymes. We list some of the problematic sentences labeled by both annotators as unsure:

(1) *"Všechno ovšem tak snadné řešení nemá." – "Not everything has such an easy solution."*

(2) *"To je dobrý důvod pro to, aby byla Japonsku vyhlášena válka." – "That's a good reason to declare war on Japan."*

(3) *"Dnes večer je to však díky napjaté atmosféře velmi obtížné." – "Tonight, the tense atmosphere makes it very difficult."*

(4) *"Drastický horor, při kterém tuhne krev v žilách" – "Drastic horror that makes your blood run cold"*

(5) *"Tak se o to postará příroda sama!" – "Nature will take care of it!"*

We decided to add these additional labels because we wanted to assign labels only in cases where the annotators are very confident with their annotations and thus obtain more reliable annotations without controversial examples and dataset of high quality.

After the update of the annotation guideline, both of the annotators assigned labels to the same 2,034 sentences. The Cohen's $\varkappa$ (Cohen, 1960) inter-annotator agreement for this 2k sentences reaches 0.68 for all five labels. Because we provide the dataset only with the objective and subjective labels, we exclude any sentence with at least one[19] of the trash, unsure or question labels. Thanks to this filtration, we obtained 1,668 sentences only with the subjective and objective labels. Cohen's $\varkappa$ for this subset is 0.83, representing a fairly

[19] Each sentence has two labels – one from each annotator.

good agreement level. The remaining 141 conflict sentences are then resolved with the help of a third person.

Finally, almost 5,000 sentences were annotated by each of the two annotators, resulting in a total of 11,907 annotated sentences, see Table 8.15. We can see that the subjective and objective sentences are relatively balanced in the annotated samples and we believe that this reflects the real data distribution. Even though we obtained more than 5,000 sentences with the subjective and objective labels, we cut the annotations to have exactly 5,000 examples for each of the two labels. We decided to provide a perfectly balanced dataset since it allows easier comparison and evaluation of experiments. We use only the sentences with the subjective and objective labels, i.e., 10,000 sentences. We refer to this dataset as `Subj-CS`.

The entire procedure of annotation can be summarized into the following steps:

1. Each annotator annotated 100 sentences as `subjective`, `objective` or `trash`.

2. Every conflict in the first 100 sentences was discussed separately between the annotators to clarify and improve the annotation guideline. We extended the annotation guideline by two more labels: `unsure` and `question`.

3. 2,034 sentences are annotated by each annotator (1,668 as subjective or objective with 141 conflicts). Cohen's $\kappa$ reaches 0.83 for subjective and objective sentences. The conflicts are resolved by a third person.

4. Almost 10k other sentences are annotated in total by both annotators. The annotations are cut down to contain exactly 5,000 subjective and objective sentences.

#### 8.3.3.1 Annotation Statistics

The manual annotation resulted in a total of 11,907 annotated sentences with one of five labels, see Table 8.15. During the annotation procedure, we set the limit of at most 15 review sentences for the same movie and at most three description sentences in the 40k sentences selected for the manual annotation. However, the average number of sentences for the same movie is only 1.43 and 1.02 for review and description sentences, respectively.

| Label | Reviews | Descriptions | Total |
|---|---|---|---|
| unsure | 866 / 13.11% | 457 / 8.62% | 1,323 |
| **object.** | **726 / 10.99%** | **4,464 / 84.22%** | **5,190** |
| **subj.** | **4,794 / 72.57%** | **208 / 3.92%** | **5 002** |
| quest. | 114 / 1.73% | 128 / 2.41% | 242 |
| trash | 106 / 1.60% | 44 / 0.83% | 150 |
| Total | 6,606 / 100% | 5,301 / 100% | 11,907 |

Table 8.15: Annotation statistics for subjective and objective

As we assumed, a considerable percentage of sentences in reviews are not subjective (only 72.57% of sentences are subjective). Similarly, a relatively large part of sentences in the movie descriptions are not objective (84.22% of the sentences are objective).

### 8.3.4 Annotation Guideline

The annotators were instructed to annotate a given sentence with one of five labels. Based on the subjectivity description from B. Liu (2012), Pang and Lee (2004), and J. M. Wiebe et al. (1999), the sentence should be annotated as subjective if it expresses or evokes some personal feelings, views, beliefs or the sentence holds an opinion about entities, events or their properties (mostly movies in our case) from the non-objective point of view. For example:

*"Samotný film se mi líbil, ale nepřekvapil." – "I liked the movie itself, but it didn't surprise me."*

The sentence should be annotated as objective if it contains some factual information about an entity, event or their properties but does not hold a personal or subjective opinion about it and it does not try to convince or impose some opinion to the reader, for example:

*"Maurice žije a pracuje v jižní Francii." – "Maurice lives and works in the south of France."*

The disputed and controversial sentences, sentences where the annotator is not sure about its subjectivity or sentences for which context from previous text is needed to decide, should be annotated with the unsure label, see Section 8.3.3 for examples. The trash label is used for sentences or phrases that do not make any sense or contain random words, characters or numbers. The question label is used for sentences that are questions.

### 8.3.5 Automatic Dataset

Besides the manually annotated dataset, we also built a large dataset (named `Subj-CS-L`) in a distant supervised way using the same approach as in Pang and Lee (2004). We labeled 100k review sentences as subjective and 100k movie description sentences as objective ones. All sentences have to have at least six tokens. We believe that even if the dataset contains some incorrect labels, it could be useful in combination with the manually created dataset, for example, in an unsupervised pre-training.

### 8.3.6 Experimental Setup

For the experiments, we split the `Subj-CS` dataset into three parts with the following ratio: 75% for training, 5% for the development evaluation and 20% for testing. For the cross-lingual experiment with the `Subj-CS-L` dataset from Czech to English, we use 5% as the development evaluation data and the rest is used for training.

Because there is no official split for the English dataset (Pang & Lee, 2004), we use 10-fold cross-validation for the English monolingual experiments to be able to compare our results with other papers. We also split the English dataset into training, development and testing parts with the same test size (see Table 8.16) that was used in S. Wang et al. (2021)[20].

| Dataset | Name | Subjective | Objective | Total |
|---|---|---|---|---|
| Subj-CS | cs-train | 3,750 | 3 750 | 7 500 |
| | cs-dev | 250 | 250 | 500 |
| | cs-test | 1,000 | 1,000 | 2 000 |
| | | 5,000 | 5,000 | 10 000 |
| Subj-CS-L | cs-L-train | 95,000 | 95,000 | 190,000 |
| | cs-L-dev | 5,000 | 5,000 | 10,000 |
| | | 100,000 | 100,000 | 200,000 |
| Subj-EN | en-train | 3,764 | 3,736 | 7,500 |
| | en-dev | 231 | 269 | 500 |
| | en-test | 1,005 | 995 | 2,000 |
| | | 5,000 | 5,000 | 10,000 |

Table 8.16: Datasets statistics.

In our experiments, we use solely the pre-trained BERT-like models based on the encoder part of the original Transformer architecture (Vaswani et al., 2017). The modified language modeling task is used to pre-train all the models. More concretly, we employ three Czech monolingual models *Czert-B* (Sido et al., 2021), *RobeCzech* (Straka et al., 2021), *Czech Electra* model (Kocián et al., 2022), two multilingual models *mBERT* (Devlin et al., 2019), *XLM-R* (Conneau et al., 2020) and the original monolingual English *BERT* model (Devlin et al., 2019). We fine-tune the Transformer-based models in the same way as is described in Section 7.3.2.1.

### 8.3.7 Experiments

We performed a series of experiments with Transformer-based models to set baseline results for the new Czech dataset and verify its usability as a cross-lingual benchmark dataset between Czech and English. The experiments can be categorized into two groups – *monolingual* and *cross-lingual*.

In monolingual experiments for Czech, we fine-tune the three Czech monolingual BERT-like models, i.e., *Czert-B*, *RobeCzech* and *Czech Electra* model and two multilingual models *mBERT* and *XLM-R*. We use the same two multilingual models and the original *BERT* model for English. In *cross-lingual* experiments, we test the ability to transfer knowledge between Czech and English using the *zero-shot cross-lingual* classification. We fine-tune the multilingual models only on the dataset in one language (Czech or English) and then evaluate the fine-tuned model on the dataset in the other language.

[20] Unfortunately, they do not provide any script or details to obtain the identical split. In other words, we do not know which sentences belong to the training part and which to the testing part.

We always fine-tune[21] on training data and measure the results on the development and testing data parts. We select the model that performs best on the development data and report the results using average accuracy with 95% confidence intervals (we repeat each experiment at least 12 times). See Appendix A.4 for the hyper-parameters details for the reported experiment results.

### 8.3.7.1 Czech Monolingual Experiments

For Czech monolingual experiments, we use two types of training data. The training part (`cs-train`) of the manually labeled dataset `Subj-CS` and the entire automatically created dataset `Subj-CS-L` (marked as `cs-L-train`). In both cases, we evaluate models on the development (`cs-dev`) and testing (`cs-test`) parts of the `Subj-CS` dataset. We report the results in Table 8.17.

| Model | Subj-CS (cs-train) | Subj-CS-L (cs-L-train) |
|---|---|---|
| | cs-test | cs-test |
| Czech Electra | 91.9 ± 0.3 | 91.2 ± 0.1 |
| Czert-B | 92.9 ± 0.2 | 91.8 ± 0.1* |
| RobeCzech | 93.3 ± 0.2* | 91.6 ± 0.1 |
| mBERT | 91.2 ± 0.2 | 91.1 ± 0.1 |
| XLM-R$_{\text{Large}}$ | **93.6 ± 0.1** | **92.0 ± 0.1** |

Table 8.17: Results for Czech monolingual experiments reported as average accuracy for the testing `cs-test` data part. The * symbol denotes results containing intersection in confidence interval with the best model.

As we expected, the XLM-R$_{\text{Large}}$ model achieves the highest average accuracy of 93.56% for both types of training data. Despite the highest achieved accuracy, there is an intersection in its confidence interval with RobeCzech model for the `cs-train` data (the * symbol in Table 8.17). Thus, we can conclude that RobeCzech and XLM-R$_{\text{Large}}$ perform very similarly for Czech monolingual experiments. Thanks to the XLM-R$_{\text{Large}}$ size (and its relatively large hardware training requirements), one could prefer the smaller RobeCzech model. The last observation is that all the models achieve better results with the `cs-train` data part. We expected XLM-R$_{\text{Large}}$ to perform very well because it is the largest model and as shown in Přibáň and Steinberger (2021) it usually outperforms smaller monolingual models.

### 8.3.7.2 English Monolingual Experiments

In English monolingual experiments, we evaluate the English dataset on training (`en-train`), development (`en-dev`) and testing (`en-test`) data split. Because models from other works

[21] The composition of data used for training and evaluation depends on the corresponding experiment. In the case of English monolingual experiments for the 10-fold split, we did not use any development data.

(Amplayo et al., 2018; Khodak et al., 2018; Nandi et al., 2021a; Reimers & Gurevych, 2019; Zhao et al., 2015) are evaluated on the 10-fold split, we evaluate the models also on the 10-fold split (`en-10-fold`) to be able to compare their and ours results.

| Model | en-test | en-10-fold |
|---|---|---|
| BERT | 96.6 ± 0.2 | 96.9 ± 0.3 |
| mBERT | 95.9 ± 0.1 | 96.0 ± 0.2 |
| XLM-R$_{\text{Large}}$ | **97.3 ± 0.1** | **97.3 ± 0.2** |
| (S. Wang et al., 2021)† | 97.4 ± 0.1 | - |
| (Nandi et al., 2021a) | - | 97.3 |
| (Zhao et al., 2015) | - | 95.5 |
| (Amplayo et al., 2018) | - | 94.8 |
| (Khodak et al., 2018) | - | 94.7 |
| (Reimers & Gurevych, 2019) | - | 94.5 |

Table 8.18: Results for English monolingual experiments reported as average accuracy for the testing `en-test` and `en-10-fold` data parts. The model in paper marked with the † symbol uses the same test size, but the distribution of sentences is different in each split part and they also use the standard deviation instead of the confidence interval.

As shown in Table 8.18, the XLM-R$_{\text{Large}}$ performs best among the other two Transformer models without any intersection of confidence intervals between the different models. We can also see that the results for `en-test` and `en-10-fold` are very similar and their confidence intervals overlap for the same model pairs (but different training data). Based on this observation, we assume that the results for `en-test` and `en-10-fold` are comparable to each other; thus, in the cross-lingual experiments, English is evaluated only on the `en-test` part. We compare our results with the current state-of-the-art results (rows below the dashed line in Table 8.18). Most of the other works use the 10-fold cross-validation and our results also achieve the SotA results and are on par with them.

#### 8.3.7.3 Cross-lingual Experiments

We perform three types of cross-lingual experiments: from English to Czech, from Czech to English and joint training and evaluation of both languages. The first two are also known as a zero-shot cross-lingual classification because the model is fine-tuned only on data from one language (source language) and evaluated on data from the second language (target language). The model has never seen the labeled data from the target language.

For the experiments from English to Czech (`EN→CS`), we fine-tune the multilingual models on English `en-train` data and we evaluate them on the `en-dev` and `cs-test`. We select the model that performs best on the `en-dev` (i.e., the same best model as for the English monolingual data) and we report results for the `cs-test` data in Table 8.19[22].

[22] We also include the monolingual results for an easier comparison of the results.

| Model | EN → CS | | Monoling. (cs-train) |
|---|---|---|---|
| | en-dev | cs-test | cs-test |
| mBERT | 95.4 ± 0.2 | 86.2 ± 0.3 | 91.2 ± 0.2 |
| XLM-R$_{\text{Large}}$ | 97.6 ± 0.2 | **90.8** ± **0.3** | 93.6 ± 0.1 |

Table 8.19: Accuracy results for cross-lingual experiments from English to Czech along with the results for models trained on monolingual data.

The XLM-R$_{\text{Large}}$ model clearly outperforms the mBERT model by 4.5% but is worse than the same model that was trained on monolingual data roughly by 2.8%. In the case of mBERT, the results are much worse (5% difference) than the model trained only on monolingual data.

For experiments from Czech to English (`CS→EN`), we fine-tune the models on `cs-train` and evaluate on `cs-dev` and `en-test`. We select the model that performs best on `cs-dev`.

We also train the model on the `cs-L-train` data, but in this case, we select the model that performs best on the `en-dev` data from the target language (English). We use the `en-dev` for selecting the best model because we found out that if we use `cs-L-dev`, we get much worse results (up to 20% worse) for the `en-test`. We are aware of this simplification of the zero-shot cross-lingual classification task, but otherwise, we would not be able to obtain a model with reasonable results. The results are stated in Table 8.20.

| Model | CS → EN (cs-train) | | CS → EN (cs-L-train) | | Monolingual (en-train) |
|---|---|---|---|---|---|
| | cs-dev | en-test | en-dev | en-test | en-test |
| mBERT | 92.1 ± 0.4 | 89.0 ± 0.9 | 85.8 ± 0.9 | 85.5 ± 0.9 | 95.9 ± 0.1 |
| XLM-R$_{\text{Large}}$ | 94.4 ± 0.4 | **92.9** ± **0.4** | 93.4 ± 0.2 | **91.0** ± **0.3** | 97.3 ± 0.1 |

Table 8.20: Accuracy results for cross-lingual experiments from Czech to English along with the results for models trained on monolingual data.

For both models trained on Czech data (`cs-train` and `cs-L-train`), the results are even worse in comparison to the previous experiment from English to Czech. For example, the difference between XLM-R$_{\text{Large}}$ trained on `cs-train` and XLM-R$_{\text{Large}}$ trained on English `en-train` data is 4.4%, whereas in the case of the previous experiment from English to Czech, it was only 2.8%. The results of the models trained on the `cs-L-train` are significantly worse (10% for mBERT).

| | Joint (cs-train + en-train) | | Monolingual (cs-train) | Monolingual (en-train) |
|---|---|---|---|---|
| Model | cs-test | en-test | cs-test | en-test |
| mBERT | 91.1 ± 0.2 | 95.7 ± 0.2 | 91.2 ± 0.2 | 95.9 ± 0.1 |
| XLM-R$_{\text{Large}}$ | **93.9** ± **0.2** | **96.9** ± **0.1** | 93.6 ± 0.1 | 97.3 ± 0.1 |

Table 8.21: Accuracy results for models jointly trained on English and Czech data along with the results for models trained on monolingual data.

Finally, we fine-tune the models jointly on `cs-train` and `en-train`, i.e., on both languages at once. We average the results obtained on `cs-dev` and `en-dev` and we select the model that achieves the highest average value. We report the results for the `cs-test` and `en-test` in Table 8.21. We can see that the obtained results are almost identical or slightly different compared to the models trained only on monolingual data. Thus, we can conclude that joint fine-tuning has no beneficial contribution in these cases.

#### 8.3.7.4 Discussion

We summarize and mention some of our main findings and conclusions from the experiments. Even though the Czech Electra model is significantly smaller than all the other models, it achieves very competitive results compared to the other models. Thanks to its smaller size, fine-tuning is much easier and faster.

The XLM-R$_{\text{Large}}$ model dominates the results, but it is also several times larger than the other models, see Table 8.2. Despite the worse results in the cross-lingual experiments, we can state that generally, the XLM-R$_{\text{Large}}$ (and in some cases even mBERT) is relatively capable of transferring knowledge between Czech and English and vice versa, at least for the subjectivity classification task. The confidence intervals for results obtained in cross-lingual experiments are usually larger than the ones for the monolingual results. Thus, we consider the cross-lingual results less stable.

During the cross-lingual experiments, we select the best model based on development results for the source language. We suppose that such a setting should be more difficult and challenging than choosing the model according to the target language results. This setting is much closer to the potential usage of the multilingual models in the industry or to solving practical, real-world tasks that are often more complicated. We do not use this approach for models trained on the large data that were obtained automatically because of its poor results.

Based on the cross-lingual results, we believe that for knowledge transfer between languages, a smaller but high-quality (manually annotated) dataset is better and more important than a large automatically created dataset to obtain more reliable results for downstream tasks.

### 8.3.8 Subjectivity Classification Conclusion

As part of the described work, we introduced the first Czech subjectivity dataset `Subj-CS` that consists of 10k manually annotated subjective and objective sentences from movie reviews and descriptions. In addition, we automatically compiled a second, much larger dataset of 200k sentences. Both datasets are freely available for research purposes. We describe the process of building and annotating the dataset. Two annotators annotated the dataset with Cohen's $\varkappa$ inter-annotator agreement equal to 0.83. We provide a summary of the annotation guidelines used by the annotators.

We perform a series of monolingual experiments with five pre-trained BERT-like models to obtain the baseline results for the newly created Czech dataset and we are able to achieve

93.6% of accuracy with the XLM-R$_{\text{Large}}$ model. We also perform monolingual experiments for the existing English subjectivity dataset with three models obtaining 97.3% of accuracy, which is on par with the current state-of-the-art results for this dataset. Finally, we conduct zero-shot cross-lingual subjectivity classification to verify the usability of our dataset as the cross-lingual benchmark for pre-trained multilingual models that allow transfer learning.

Our experiments confirm that we provide a dataset of high quality and it can be used as an evaluation benchmark to test the ability of models to transfer knowledge between Czech and English.

## 8.4 Improving Aspect-based Sentiment Analysis with Semantic Role Labeling

This section presents results from the paper called "Improving Aspect-Based Sentiment with End-to-End Semantic Role Labeling Model" (Přibáň & Pražák, 2023). The paper focuses on the task of ABSA for the Czech and English languages.

We introduce a novel approach aiming at enhancing the performance of ABSA task. Our approach utilizes information from a Semantic Role Labeling (SRL) model to improve the results of the ABSA task. Firstly, We propose a novel end-to-end SRL model that effectively captures the structured semantic information within the Transformer's hidden state. This end-to-end SRL model is used to extract the information needed to improve the ABSA task. Next, we incorporate this information into the ABSA model and improve its performance. The approach is evaluated on English and Czech ABSA datasets, showing its effectiveness by employing ELECTRA-small models for both languages. Our combined models improve the performance of the aspect category polarity task by more than 4% for Czech. Moreover, we achieved new state-of-the-art results for the Czech dataset.

### 8.4.1 Tasks Definition for ABSA

Aspect-based Sentiment Analysis (B. Liu, 2012; Pontiki et al., 2014) focuses on detecting aspects (e.g., food or service in the restaurant reviews domain) and determining their polarity, enabling a more detailed analysis and understating of the expressed sentiment. In this work, we rely on the definition of ABSA tasks provided by Pontiki et al. (2014). The task is divided into four subtasks: *Aspect term extraction* (TE), *Aspect term polarity* (TP), *Aspect category extraction* (CE), and *Aspect category polarity* (CP).

We aim at the CE and CP subtasks,[23] and we treat them as classification tasks, see Section 8.4.4.2. As depicted in Figure 8.3, the goal of the CE subtask is to detect a set of aspect categories within a given sentence, i.e., for a given text $S = \{w_1, w_2, \dots w_n\}$ assign set $M = \{a_1, a_2, \dots, a_m\}$ of $m$ aspect categories, where $m \in [0, k]$, $M \subset A$ and $A$ is a set of $k$ predefined aspect categories $A = \{a_1, a_2, \dots, a_k\}$. The goal of CP is to assign one of the predefined

[23] See Pontiki et al. (2014) for a detailed description of all the subtasks.

polarity labels $p$ for each of the given (or predicted) aspect categories of the set $M$ for the given text $S$, where $p \in P = \{positive, negative, neutral\}$.

*"The burger was excellent but the waitress was unpleasant"*

CE ⇒ food, service

CP ⇒ food:*positive*, service:*negative*

Figure 8.3: Example of CE and CP subtasks of ABSA.

### 8.4.2 Semantic Role Labeling

The Semantic Role Labeling task (Gildea & Jurafsky, 2002) belongs among shallow semantic parsing techniques. The SRL aims to identify and categorize semantic relationships or *semantic roles* of given *predicates*. Verbs, such as "believe" or "cook", are natural predicates, but certain nouns are also accepted as predicates. The simplified definition of semantic roles is that semantic roles are abstractions of predicate arguments. For example, the semantic roles for "believe" can be *Agent* (a believer) and *Theme* (a statement) and for "cook" *Agent* (a chef), *Patient* (a food), *Instrument* (a device for cooking) – see examples in Figure 8.4. The theory of predicates and their roles is very well established in several linguistic resources such as PropBank (Palmer et al., 2005) or FrameNet (Baker et al., 1998).

(1) [He]$_{\text{AGENT}\,|\text{A0}}$ believes [in what he plays] $_{\text{THEME}|\text{A1}}$ .

(2) Can [you] $_{\text{AGENT}\,|\text{A0}}$ cook [the dinner] $_{\text{PATIENT}|\text{A1}}$ ?

Figure 8.4: Examples of SRL annotations.

### 8.4.3 Approach Motivation Details

For our experiments, we need an end-to-end SRL model which encodes most of the information in the Transformer's hidden state. However, to the best of our knowledge, there is no such model. As a result, we propose a new end-to-end model to fulfil this need. The model effectively captures the structured semantic information and offers enhanced compatibility with other NLP tasks. Unlike other SRL BERT-based models (Papay et al., 2022; Shi & Lin, 2019), our proposed model integrates the complete semantic information into the hidden state of the Transformer. This end-to-end SRL model is particularly well-suited for combination with the ABSA task, as it encapsulates the entire predicate-argument structure of the sentence within a single hidden state, in contrast to the approach of Shi and Lin (2019), which encodes each predicate separately and requires gold predicates on input. Our model, on the other hand, only requires the input text.

We assume that leveraging the syntax and semantic information extracted from SRL can enhance the performance of the aspect category polarity subtask. This assumption is grounded in the notion that the SRL information has the potential to unveil valuable and pertinent relations between entities within a given sentence, which play a crucial role in

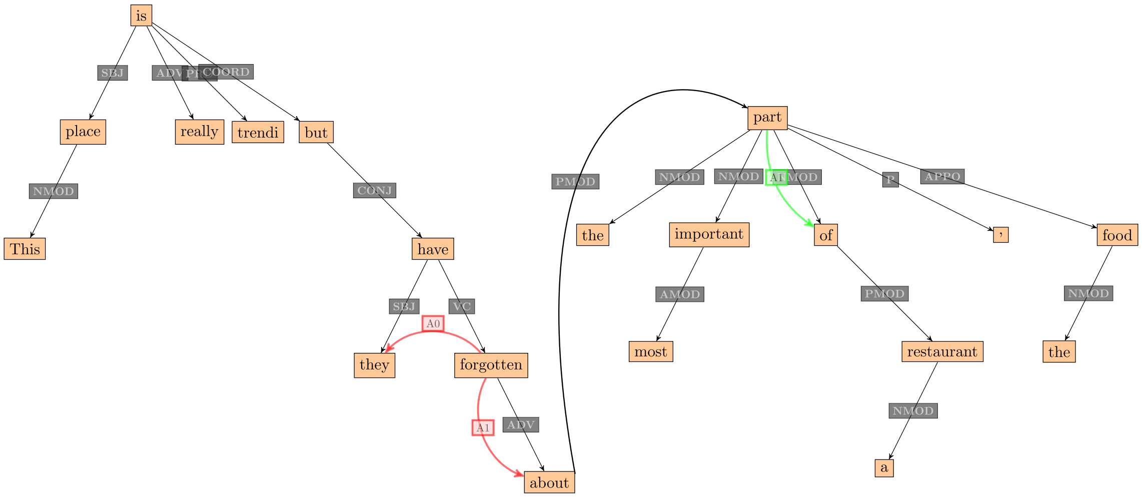


Figure 8.5: Example of syntactic and semantic parse tree of the following sentence "*This place is really trendy but they have forgotten about the most important part of a restaurant, the food*".

accurate aspect category polarity predictions. This holds particularly true for longer and more complex sentences, where a broader contextual understanding becomes essential. To illustrate this point, consider the annotation depicted in Figure 8.6, where we can observe the SRL relation extracted (see Figure 8.5) between the words *forgotten* and *food*. The information about this relation can help to understand the model that these words are related and help the model to predict the negative polarity of the food aspect category.

> "*This place is really trendy but they have forgotten about the most important part of a restaurant, the food.*"
>
> CE ⇒ food, ambience
>
> CP ⇒ food:*negative*, ambience:*positive*

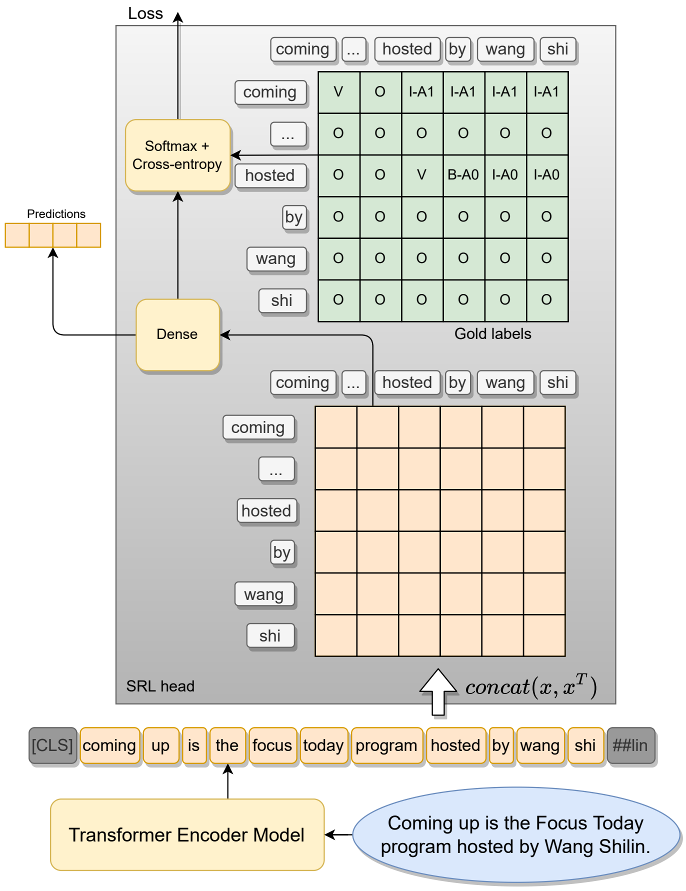
Figure 8.6: Example of CE and CP annotations.

## 8.4.4 Models

To find an effective way to combine the models, we first fine-tune the individual models separately to find the optimal set of hyper-parameters for individual tasks. Moreover, we need an SRL fine-tuned model as the input for the combined models. For ABSA, we adopt the model proposed by Sun, Huang, and Qiu (2019). We propose a new SRL end-to-end model specifically designed for seamless integration with other tasks.

### 8.4.4.1 Semantic Role Labeling

Our goal is to train a universal encoder that effectively captures SRL information from a plain-text input. To accomplish this, we propose an end-to-end model with a single projection layer on the top of the ELECTRA encoder (or any other pre-trained language model). This way, all the information useful to predict role labels is encoded in the last hidden state of the


Loss
Softmax + Cross-entropy
Predictions
Dense
Gold labels
SRL head
$concat(x, x^T)$
[CLS] coming up is the focus today program hosted by wang shi ##lin
Transformer Encoder Model
Coming up is the Focus Today program hosted by Wang Shilin.
coming ... hosted by wang shi

| | coming | ... | hosted | by | wang | shi |
|---|---|---|---|---|---|---|
| coming | V | O | I-A1 | I-A1 | I-A1 | I-A1 |
| ... | O | O | O | O | O | O |
| hosted | O | O | V | B-A0 | I-A0 | I-A0 |
| by | O | O | O | O | O | O |
| wang | O | O | O | O | O | O |
| shi | O | O | O | O | O | O |

Figure 8.7: End-to-end SRL model architecture.

encoder. Consequently, we can use this representation in other tasks. Although our end-to-end model exhibits lower performance than the commonly used BERT SRL model (Shi & Lin, 2019; Sido et al., 2021), we believe it is more suitable for this task.

In our end-to-end model, we first encode the whole sentence and then iterate over all possible word pairs (the first word is a potential predicate and the second is a potential argument). For each potential predicate-argument pair, we first concatenate the representations of predicate and argument and then classify the argument role. If the potential predicate is not a real predicate word or the potential argument is not an argument of the predicate, the role of the pair is set to *Other*. If a word is represented by multiple subword tokens, only the first token is classified. This is common practice in tagging tasks where the model learns to encode the semantics of a multi-token word into the first subword, then each word has a single token on the output for its classification.

Our approach differs from that of Shi and Lin (2019) in terms of how the predicate-argument structure of the sentence is encoded within the Transformer model. While Shi and Lin (2019) encodes each predicate separately and requires gold predicates on input, our model only requires plain text as input. In other words, our model requires only text as input, but the model proposed by Shi and Lin (2019) operates on pairs of text-predicate, producing representations solely for the input pair rather than the entire SRL output encompassing all predicates within the sentence. Figure 8.7 shows the schema of our end-to-end SRL model.

To implement our multitask approach, it is necessary to have the same format of input (i.e., plain text) for both tasks that are combined. This is the reason why we need our end-to-end SRL model. For multitask learning, we need a general-purpose model, the same for both tasks. The task-specific models may yield better results on the SRL task, but they are specifically oriented only on the SRL task and make their integration with ABSA or utilization in multitask learning challenging, if not impossible.

### 8.4.4.2 Aspect-based Sentiment

As we mentioned, we tackle the CE and CP subtasks of ABSA, as one classification task. We adopt the same approach as Sun, Huang, and Qiu (2019), and we construct auxiliary sentences and convert the subtasks to a binary classification task.

We use the NLI-B approach from Sun, Huang, and Qiu (2019) to build the auxiliary sentences. For each sentence, we build multiple auxiliary pseudo sentences that are generated for every combination of all polarity labels and aspect categories[24]. Each example has a binary label $l \in \{0, 1\}$; $l = 1$ if the auxiliary sentence corresponds to the original labels, $l = 0$ otherwise. We also add the artificial polarity class *none* that has assigned binary label $l = 1$ if there is no aspect category for a given sentence. The pseudo auxiliary sentence consists only of a polarity label and aspect category in a given language. For example, the auxiliary sentences for all aspects of the sentence "*The burger was excellent but the waitress was unpleasant*" are shown in Figure 8.8.

| label | | sentence | label | | sentence | label | | sentence | label | | sentence | label | | sentence |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | food | | | service | | | price | | | ambience | | | general |
| 1 | ⇒ | positive – food | 0 | ⇒ | positive – service | 0 | ⇒ | positive – price | 0 | ⇒ | positive – ambience | 0 | ⇒ | positive – general |
| 0 | ⇒ | negative – food | 1 | ⇒ | negative – service | 0 | ⇒ | negative – price | 0 | ⇒ | negative – ambience | 0 | ⇒ | negative – general |
| 0 | ⇒ | neutral – food | 0 | ⇒ | neutral – service | 0 | ⇒ | neutral – price | 0 | ⇒ | neutral – ambience | 0 | ⇒ | neutral – general |
| 0 | ⇒ | conflict – food | 0 | ⇒ | conflict – service | 0 | ⇒ | conflict – price | 0 | ⇒ | conflict – ambience | 0 | ⇒ | conflict – general |
| 0 | ⇒ | none – food | 0 | ⇒ | none – service | 1 | ⇒ | none – price | 1 | ⇒ | none – ambience | 1 | ⇒ | none – general |

Figure 8.8: Example of auxiliary sentences.

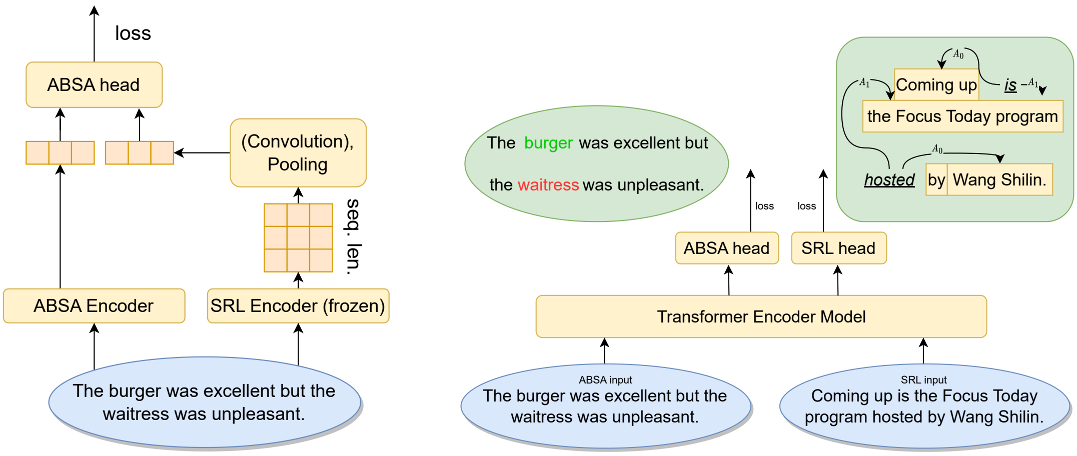
Each auxiliary sentence is combined with the original sentence and separated with `[SEP]` token and forms one training example, e.g., `[CLS]` *positive - food* `[SEP]` *the burger was excellent but the waitress was unpleasant* `[SEP]`. We fine-tune the pre-trained Transformer model for the binary classification task on all generated training examples as Sun, Huang, and Qiu (2019).

### 8.4.4.3 Combined Models

We propose multiple models designed to utilize SRL representation to enhance ABSA performance. The first model type predicts aspect and sentiment using concatenated representa-

[24] For English we have four polarity labels plus artificial label *none* and five aspect categories, i.e. 25 possible auxiliary sentences. For Czech, there are 20 possible sentences (3 + 1 polarity labels and five aspect categories).

tions from the SRL and ABSA encoders. The SRL encoder is pre-trained (pre-fine-tuned) on the SRL data, and its weights remain fixed during sentiment training. Since SRL is a token-level task, we need to reduce the sequential dimension before performing the concatenation step. To address this, we employ two approaches: simple average-over-time pooling (named *concat-avg*) and a convolution layer followed by max-over-time pooling (named *concat-conv*). Figure 8.9a shows the model architecture.


loss
ABSA head
(Convolution), Pooling
seq. len.
ABSA Encoder
SRL Encoder (frozen)
The burger was excellent but the waitress was unpleasant.
The burger was excellent but the waitress was unpleasant.
loss
loss
ABSA head
SRL head
Transformer Encoder Model
ABSA input
The burger was excellent but the waitress was unpleasant.
SRL input
Coming up is the Focus Today program hosted by Wang Shilin.
A0
A1
Coming up
is
A1
the Focus Today program
A0
hosted
by
Wang Shilin.

(a) Concat model architecture.

(b) Multi-task model architecture.

Figure 8.9: Illustration of combined models' architectures.

The last model uses standard multi-task learning. We utilize a single Transformer encoder with two classification heads: one for the sentiment (standard head for sequence classification) and the other for SRL (the head architecture is presented in Section 8.4.4.1). The model is trained using alternating batches, which means that we use different training data for both tasks and do not mix them in a batch. In a single batch, we provide only ABSA or SRL data. See Figure 8.9b model's architecture.

### 8.4.5 Datasets

For Semantic Role Labeling, we use OntoNotes 5.0 dataset (Weischedel et al., 2013) for English and CoNLL 2009 (Hajic et al., 2009) for Czech. As metrics, we report the whole role F1 score for both datasets. Additionally, for English, we report CoNLL 2003 official score as a comparative metric as it is the standard metric used with OntoNotes.

For English ABSA tasks, we utilize the widely-used English dataset from Pontiki et al. (2014) that consists of 3,044 train and 800 test sentences from the restaurant domain. The English dataset contains four sentiment labels: *positive*, *negative*, *neutral*, and *conflict*. Further, we split[25] the original training part of 3,044 sentences into development (10%) and training parts (90%).

[25] For both English and Czech we provide a script to obtain the same split distribution.

For Czech experiments, we employ the dataset from Hercig, Brychcín, Svoboda, Konkol, and Steinberger (2016) with 2,149 sentences from the restaurant domain. Unlike in the English dataset, there are only three polarity labels: *positive*, *negative*, and *neutral*. Because the dataset has no official split, we divided[25] the data into training, development, and testing parts with the following ratio: 72% for training, 8% for the development evaluation, and 20% for testing. Both Czech and English datasets contain five aspect categories: *food*, *service*, *price*, *ambience*, and *general*.

### 8.4.6 Models Fine-Tuning

For our experiments on English, we use the pre-trained *ELECTRA-small* model introduced by Clark et al. (2020), which has 14M parameters. For Czech, we employ the pre-trained monolingual model *Small-E-Czech* (Kocián et al., 2022) with the same size and architecture. Firstly, we train separate models for both tasks (ABSA and SRL) and select the optimal set of hyper-parameters on the development data. We then use the same hyper-parameters in combined models. For the details of hyper-parameters, see our publication (Přibáň & Pražák, 2023).

### 8.4.7 Results & Discussion

In our experiments, we aim to verify our idea that injected SRL information can improve the results of the ABSA task, particularly the CP subtask. We report the results of our end-to-end SRL model in Table 8.22. As we expected, our model performs worse than the model proposed by Shi and Lin (2019), but the results are reasonably high (considering that it does not have gold predicates on input).

| Model | EN | EN-conll05 | CS |
|---|---|---|---|
| (Shi & Lin, 2019) | 88.89 | 85.20 | 83.09 |
| end-to-end (ours) | 84.54 | 81.51 | 79.74 |

Table 8.22: Comparison of results of the standard model and our end-to-end SRL model (reported in F1 scores, the official metrics, for the datasets used).

Results for our ABSA experiments in Czech and English are shown in Tables 8.23 and 8.24, respectively. The *baseline* refers to the model described in Section 8.4.4.2 without any injected SRL information. The SotA results are underlined and the best results for our experiments are bold. We include the results with the 95% confidence interval (experiments repeated 12 times). We use the F1 Micro and accuracy for the CE and CP subtasks, respectively.

Based on the results presented in Tables 8.23 and 8.24, we can observe that our proposed models (*concat-conv* and *concat-avg*) with injected SRL information consistently enhance results for the CP subtask in both languages. The performance of the *concat-conv* and *concat-avg* models does not exhibit a significant difference. In the CE subtask, we achieve the same results as the *baseline* model. We think that the CE subtask is more distant from the SRL

| Model | Category Extraction | | | Category Polarity | |
|---|---|---|---|---|---|
| | F1 Micro | Precision | Recall | Acc #3 | Acc #2 |
| baseline | $86.04^{\pm0.36}$ | $86.48^{\pm0.97}$ | $85.62^{\pm0.65}$ | $75.58^{\pm0.55}$ | $88.69^{\pm0.26}$ |
| concat-conv | **86.58**$^{\pm0.54}$ | **86.90**$^{\pm0.51}$ | **86.28**$^{\pm0.94}$ | **79.20**$^{\pm0.48}$ | **90.26**$^{\pm0.58}$ |
| concat-avg | $86.34^{\pm0.57}$ | $86.57^{\pm0.84}$ | $86.12^{\pm1.08}$ | $78.33^{\pm0.64}$ | $90.06^{\pm0.79}$ |
| multi-task | $85.62^{\pm0.63}$ | $86.24^{\pm0.66}$ | $85.01^{\pm0.66}$ | $77.27^{\pm0.69}$ | $89.00^{\pm0.63}$ |
| baseline (Hercig, Brychcín, Svoboda, Konkol, & Steinberger, 2016)* | 71.70 | - | - | 69.70 | - |
| best (Hercig, Brychcín, Svoboda, Konkol, & Steinberger, 2016)* | 80.00 | - | - | 75.20 | - |
| CNN2 (Lenc & Hercig, 2016) | - | - | - | $69.00^{\pm2.00}$ | - |

Table 8.23: Czech results for the category extraction (CE) subtask as F1 Micro score, Precision and Recall. Results for the category polarity (CP) subtask as accuracy for three polarity labels (Acc #3) and binary polarity labels (Acc #2). Results marked with * symbol were obtained by 10-fold cross-validation.

| Model | Category Extraction | | | Category Polarity | | |
|---|---|---|---|---|---|---|
| | F1 Micro | Precision | Recall | Acc #4 | Acc #3 | Acc #2 |
| baseline | $89.50^{\pm0.45}$ | $90.95^{\pm0.70}$ | $88.09^{\pm0.48}$ | $83.03^{\pm0.43}$ | $86.91^{\pm0.55}$ | $92.74^{\pm0.53}$ |
| concat-conv | **89.74**$^{\pm0.55}$ | **91.24**$^{\pm0.54}$ | **88.28**$^{\pm0.77}$ | **84.19**$^{\pm0.49}$ | **88.08**$^{\pm0.41}$ | **93.76**$^{\pm0.46}$ |
| concat-avg | $89.58^{\pm0.43}$ | $91.15^{\pm0.60}$ | $88.08^{\pm0.66}$ | $84.13^{\pm0.51}$ | $87.95^{\pm0.46}$ | $93.49^{\pm0.44}$ |
| multi-task | $89.36^{\pm0.15}$ | $90.72^{\pm0.52}$ | $88.05^{\pm0.44}$ | $82.83^{\pm1.10}$ | $87.05^{\pm1.21}$ | $92.74^{\pm0.79}$ |
| XRCE (Brun et al., 2014) | 82.29 | 83.23 | 81.37 | 78.10 | - | - |
| NRC (Kiritchenko, Zhu, Cherry, & Mohammad, 2014) | 88.58 | 91.04 | 86.24 | 82.90 | - | - |
| BERT single (Sun, Huang, & Qiu, 2019) | 90.89 | 92.78 | 89.07 | 83.70 | 86.90 | 93.30 |
| NLI-B (Sun, Huang, & Qiu, 2019) | 92.18 | 93.57 | 90.83 | 84.60 | 88.70 | 95.10 |
| QACG-B (Wu & Ong, 2021) | 92.64 | $94.38^{\pm0.31}$ | $90.97^{\pm0.28}$ | $86.80^{\pm0.80}$ | $90.10^{\pm0.30}$ | $95.60^{\pm0.40}$ |
| BART generation (J. Liu et al., 2021) | 92.80 | 95.18 | 90.54 | - | $90.55^{\pm0.32}$ | - |

Table 8.24: English results for the category extraction (CE) subtask as F1 Micro score, Precision and Recall. Results for category polarity (CP) subtask as accuracy for four polarity labels (Acc #4), three polarity labels (Acc #3) and binary polarity labels (Acc #2).

task than the CP subtask and therefore, the injection of the semantic information does not help. In other words, the semantic structure of the sentence may not play a crucial role in aspect detection (that can be viewed as multi-label text classification). On the other hand, for the CP subtask, the combined models can leverage the semantic structure of the sentence to their advantage.

For the Czech ABSA dataset we achieve new SotA results on both subtasks[26]. As we expected, we did not outperform the current SotA results for the English dataset, as our ELECTRA model has considerably fewer parameters than SotA models. For Czech, the *multi-task* model exhibited a marginal improvement in the results and generally, the model was significantly inferior to our other models. We decided to use the smaller ELECTRA-based models because of their much smaller computation requirements.

[26] It is worth noting that although the test data we used differ from those used by Hercig, Brychcín, Svoboda, Konkol, and Steinberger (2016) due to their 10-fold cross-validation, the performance difference is substantial enough to demonstrate the superiority of our approach.

### 8.4.8 Conclusion

In this work, we introduced a novel end-to-end SRL model that improves the aspect category polarity task. Our contribution lies in proposing several methods to integrate SRL and ABSA models, which ultimately lead to improved performance. The experimental results validated our initial assumption that leveraging semantic information extracted from an SRL model can significantly enhance the aspect category polarity task. Importantly, the approaches we proposed are versatile and can be applied to combine Transformer-based models for other related tasks as well, extending the scope of their applicability.

Moreover, we believe that our approaches hold even greater potential in addressing other ABSA subtasks, namely term extraction and term polarity classification. These subtasks could benefit from the integration of SRL and ABSA models in a similar manner.

## 8.5 Prompt-based Approach for Aspect-based Sentiment Analysis

In a paper named “Prompt-Based Approach for Czech Sentiment Analysis” (Šmíd & Přibáň, 2023), we introduced the first prompt-based methods for ABSA and sentiment classification in Czech.

We propose a novel approach for solving Czech sentiment classification and ABSA tasks using the new paradigm called *prompt-based learning* or *prompting*. Nowadays, the traditional approach is to pre-train a Transformer-based model on a large amount of text, for example, BERT (Devlin et al., 2019) and then fine-tune it for a specific task. Prompting is a technique that encourages a pre-trained model to make specific predictions by providing a prompt specifying the task to be done (P. Liu et al., 2023). This new approach became very popular in solving NLP problems in zero-shot or few-shot scenarios, including SA (Gao et al., 2022; Gao et al., 2021; Hosseini-Asl et al., 2022). Most of the current research aimed at languages other than Czech, especially English. To the best of our knowledge, no research has focused on any SA task in the Czech language by using prompting. To address this lack of research, we performed an initial study focusing on two sentiment-related tasks, i.e., ABSA and sentiment classification for the Czech language, by applying prompt-based fine-tuning.

ABSA is a more detailed task compared to sentiment classification, which aims to extract fine-grained information about entities, their aspects and opinions expressed towards them. There are multiple definitions and versions of the ABSA task (Barnes et al., 2022; Pontiki et al., 2014; Saeidi et al., 2016). In this work, we focus on the version of *aspect-based sentiment analysis* presented in the SemEval competitions (Pontiki et al., 2015, 2016), which includes several subtasks. Specifically, the tasks are aspect category detection (ACD), aspect term extraction (ATE), simultaneously detecting (aspect category, aspect term) tuples (ACTE), and detecting the sentiment polarity (APD)[27] of a given aspect term and category (see Figure

[27] The ACD, ATE, ACTE and APD tasks are named Slot1, Slot2, Slot1&2 and Slot3, respectively, in Pontiki et al. (2015, 2016) under Subtask 1.

8.10 for examples). In addition, we solve the target-aspect-sentiment detection task (TASD) (H. Wan et al., 2020), which aims to simultaneously detect the aspect category, aspect term and sentiment polarity.

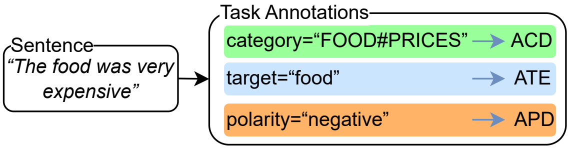

Sentence
"The food was very expensive"
Task Annotations
category="FOOD#PRICES" → ACD
target="food" → ATE
polarity="negative" → APD

Figure 8.10: The example of the ABSA tasks.

Unlike in our previous work (Přibáň & Pražák, 2023) that is described in Section 8.4, in this work we solve slightly different versions of ABSA tasks, thus we cannot compare our results with the work. In addition, we reannotated the entire Czech ABSA dataset from Hercig, Brychcín, Svoboda, Konkol, and Steinberger (2016) to be in the same format as English, Dutch, Russian, Spanish and Turkish datasets from Pontiki et al. (2015), allowing cross-lingual and multilingual experiments between Czech and these languages. The newly reannotated dataset is described in Šmíd et al. (2024)[28].

We utilize Czech monolingual BERT-like models and their language modeling ability to perform *prompting* for the APD and sentiment classification tasks. We use multilingual text-to-text generative models such as mT5 (Xue et al., 2021) and large mBART (Tang et al., 2021) for the remaining ABSA tasks to generate textual predictions based on a prompted input. Our approach enables us to solve all these ABSA tasks at once, and we show its superiority to the traditional fine-tuning approach for them.

We also explore zero-shot and few-shot learning scenarios for APD and SC tasks and show that prompting leads to significantly better results with fewer training examples compared to traditional fine-tuning. Additionally, we demonstrate that pre-training on data from a target domain results in great improvements in a zero-shot scenario.

At the time of publication of the study, it provided pioneered results for prompt-based fine-tuning in Czech sentiment. Overall, the key contributions are the following: 1) to the best of our knowledge, we propose the first prompt-based approach for SA tasks in Czech. 2) We show the superior performance of our prompting approach over traditional fine-tuning for ABSA tasks. 3) We compare the two approaches and show that prompting achieves better results than traditional fine-tuning in few-shot scenarios[29].

[28] The paper was accepted at the LREC-COLING 2024 conference https://lrec-coling-2024.org/ and it will be published in May 2024. The paper is accessible from https://home.zcu.cz/~pribanp/LREC-2024/paper.pdf.

[29] Because the author of this thesis is not the first author of the paper, we do not consider this publication as a core part of this thesis and thus we do not describe the approach and results in detail. For details, see the corresponding paper (Šmíd & Přibáň, 2023)

## 8.6 Emotion Analysis

We focused on emotion analysis tasks in two publications: "UWB at SemEval-2018 Task 1: Emotion Intensity Detection in Tweets" (Přibáň et al., 2018) and "UWB at IEST 2018: Emotion Prediction in Tweets with Bidirectional Long Short-Term Memory Neural Network" (Přibáň & Martínek, 2018).

### 8.6.1 Emotion Intensity

In Přibáň et al. (2018), we presented a system developed for the SemEval-2018 Task 1: Affect in Tweets (Mohammad et al., 2018) competition. This competition was focused on the detection of emotion intensity in posts from Twitter across three languages: Spanish, English, and Arabic. The task was divided into two subtasks. In the emotion intensity regression subtask, the goal was to predict the intensity on a scale of zero to one for a given tweet and emotion. In the second subtask, the intensity was split into four distinct categories.

The task posed notable challenges due to the prevalence of slang expressions, misspelt words, emoticons or abbreviations in tweets. To address this complexity, our system employed a hybrid approach, incorporating traditional language features such as word n-grams, emotion lexicons, and topic distribution obtained through Latent Dirichlet Allocation (LDA) (Blei et al., 2003), alongside word embeddings. For supervised training, we use SVM and logistic regression classifiers. We achieved scores of 0.64, 0.57 and 0.63 of Pearson correlation for English, Arabic and Spanish, respectively, in the intensity regression subtask. In the case of the classification subtask, we achieved scores of 0.50, 0.39 and 0.50 of Pearson correlation for English, Arabic and Spanish, respectively.

### 8.6.2 Emotion Prediction

In Přibáň and Martínek (2018), we proposed a model based on an LSTM neural network to predict an emotion of a given tween from which a certain emotion word is removed. The removed word can be *sad*, *happy*, *disgusted*, *angry*, *afraid* or a synonym of any of these. The model was developed for the WASSA 2018 Implicit Emotion Shared Task (Klinger et al., 2018) competition.

Our approach is based on a neural network that combines word embeddings and emoji-based features as input. The model incorporates BiLSTM layer for word embeddings input and dense layers for the other inputs, i.e., emoji2vec (Eisner et al., 2016) and DeepMoji (Felbo et al., 2017), connected to one dense layer, see the Figure 8.11 with a model architecture. Outputs of these three layers are concatenated and then a dropout (Srivastava et al., 2014) technique is applied. After the concatenating, the next dense layer is employed. An output from the previous dense layer is then passed to a fully connected softmax layer. An output of the softmax layer is a probability distribution over all six possible emotion classes.

Our system system performed best for the joy emotion. It achieved 65.7% macro $F_1$ score and our rank was 13th out of 30 participated teams in the competition.

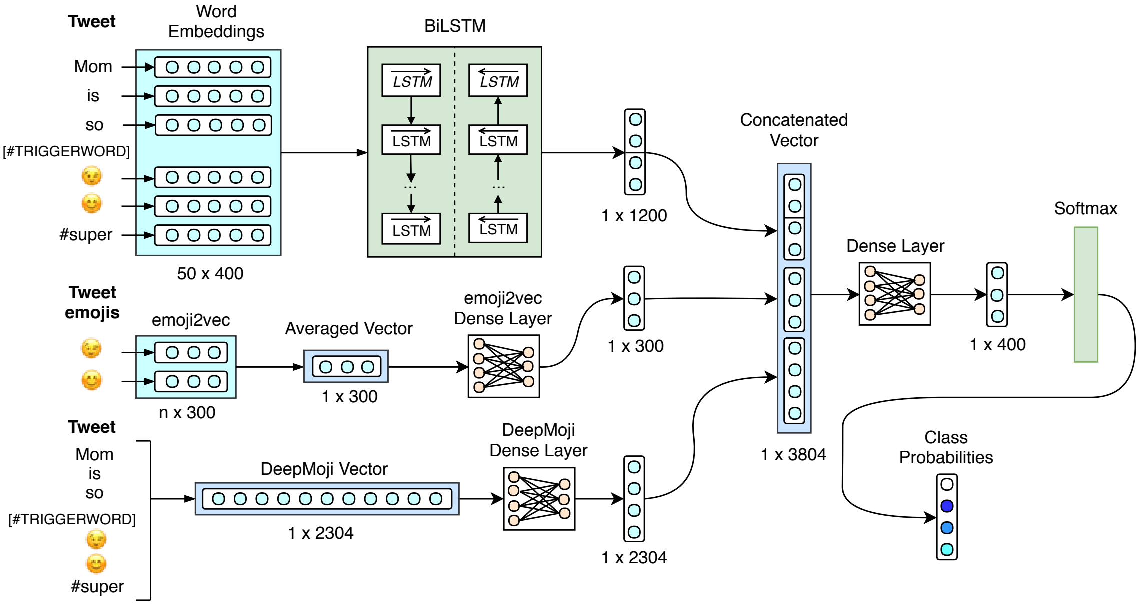


Figure 8.11: Emotion prediction system architecture.

## 8.7 Other Research Contributions

Apart from our contribution to the field of SA, we also published multiple works related to other NLP tasks, including medical classification (Přibáň et al., 2023), named entity recognition (Piskorski et al., 2019, 2021; Yangarber et al., 2023), fact-checking (Přibáň et al., 2019), dialect recognition (Přibáň & Taylor, 2019), lexical semantic change (Pražák, Přibáň, & Taylor, 2020; Pražák, Přibáň, Taylor, & Sido, 2020; Přibáň et al., 2021; S. Taylor et al., 2021) and building pre-trained language models (Sido et al., 2021).

Further, we briefly describe works that we believe are important to the research community or worthy of mentioning even in terms of this thesis.

### 8.7.1 Czech BERT-like Model

In our work presented in Sido et al. (2021), we introduced the first BERT-like model tailored for the Czech language. This contribution outlines the process of training two distinct BERT-like models for the Czech language and evaluates their performance across six tasks, comparing them to two existing multilingual models, namely mBERT (Devlin et al., 2019) and SlavicBERT (Arkhipov et al., 2019). The models, named Czert-A and Czert-B, are publicly available[30]. More concretely, the architectures of our models are based on the ALBERT (Lan et al., 2020) model (Czert-A) and the original BERT (Devlin et al., 2019) model (Czert-B). Both models are trained from scratch, utilizing a text corpus of approximately 36 GB of plain text,

[30] The model is available at https://github.com/kiv-air/Czert

comprising Czech Wikipedia articles, crawled Czech news, and the Czech National Corpus (Křen et al., 2016).

We trained the models from scratch (i.e., with random initialization) using *Masked Language Model* (MLM) and *Next Sentence Prediction* (NSP) tasks as training objectives with a slight modification of the NSP task. We evaluated our models on six tasks: Semantic Text Similarity, Named Entity Recognition, Morphological Tagging, Semantic Role Labeling, Sentiment Classification and Multi-label Document Classification. Our models outperformed the multilingual counterparts on 9 out of 11 datasets. In addition, at the time of releasing the models, we established the new state-of-the-art results on nine datasets, including sentiment classification.

### 8.7.2 Lexical Semantic Change

In Pražák, Přibáň, Taylor, and Sido (2020), we proposed a model for the task of lexical semantic change in English, German, Latin and Swedish. Our approach is based on the same linear transformations (described in Section6.2.1) as we used for our cross-lingual sentiment classification experiments presented in Chapter 7. The model was designed for the SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection (Schlechtweg et al., 2020) competition, in which we ranked first and won the competition.

The lexical semantic change task aims to identify shifts in the meanings of words over time. Typically, two corpora from distinct time periods are provided for analysis. Our method is fully unsupervised and language-independent. It consists of preparing a semantic vector space for each corpus, earlier and later; computing a linear transformation between earlier and later spaces, using Canonical Correlation Analysis and Orthogonal Transformation; and measuring the cosines between the transformed vector for the target word from the earlier corpus and the vector for the target word in the later corpus. The competition was divided into two subtasks. We ranked $1^{st}$ in Sub-task 1: binary change detection, and $4^{th}$ in Sub-task 2: ranked change detection.

We further successfully applied our approach (S. Taylor et al., 2021) in other languages such as Italian (Pražák, Přibáň, & Taylor, 2020) and Russian (Přibáň et al., 2021).

### 8.7.3 Dataset for Multilingual Named Entity Recognition

In terms of building NLP resources, we contributed by building a manually annotated multilingual corpus (Piskorski et al., 2019, 2021; Yangarber et al., 2023) for the tasks of *named entity recognition and classification, name normalization* and *cross-lingual entity linking* in Czech, Polish, Bulgarian, Russian, Slovene and Ukrainian.

The goal of *named entity recognition and classification* task is to recognize all named mentions of five types: persons, organizations, locations, products and events. In *name normalization*, the goal is to convert the detected named mentions to their base forms, usually its lemma. Finally, in the *entity linking* task, all mentions of the detected named entity should be linked across all its mentions, including mentions in other languages.

The dataset contains around 4.1k documents that come mainly from news articles which cover multiple topics like *Brexit, Nord Stream, Covid-19, USAA 2020 elections, Russian invasion of Ukraine*. All documents are manually annotated for each of the three named tasks. See Yangarber et al. (2023) for detailed statistics of language, topic and class distribution.

# Contributions Summary 9

This chapter summarizes our contributions and the fulfilment of the objectives of this thesis. In Section 9.1, we shortly outline our published works and their respective contributions. Section 9.2 provides details on the fulfilment of the defined goals of this thesis.

## 9.1 Contributions Overview

This section concisely outlines the primary contributions stemming from our publications, focusing on their relevance to sentiment analysis and the central theme of this thesis. Further, we also provide an overview of other additional research contributions.

### Contributions:

- Conducting preliminary experiments, we explored cross-lingual sentiment classification between Czech and English utilizing Transformer-based models. Furthermore, we applied monolingual models to Czech sentiment classification and achieved new SotA results (Přibáň & Steinberger, 2021).

- We proposed methods for cross-lingual sentiment classification between Czech, French and English based on linear transformations and compared their competitive performance with Transformer-based models (Přibáň et al., 2022).

- We thoroughly evaluated methods for cross-lingual sentiment classification, including the latest LLMs (ChatGPT and LLama 2), linear transformations and Transformer-based approaches. Our extensive comparison shed light on their advantages and disadvantages, considering factors such as performance, training/inference speed, and their applicability in real-world scenarios (Přibáň et al., 2024).

- We created a new Czech dataset of 10k manually labeled sentences for subjectivity classification. Subsequently, we conducted cross-lingual experiments, specifically between Czech and English, utilizing this new dataset (Přibáň & Steinberger, 2022).

- We thoroughly evaluated multilingual systems deployed in a real-world production environment, as discussed in Přibáň and Balahur (2023).

- We proposed a novel multitask approach that improves the performance of the aspect-based sentiment analysis by leveraging information from the semantic role labeling task and achieved new SotA results for Czech (Přibáň & Pražák, 2023).

- We applied prompt-based learning to the aspect-based sentiment analysis and sentiment classification tasks for the Czech language (Šmíd & Přibáň, 2023).

- We completely reannotated the existing Czech dataset for ABSA to the same format as its counterparts in other languages. As a result, the dataset can be utilized in cross-lingual experiments. The annotation process is described in (Šmíd et al., 2024).

- We proposed and built models for emotion intensity detection and emotion prediction tasks (Přibáň & Martínek, 2018; Přibáň et al., 2018).

**Additional Research Contributions:**

- We created the first BERT-like model for the Czech language (Sido et al., 2021).

- We proposed a novel approach based on linear transformations for the task of lexical semantic change and we won the SemEval competition (SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection) (Pražák, Přibáň, & Taylor, 2020; Pražák, Přibáň, Taylor, & Sido, 2020; Přibáň et al., 2021; S. Taylor et al., 2021).

- We built a manually annotated multilingual dataset for the tasks of named entity recognition and classification, name normalization and cross-lingual entity linking in Czech, Polish, Bulgarian, Russian, Slovene and Ukrainian (Piskorski et al., 2019, 2021; Yangarber et al., 2023).

- We built a system for automatical clinical document classification (coding) (Přibáň et al., 2023)

- We explored the task of Arabic dialect recognition (Přibáň & Taylor, 2019).

- We compiled a new Czech, Polish and Slovak dataset for the fact-checking task (Přibáň et al., 2019).

## 9.2 Fulfilment of the Thesis Goals

Based on the predefined objectives for this thesis that were set and defined in the author's Ph.D. thesis exposé document (Přibáň, 2020), we present a concise overview of how these goals were successfully met in this thesis. Corresponding publications are also referenced for a comprehensive understanding of the achievements.

1. **Tackle the problem of lack of annotated data in languages other than English by introducing new resources.**
   This goal is notably achieved by creating a novel manually annotated dataset tailored for subjectivity classification in Czech (Přibáň & Steinberger, 2022). Prior to this effort, there was a conspicuous absence of a dedicated dataset in the Czech language for this specific task. The construction of this dataset was approached with a keen consideration for its applicability in cross-lingual experiments. As a direct result, it is a valuable resource for conducting cross-lingual benchmarks between English and Czech.
   Next, in Šmíd and Přibáň (2023), we completely reannotated the existing Czech dataset for ABSA to the same format as its counterparts in other languages. Consequently, the dataset can be used for cross-lingual experiments between Czech and several other languages, including English. We describe the annotation process in Šmíd et al. (2024)[1].
   Additionally to this goal, we also built new multilingual resources for NER (Piskorski et al., 2019, 2021; Yangarber et al., 2023) and fact-checking (Přibáň et al., 2019) tasks.

2. **Perform sentiment analysis (and other related) tasks in languages other than English by applying cross-lingual methods and transforming knowledge between languages.**
   We perform zero-shot cross-lingual sentiment analysis between English, Czech and French by using linear transformations that allow external transfer of knowledge between the languages in Přibáň et al. (2022). In our subsequent investigation, detailed in Přibáň et al. (2024), we study the usage of linear transformations for CLSA deeper and in more detail, including their performance and speed comparison with the most recent approaches, such as LLMs. Notably, our findings revealed that the linear transformation-based approach exhibits performance levels comparable to smaller Transformer-based models while significantly outpacing them in terms of speed.
   The fulfilment of this goal is also partly supported in Přibáň and Balahur (2023), where we compared multilingual systems for a real-world application.

3. **Apply recent state-of-the-art pre-trained models and transfer learning approaches to sentiment analysis (and other related) tasks to textual data other than English.**
   First, we fulfil this goal in Přibáň and Steinberger (2021), where we delved into the CLSA by leveraging recent multilingual Transformer-based models in the context of Czech and English. Building upon this exploration, our subsequent work in Přibáň et al. (2024), expands the scope to include the French language. We employed the most recent LLMs such as Llama 2 and ChatGPT. These recent LLMs use very little or no training data and achieve results that are on par or better than the multilingual Transformer-based models but with significant additional hardware requirements and

[1] The paper is accepted and it is going to be published in proceedings of the LREC-COLING 2024 conference https://lrec-coling-2024.org/. The paper is accessible from https://home.zcu.cz/~pribanp/LREC-2024/paper.pdf.

limitations associated with these LLMs.

In the same work, we provide a comprehensive overview of current approaches to CLSA, offering an exhaustive evaluation of selected methods and a discussion of their merits and drawbacks in terms of performance and training speeds.

Additionally, we performed cross-lingual experiments on our newly built subjectivity dataset (Přibáň & Steinberger, 2022) with multilingual Transformer-based models.

# Author’s Publications

## Journal Publications

1. Přibáň, P., Šmíd, J., Steinberger, J., & Mištera, A. (2024). A comparative study of cross-lingual sentiment analysis. *Expert Systems with Applications*, *247*, 123247. https://doi.org/https://doi.org/10.1016/j.eswa.2024.123247

2. Taylor, S., Přibáň, P., & Prazák, O. (2021). Comparewords: Measuring semantic change in word usage in different corpora. *Softw. Impacts*, *8*, 100067. https://doi.org/10.1016/J.SIMPA.2021.100067

## Conference Publications

1. Přibáň, P., & Steinberger, J. (2021). Are the multilingual models better? improving Czech sentiment with transformers. *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, 1138–1149. https://aclanthology.org/2021.ranlp-1.128

2. Přibáň, P., Šmíd, J., Mištera, A., & Král, P. (2022). Linear transformations for cross-lingual sentiment analysis. In P. Sojka, A. Horák, I. Kopeček, & K. Pala (Eds.), *Text, speech, and dialogue* (pp. 125–137). Springer International Publishing

3. Přibáň, P., & Steinberger, J. (2022). Czech dataset for cross-lingual subjectivity classification. *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, 1381–1391. https://aclanthology.org/2022.lrec-1.148

4. Přibáň, P., & Balahur, A. (2023). Comparative analyses of multilingual sentiment analysis systems for news and social media. In A. Gelbukh (Ed.), *Computational linguistics and intelligent text processing* (pp. 260–279). Springer Nature Switzerland

5. Přibáň, P., & Pražák, O. (2023, September). Improving aspect-based sentiment with end-to-end semantic role labeling model. In R. Mitkov & G. Angelova (Eds.), *Proceedings of the 14th international conference on recent advances in natural language processing* (pp. 888–897). INCOMA Ltd., Shoumen, Bulgaria. https://aclanthology.org/2023.ranlp-1.96

6. Šmíd, J., & Přibáň, P. (2023, September). Prompt-based approach for Czech sentiment analysis. In R. Mitkov & G. Angelova (Eds.), *Proceedings of the 14th international conference on recent advances in natural language processing* (pp. 1110–1120). INCOMA Ltd., Shoumen, Bulgaria. https://aclanthology.org/2023.ranlp-1.118

7. Šmíd, J., Přibáň, P., & Pražák, O. (2024). Czech dataset for complex aspect-based sentiment analysis tasks. *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation*. https://home.zcu.cz/~pribanp/LREC-2024/paper.pdf

8. Přibáň, P., Hercig, T., & Lenc, L. (2018). UWB at SemEval-2018 task 1: Emotion intensity detection in tweets. *Proceedings of the 12th International Workshop on Semantic Evaluation*, 133–140. https://doi.org/10.18653/v1/S18-1018

9. Přibáň, P., & Martínek, J. (2018). UWB at IEST 2018: Emotion prediction in tweets with bidirectional long short-term memory neural network. *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 224–230. https://doi.org/10.18653/v1/W18-6232

10. Sido, J., Pražák, O., Přibáň, P., Pašek, J., Seják, M., & Konopík, M. (2021). Czert – Czech BERT-like model for language representation. *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, 1326–1338. https://aclanthology.org/2021.ranlp-1.149

11. Pražák, O., Přibáň, P., Taylor, S., & Sido, J. (2020, December). UWB at SemEval-2020 task 1: Lexical semantic change detection. In A. Herbelot, X. Zhu, A. Palmer, N. Schneider, J. May, & E. Shutova (Eds.), *Proceedings of the fourteenth workshop on semantic evaluation* (pp. 246–254). International Committee for Computational Linguistics. https://doi.org/10.18653/v1/2020.semeval-1.30

12. Pražák, O., Přibáň, P., & Taylor, S. (2020). UWB @ DIACR-Ita: Lexical semantic change detection with CCA and orthogonal transformation. In V. Basile, D. Croce, M. D. Maro, & L. C. Passaro (Eds.), *Proceedings of the seventh evaluation campaign of natural language processing and speech tools for italian. final workshop (EVALITA 2020), online event, december 17th, 2020* (Vol. 2765). CEUR-WS.org. http://ceur-ws.org/Vol-2765/paper110.pdf

13. Přibáň, P., Ondřej, P., & Stephen, T. (2021). Uwb@ rushifteval: Measuring semantic difference as per-word variation in aligned semantic spaces. *Komp'yuternaya Lingvistika i Intellektual'nye Tekhnologii: Dialogue conference.—-2021*. https://doi.org/10.28995/2075-7182-2021-20-1188-1196

14. Piskorski, J., Laskova, L., Marcińczuk, M., Pivovarova, L., Přibáň, P., Steinberger, J., & Yangarber, R. (2019, August). The second cross-lingual challenge on recognition,

normalization, classification, and linking of named entities across Slavic languages. In T. Erjavec, M. Marcińczuk, P. Nakov, J. Piskorski, L. Pivovarova, J. Šnajder, J. Steinberger, & R. Yangarber (Eds.), *Proceedings of the 7th workshop on balto-slavic natural language processing* (pp. 63–74). Association for Computational Linguistics. https://doi.org/10.18653/v1/W19-3709

15. Piskorski, J., Babych, B., Kancheva, Z., Kanishcheva, O., Lebedeva, M., Marcińczuk, M., Nakov, P., Osenova, P., Pivovarova, L., Pollak, S., Přibáň, P., Radev, I., Robnik-Sikonja, M., Starko, V., Steinberger, J., & Yangarber, R. (2021, April). Slav-NER: The 3rd cross-lingual challenge on recognition, normalization, classification, and linking of named entities across Slavic languages. In B. Babych, O. Kanishcheva, P. Nakov, J. Piskorski, L. Pivovarova, V. Starko, J. Steinberger, R. Yangarber, M. Marcińczuk, S. Pollak, P. Přibáň, & M. Robnik-Šikonja (Eds.), *Proceedings of the 8th workshop on balto-slavic natural language processing* (pp. 122–133). Association for Computational Linguistics. https://aclanthology.org/2021.bsnlp-1.15

16. Yangarber, R., Piskorski, J., Dmitrieva, A., Marcińczuk, M., Přibáň, P., Rybak, P., & Steinberger, J. (2023, May). Slav-NER: The 4th cross-lingual challenge on recognition, normalization, classification, and linking of named entities across Slavic languages. In J. Piskorski, M. Marcińczuk, P. Nakov, M. Ogrodniczuk, S. Pollak, P. Přibáň, P. Rybak, J. Steinberger, & R. Yangarber (Eds.), *Proceedings of the 9th workshop on slavic natural language processing 2023 (slavicnlp 2023)* (pp. 179–189). Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.bsnlp-1.21

17. Přibáň, P., Baloun, J., Martínek, J., Lenc, L., Prantl, M., & Král, P. (2023). Towards automatic medical report classification in czech. *ICAART (3)*, 228–233

18. Přibáň, P., & Taylor, S. (2019, August). ZCU-NLP at MADAR 2019: Recognizing Arabic dialects. In W. El-Hajj, L. H. Belguith, F. Bougares, W. Magdy, I. Zitouni, N. Tomeh, M. El-Haj, & W. Zaghouani (Eds.), *Proceedings of the fourth arabic natural language processing workshop* (pp. 208–213). Association for Computational Linguistics. https://doi.org/10.18653/v1/W19-4623

19. Přibáň, P., Hercig, T., & Steinberger, J. (2019, September). Machine learning approach to fact-checking in West Slavic languages. In R. Mitkov & G. Angelova (Eds.), *Proceedings of the international conference on recent advances in natural language processing (ranlp 2019)* (pp. 973–979). INCOMA Ltd. https://doi.org/10.26615/978-954-452-056-4_113

# List of Abbreviations

| | |
|---|---|
| **ABSA** | aspect-based sentiment analysis |
| **AI** | artificial intelligence |
| **API** | application programming interface |
| **BiLSTM** | bidirectional long short-term memory |
| **BiRNN** | bidirectional recurrent neural network |
| **BPE** | byte-pair encoding |
| **BWE** | bilingual word embeddings |
| **CBOW** | continuous bag-of-words |
| **CCA** | canonical correlation analysis |
| **CLS** | classification token |
| **CLSA** | cross-lingual sentiment analysis |
| **CLWE** | cross-lingual word embeddings |
| **CNN** | convolutional neural network |
| **CRF** | conditional random fields |
| **CSFD** | czech-slovak film database |
| **FB** | facebook |
| **FFN** | feed-forward layer |
| **GPT** | generative pre-trained transformer |
| **GRU** | gated recurrent unit |
| **HMM** | hidden markov models |

| | |
|---|---|
| **IEST** | implicit emotion shared task |
| **IMDB** | internet movie database |
| **LLM** | large language model |
| **LLMs** | large language models |
| **LM** | language model, language modeling |
| **LSA** | latent semantic analysis |
| **LSTM** | long short-term memory |
| **MLM** | masked language modeling |
| **MLP** | multilayer perceptron |
| **MSE** | mean squared error |
| **NB** | naive bayes |
| **NLP** | natural language processing |
| **NLTK** | natural language toolkit |
| **NNLM** | neural network for language modeling |
| **NSP** | next sentence prediction |
| **OOV** | out-of-vocabulary |
| **POS** | part-of-speech |
| **ReLU** | rectified linear unit |
| **RNN** | recurrent neural network |
| **SA** | sentiment analysis |
| **SC** | sentiment classification |
| **SemEval** | international workshop on semantic evaluation |
| **SotA** | state-of-the-art |
| **SST** | stanford sentiment treebank |
| **SVD** | singular value decomposition |
| **SVM** | support vector machines |

| | |
|---|---|
| **VAD** | valence arousal dominance |
| **WASSA** | workshop on computational approaches to subjectivity, sentiment & social media analysis |

# Declaration of Generative AI and AI-assisted Technologies in the Writing Process

During the preparation of this thesis, we used the ChatGPT[2] and Grammarly[3] tools in order to improve readability and language and correct grammatical errors. After using these tools, we reviewed and edited the content as needed and took full responsibility for the content of the thesis. The tools were not in any case used to analyse and draw insights from data as part of the research process.

[2] https://chat.openai.com

[3] https://app.grammarly.com

# Bibliography

Abdalla, M., & Hirst, G. (2017). Cross-lingual sentiment analysis without (good) translation. *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 506–515. https://www.aclweb.org/anthology/I17-1051

Abdul-Mageed, M., & Ungar, L. (2017). EmoNet: Fine-grained emotion detection with gated recurrent neural networks. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 718–728. https://doi.org/10.18653/v1/P17-1067

Adams, O., Makarucha, A., Neubig, G., Bird, S., & Cohn, T. (2017). Cross-lingual word embeddings for low-resource language modeling. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 937–947. https://aclanthology.org/E17-1088

Agrawal, P., & Suri, A. (2019). NELEC at SemEval-2019 task 3: Think twice before going deep. *Proceedings of the 13th International Workshop on Semantic Evaluation*, 266–271. https://doi.org/10.18653/v1/S19-2045

Agüero-Torales, M. M., Abreu Salas, J. I., & López-Herrera, A. G. (2021). Deep learning and multilingual sentiment analysis on social media data: An overview. *Applied Soft Computing*, *107*, 107373. https://doi.org/https://doi.org/10.1016/j.asoc.2021.107373

Alhuzali, H., & Ananiadou, S. (2021). SpanEmo: Casting multi-label emotion classification as span-prediction. *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 1573–1584. https://doi.org/10.18653/v1/2021.eacl-main.135

Aliramezani, M., Doostmohammadi, E., Bokaei, M. H., & Sameti, H. (2020). Persian sentiment analysis without training data using cross-lingual word embeddings. *2020 10th International Symposium onTelecommunications (IST)*, 78–82. https://doi.org/10.1109/IST50524.2020.9345882

Ammar, W., Mulcaire, G., Tsvetkov, Y., Lample, G., Dyer, C., & Smith, N. A. (2016). Massively multilingual word embeddings. *ArXiv*, *abs/1602.01925*.

Amplayo, R. K., Lee, K., Yeo, J., & Hwang, S.-W. (2018). Translations as additional contexts for sentence classification. *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 3955–3961.

Anil, R., Dai, A. M., Firat, O., Johnson, M., Lepikhin, D., Passos, A., Shakeri, S., Taropa, E., Bailey, P., Chen, Z., et al. (2023). Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

Arkhipov, M., Trofimova, M., Kuratov, Y., & Sorokin, A. (2019). Tuning multilingual transformers for language-specific named entity recognition. *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*, 89–93. https://doi.org/10.18653/v1/W19-3712

Artetxe, M., Labaka, G., & Agirre, E. (2016). Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2289–2294. https://doi.org/10.18653/v1/D16-1250

Artetxe, M., Labaka, G., & Agirre, E. (2018). A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 789–798. https://doi.org/10.18653/v1/P18-1073

Ba, L. J., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. http://arxiv.org/abs/1607.06450

Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Y. Bengio & Y. LeCun (Eds.), *3rd international conference on learning representations, ICLR 2015, san diego, ca, usa, may 7-9, 2015, conference track proceedings*. http://arxiv.org/abs/1409.0473

Baker, C. F., Fillmore, C. J., & Lowe, J. B. (1998). The Berkeley FrameNet project. *Proceedings of the 17th International Conference on Computational Linguistics - Volume 1*, 86–90. https://doi.org/10.3115/980451.980860

Balabantaray, R. C., Mohammad, M., & Sharma, N. (2012). Multi-class twitter emotion classification: A new approach. *International Journal of Applied Information Systems*, *4*(1), 48–53.

Balahur, A., Hermida, J. M., & Montoyo, A. (2012). Detecting implicit expressions of emotion in text: A comparative analysis. *Decision Support Systems*, *53*(4), 742–753.

Balahur, A., & Turchi, M. (2012). Multilingual sentiment analysis using machine translation? *Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis*, 52–60. https://www.aclweb.org/anthology/W12-3709

Balahur, A., & Turchi, M. (2014). Comparative experiments using supervised learning and machine translation for multilingual sentiment analysis. *Computer Speech & Language*, *28*(1).

Balahur, A., Turchi, M., Steinberger, R., Ortega, J. M. P., Jacquet, G., Küçük, D., Zavarella, V., & El Ghali, A. (2014). Resource creation and evaluation for multilingual sentiment analysis in social media texts. *LREC*, 4265–4269.

Barbieri, F., Basile, V., Croce, D., Nissim, M., Novielli, N., & Patti, V. (2016). Overview of the evalita 2016 sentiment polarity classification task. In P. Basile, A. Corazza, F. Cutugno, S. Montemagni, M. Nissim, V. Patti, G. Semeraro, & R. Sprugnoli (Eds.), *Proceedings of third italian conference on computational linguistics (clic-it 2016) & fifth evaluation campaign of natural language processing and speech tools for italian. final* *workshop (EVALITA 2016), napoli, italy, december 5-7, 2016* (Vol. 1749). CEUR-WS.org. http://ceur-ws.org/Vol-1749/paper%5C_026.pdf

Barbosa, L., & Feng, J. (2010). Robust sentiment detection on twitter from biased and noisy data. *Coling 2010: Posters*, 36–44. https://www.aclweb.org/anthology/C10-2005

Barnes, J., Klinger, R., & Schulte im Walde, S. (2018). Bilingual sentiment embeddings: Joint projection of sentiment across languages. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2483–2493. https://doi.org/10.18653/v1/P18-1231

Barnes, J., Lambert, P., & Badia, T. (2016). Exploring distributional representations and machine translation for aspect-based cross-lingual sentiment classification. *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 1613–1623. https://aclanthology.org/C16-1152

Barnes, J., Oberlaender, L., Troiano, E., Kutuzov, A., Buchmann, J., Agerri, R., Øvrelid, L., & Velldal, E. (2022). SemEval 2022 task 10: Structured sentiment analysis. *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, 1280–1295. https://doi.org/10.18653/v1/2022.semeval-1.180

Barriere, V., & Balahur, A. (2020). Improving sentiment analysis over non-English tweets using multilingual transformers and automatic translation for data-augmentation. *Proceedings of the 28th International Conference on Computational Linguistics*, 266–271. https://doi.org/10.18653/v1/2020.coling-main.23

Baziotis, C., Nikolaos, A., Chronopoulou, A., Kolovou, A., Paraskevopoulos, G., Ellinas, N., Narayanan, S., & Potamianos, A. (2018). NTUA-SLP at SemEval-2018 task 1: Predicting affective content in tweets with deep attentive RNNs and transfer learning. *Proceedings of The 12th International Workshop on Semantic Evaluation*, 245–255. https://doi.org/10.18653/v1/S18-1037

Baziotis, C., Pelekis, N., & Doulkeridis, C. (2017). DataStories at SemEval-2017 task 4: Deep LSTM with attention for message-level and topic-based sentiment analysis. *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 747–754. https://doi.org/10.18653/v1/S17-2126

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, *3*(Feb), 1137–1155.

Bird, E. L., Steven, & Klein, E. (2009). *Natural language processing with python*. O'Reilly Media Inc.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993–1022.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135–146. https://doi.org/10.1162/tacl_a_00051

Bostan, L.-A.-M., & Klinger, R. (2018). An analysis of annotated corpora for emotion classification in text. *Proceedings of the 27th International Conference on Computational Linguistics*, 2104–2119. https://www.aclweb.org/anthology/C18-1179

Bragg, J., Cohan, A., Lo, K., & Beltagy, I. (2021). Flex: Unifying evaluation for few-shot nlp. *Advances in Neural Information Processing Systems*, *34*.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, *33*, 1877–1901.

Brun, C., Popa, D. N., & Roux, C. (2014, August). XRCE: Hybrid classification for aspect-based sentiment analysis. In P. Nakov & T. Zesch (Eds.), *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)* (pp. 838–842). Association for Computational Linguistics. https://doi.org/10.3115/v1/S14-2149

Brychcín, T. (2020). Linear transformations for cross-lingual semantic textual similarity. *Knowledge - Based Systems*, *187*, 104819. https://doi.org/https://doi.org/10.1016/j.knosys.2019.06.027

Brychcín, T., Taylor, S. E., & Svoboda, L. (2019). Cross-lingual word analogies using linear transformations between semantic spaces. *Expert Systems with Applications*, *135*.

Brychcín, T., & Habernal, I. (2013). Unsupervised improving of sentiment analysis using global target context. *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, 122–128. https://www.aclweb.org/anthology/R13-1016

Buechel, S., & Hahn, U. (2016). Emotion analysis as a regression problem — dimensional models and their implications on emotion representation and metrical evaluation. *Proceedings of the Twenty-Second European Conference on Artificial Intelligence*, 1114–1122. https://doi.org/10.3233/978-1-61499-672-9-1114

Buechel, S., & Hahn, U. (2017). EmoBank: Studying the impact of annotation perspective and representation format on dimensional emotion analysis. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 578–585. https://www.aclweb.org/anthology/E17-2092

Can, E. F., Ezen-Can, A., & Can, F. (2018). Multilingual sentiment analysis: An rnn-based framework for limited data. *CoRR*, *abs/1806.04511*.

Canales, L., & Martínez-Barco, P. (2014). Emotion detection from text: A survey. *Proceedings of the Workshop on Natural Language Processing in the 5th Information Systems Research Working Days (JISIC)*, 37–43. https://doi.org/10.3115/v1/W14-6905

Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al. (2021). Extracting training data from large language models. *30th USENIX Security Symposium (USENIX Security 21)*, 2633–2650.

Catelli, R., Bevilacqua, L., Mariniello, N., Scotto di Carlo, V., Magaldi, M., Fujita, H., De Pietro, G., & Esposito, M. (2022). Cross lingual transfer learning for sentiment analysis of italian tripadvisor reviews. *Expert Systems with Applications*, *209*, 118246. https://doi.org/https://doi.org/10.1016/j.eswa.2022.118246

Cer, D., Yang, Y., Kong, S.-y., Hua, N., Limtiaco, N., St. John, R., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Strope, B., & Kurzweil, R. (2018). Universal sentence encoder for English. *Proceedings of the 2018 Conference on Empirical Methods in Nat-*

*ural Language Processing: System Demonstrations*, 169–174. https://doi.org/10.18653/v1/D18-2029

Chan, B., Schweter, S., & Möller, T. (2020). German's next language model. *Proceedings of the 28th International Conference on Computational Linguistics*, 6788–6796. https://doi.org/10.18653/v1/2020.coling-main.598

Chatterjee, A., Narahari, K. N., Joshi, M., & Agrawal, P. (2019). SemEval-2019 task 3: EmoContext contextual emotion detection in text. *Proceedings of the 13th International Workshop on Semantic Evaluation*, 39–48. https://doi.org/10.18653/v1/S19-2005

Chen, P., Sun, Z., Bing, L., & Yang, W. (2017). Recurrent attention network on memory for aspect sentiment analysis. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 452–461. https://doi.org/10.18653/v1/D17-1047

Chen, Q., Zhang, R., Zheng, Y., & Mao, Y. (2022). Dual contrastive learning: Text classification via label-aware data augmentation. *arXiv preprint*.

Chen, X., Sun, Y., Athiwaratkun, B., Cardie, C., & Weinberger, K. (2018). Adversarial deep averaging networks for cross-lingual sentiment classification. *Transactions of the Association for Computational Linguistics*, *6*, 557–570. https://doi.org/10.1162/tacl_a_00039

Cho, K., van Merriënboer, B., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder–decoder approaches. *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, 103–111. https://doi.org/10.3115/v1/W14-4012

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1724–1734. https://doi.org/10.3115/v1/D14-1179

Choi, Y., & Cardie, C. (2010). Hierarchical sequential learning for extracting opinions and their attributes. *Proceedings of the ACL 2010 Conference Short Papers*, 269–274. https://www.aclweb.org/anthology/P10-2050

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2015). Gated feedback recurrent neural networks. *International conference on machine learning*, 2067–2075.

Clark, K., Luong, M.-T., Le, Q. V., & Manning, C. D. (2020). ELECTRA: Pre-training text encoders as discriminators rather than generators. *ICLR*. https://openreview.net/pdf?id=r1xMH1BtvB

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement*, *20*(1), 37–46.

Commons, W. (2020). File:plutchik-wheel.svg — wikimedia commons, the free media repository [[Online; accessed 18-March-2020]]. https://commons.wikimedia.org/w/index.php?title=File:Plutchik-wheel.svg&oldid=386724898

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2020). Unsupervised cross-lingual representation learning at scale. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 8440–8451. https://doi.org/10.18653/v1/2020.acl-main.747

Conneau, A., & Lample, G. (2019). Cross-lingual language model pretraining. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 32). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2019/file/c04c19c2c2474dbf5f7ac4372c5b9af1-Paper.pdf

Dashtipour, K., Poria, S., Hussain, A., Cambria, E., Hawalah, A. Y., Gelbukh, A., & Zhou, Q. (2016). Multilingual sentiment analysis: State of the art and independent comparison of techniques. *Cognitive computation*, *8*(4), 757–771.

Davidov, D., Tsur, O., & Rappoport, A. (2010). Semi-supervised recognition of sarcasm in twitter and Amazon. *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, 107–116. https://www.aclweb.org/anthology/W10-2914

Delobelle, P., Winters, T., & Berendt, B. (2020). RobBERT: A Dutch RoBERTa-based Language Model. *Findings of the Association for Computational Linguistics: EMNLP 2020*, 3255–3265. https://doi.org/10.18653/v1/2020.findings-emnlp.292

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186. https://doi.org/10.18653/v1/N19-1423

de Vries, W., van Cranenburgh, A., Bisazza, A., Caselli, T., Noord, G. v., & Nissim, M. (2019, December). BERTje: A Dutch BERT Model. http://arxiv.org/abs/1912.09582

Dong, L., Wei, F., Tan, C., Tang, D., Zhou, M., & Xu, K. (2014). Adaptive recursive neural network for target-dependent Twitter sentiment classification. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 49–54. https://doi.org/10.3115/v1/P14-2009

Dong, X., & de Melo, G. (2018). Cross-lingual propagation for deep sentiment analysis. *Proceedings of the AAAI Conference on Artificial Intelligence*, *32*(1). https://doi.org/10.1609/aaai.v32i1.12071

Dumitrescu, S., Avram, A.-M., & Pyysalo, S. (2020). The birth of Romanian BERT. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, 4324–4328. https://www.aclweb.org/anthology/2020.findings-emnlp.387

Duppada, V., Jain, R., & Hiray, S. (2018). SeerNet at SemEval-2018 task 1: Domain adaptation for affect in tweets. *Proceedings of The 12th International Workshop on Semantic Evaluation*, 18–23. https://doi.org/10.18653/v1/S18-1002

Eisner, B., Rocktäschel, T., Augenstein, I., Bošnjak, M., & Riedel, S. (2016, November). Emoji 2 vec: Learning emoji representations from their description. In L.-W. Ku, J. Y.-j. Hsu, & C.-T. Li (Eds.), *Proceedings of the fourth international workshop on natural language* *processing for social media* (pp. 48–54). Association for Computational Linguistics. https://doi.org/10.18653/v1/W16-6208

Ekman, P. (1992). An argument for basic emotions. *Cognition & emotion*, *6*(3-4), 169–200.

Elman, J. L. (1990). Finding structure in time. *Cognitive science*, *14*(2), 179–211.

Eriguchi, A., Johnson, M., Firat, O., Kazawa, H., & Macherey, W. (2018). Zero-shot cross-lingual classification using multilingual neural machine translation. *arXiv preprint arXiv : 1809.04686*.

Farra, N. (2019). *Cross-lingual and low-resource sentiment analysis*. Columbia University.

Faruqui, M., & Dyer, C. (2014). Improving vector space word representations using multilingual correlation. *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, 462–471. https://doi.org/10.3115/v1/E14-1049

Fei, H., Li, B., Liu, Q., Bing, L., Li, F., & Chua, T.-S. (2023). Reasoning implicit sentiment with chain-of-thought prompting. *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 1171–1182. https://doi.org/10.18653/v1/2023.acl-short.101

Felbo, B., Mislove, A., Søgaard, A., Rahwan, I., & Lehmann, S. (2017, September). Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In M. Palmer, R. Hwa, & S. Riedel (Eds.), *Proceedings of the 2017 conference on empirical methods in natural language processing* (pp. 1615–1625). Association for Computational Linguistics. https://doi.org/10.18653/v1/D17-1169

Feldman, R. (2013). Techniques and applications for sentiment analysis. *Communications of the ACM*, *56*(4), 82–89.

Firth, J. R. (1957). A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*.

Frijda, N. H. (1988). The laws of emotion. *The American psychologist*, *43*(5), 349–358.

Gao, T., Fang, J., Liu, H., Liu, Z., Liu, C., Liu, P., Bao, Y., & Yan, W. (2022). LEGO-ABSA: A prompt-based task assemblable unified generative framework for multi-task aspect-based sentiment analysis. *Proceedings of the 29th International Conference on Computational Linguistics*, 7002–7012. https://aclanthology.org/2022.coling-1.610

Gao, T., Fisch, A., & Chen, D. (2021). Making pre-trained language models better few-shot learners. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 3816–3830. https://doi.org/10.18653/v1/2021.acl-long.295

Ghorbel, H. (2012). Experiments in cross-lingual sentiment analysis in discussion forums. In K. Aberer, A. Flache, W. Jager, L. Liu, J. Tang, & C. Guéret (Eds.), *Social informatics* (pp. 138–151). Springer Berlin Heidelberg.

Giachanou, A., & Crestani, F. (2016). Like it or not: A survey of twitter sentiment analysis methods. *ACM Computing Surveys (CSUR)*, *49*(2), 28.

Gildea, D., & Jurafsky, D. (2002). Automatic labeling of semantic roles. *Computational linguistics*, *28*(3), 245–288.

Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, *1*(12), 2009.

Goel, P., Kulshreshtha, D., Jain, P., & Shukla, K. K. (2017). Prayas at EmoInt 2017: An ensemble of deep neural architectures for emotion intensity prediction in tweets. *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 58–65. https://doi.org/10.18653/v1/W17-5207

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning* [http://www.deeplearningbook.org]. MIT Press.

Graves, A., Mohamed, A.-r., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, 6645–6649.

Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, *18*(5-6), 602–610.

Habernal, I., Ptáček, T., & Steinberger, J. (2013). Sentiment analysis in Czech social media using supervised machine learning. *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 65–74. https://www.aclweb.org/anthology/W13-1609

Hajic, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., et al. (2009). The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, 1–18.

Han, R., Peng, T., Yang, C., Wang, B., Liu, L., & Wan, X. (2023). Is information extraction solved by chatgpt? an analysis of performance, evaluation criteria, robustness and errors. *arXiv preprint arXiv:2305.14450*.

Hardoon, D. R., Szedmak, S., & Shawe-Taylor, J. (2004). Canonical correlation analysis: An overview with application to learning methods. *Neural computation*, *16*(12).

Harris, Z. S. (1954). Distributional structure. *Word*, *10*(2-3), 146–162.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Hercig, T., Brychcín, T., Svoboda, L., & Konkol, M. (2016). UWB at SemEval-2016 task 5: Aspect based sentiment analysis. *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, 342–349. https://doi.org/10.18653/v1/S16-1055

Hercig, T., Brychcín, T., Svoboda, L., Konkol, M., & Steinberger, J. (2016). Unsupervised methods to improve aspect-based sentiment analysis in czech. *Computación y Sistemas*, *20*(3), 365–375.

Hewitt, J. (2019, April). *Finding syntax with structural probes*. Retrieved May 11, 2020, from https://nlp.stanford.edu//~johnhew//structural-probe.html?utm_source=quora&utm_medium=referral#the-structural-probe

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hosseini-Asl, E., Liu, W., & Xiong, C. (2022). A generative language model for few-shot aspect-based sentiment analysis. *Findings of the Association for Computational Linguistics: NAACL 2022*, 770–787. https://doi.org/10.18653/v1/2022.findings-naacl.58

Huang, J., Xiang, C., Yuan, S., Yuan, D., & Huang, X. (2019). Character-aware convolutional recurrent networks with self-attention for emotion detection on twitter. *2019 International Joint Conference on Neural Networks (IJCNN)*, 1–8.

Jain, S., & Batra, S. (2015). Cross lingual sentiment analysis using modified BRAE. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 159–168. https://doi.org/10.18653/v1/D15-1016

Jakob, N., & Gurevych, I. (2010). Extracting opinion targets in a single and cross-domain setting with conditional random fields. *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 1035–1045. https://www.aclweb.org/anthology/D10-1101

Jiang, H., He, P., Chen, W., Liu, X., Gao, J., & Zhao, T. (2020). SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2177–2190. https://doi.org/10.18653/v1/2020.acl-main.197

Jin, W., & Ho, H. H. (2009). A novel lexicalized hmm-based learning framework for web opinion mining. *Proceedings of the 26th Annual International Conference on Machine Learning*, 465–472. https://doi.org/10.1145/1553374.1553435

Jindal, N., & Liu, B. (2006a). Identifying comparative sentences in text documents. *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 244–251. https://doi.org/10.1145/1148170.1148215

Jindal, N., & Liu, B. (2006b). Mining comparative sentences and relations. *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, 1331–1336. http://dl.acm.org/citation.cfm?id=1597348.1597400

Jurafsky, D., & Martin, J. H. (2009). *Speech and language processing (2nd edition)*. Prentice-Hall, Inc.

Jurafsky, D., & Martin, J. H. (2024). *Speech and language processing (3nd edition draft)*.

Karita, S., Chen, N., Hayashi, T., Hori, T., Inaguma, H., Jiang, Z., Someki, M., Soplin, N. E. Y., Yamamoto, R., Wang, X., et al. (2019). A comparative study on transformer vs rnn in speech applications. *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 449–456.

Khalil, T., & El-Beltagy, S. R. (2016). NileTMRG at SemEval-2016 task 5: Deep convolutional neural networks for aspect category and sentiment extraction. *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, 271–276. https://doi.org/10.18653/v1/S16-1043

Khodak, M., Saunshi, N., Liang, Y., Ma, T., Stewart, B., & Arora, S. (2018). A la carte embedding: Cheap but effective induction of semantic feature vectors. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 12–22. https://doi.org/10.18653/v1/P18-1002

Kim, S.-M., & Hovy, E. (2004). Determining the sentiment of opinions. *Proceedings of the 20th International Conference on Computational Linguistics*. https://doi.org/10.3115/1220355.1220555

Kim, Y. (2014). Convolutional neural networks for sentence classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1746–1751. https://doi.org/10.3115/v1/D14-1181

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In Y. Bengio & Y. LeCun (Eds.), *3rd international conference on learning representations, ICLR 2015, san diego, ca, usa, may 7-9, 2015, conference track proceedings*. http://arxiv.org/abs/1412.6980

Kiritchenko, S., Zhu, X., Cherry, C., & Mohammad, S. (2014, August). NRC-Canada-2014: Detecting aspects and sentiment in customer reviews. In P. Nakov & T. Zesch (Eds.), *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)* (pp. 437–442). Association for Computational Linguistics. https://doi.org/10.3115/v1/S14-2076

Kiritchenko, S., Zhu, X., & Mohammad, S. M. (2014). Sentiment analysis of short informal texts. *J. Artif. Int. Res.*, *50*(1), 723–762.

Kłeczek, D. (2020). Polbert: Attacking polish nlp tasks with transformers. In M. Ogrodniczuk & Ł. Kobyliński (Eds.), *Proceedings of the poleval 2020 workshop*. Institute of Computer Science, Polish Academy of Sciences.

Klinger, R., De Clercq, O., Mohammad, S., & Balahur, A. (2018, October). IEST: WASSA-2018 implicit emotions shared task. In A. Balahur, S. M. Mohammad, V. Hoste, & R. Klinger (Eds.), *Proceedings of the 9th workshop on computational approaches to subjectivity, sentiment and social media analysis* (pp. 31–42). Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-6206

Kocián, M., Náplava, J., Štancl, D., & Kadlec, V. (2022). Siamese bert-based model for web search relevance ranking evaluated on a new czech dataset. *Proceedings of the AAAI Conference on Artificial Intelligence*, *36*(11), 12369–12377. https://doi.org/10.1609/aaai.v36i11.21502

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. *MT summit*, *5*, 79–86.

Köper, M., Kim, E., & Klinger, R. (2017). IMS at EmoInt-2017: Emotion intensity prediction with affective norms, automatically extended resources and deep learning. *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 50–57. https://doi.org/10.18653/v1/W17-5206

Křen, M., Cvrček, V., Čapka, T., Čermáková, A., Hnátková, M., Chlumská, L., Jelínek, T., Kováříková, D., Petkevič, V., Procházka, P., Skoumalová, H., Škrabal, M., Truneček, P., Vondřička, P., & Zasina, A. (2016). SYN v4: Large corpus of written czech [LINDAT / CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University]. http://hdl.handle.net/11234/1-1846

Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 66–75. https://doi.org/10.18653/v1/P18-1007

Kudo, T., & Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 66–71. https://doi.org/10.18653/v1/D18-2012

Kuriyozov, E., Doval, Y., & Gómez-Rodríguez, C. (2020). Cross-lingual word embeddings for Turkic languages. *Proceedings of the Twelfth Language Resources and Evaluation Conference*, 4054–4062. https://aclanthology.org/2020.lrec-1.499

Lafferty, J. D., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of the Eighteenth International Conference on Machine Learning*, 282–289.

Lakew, S. M., Cettolo, M., & Federico, M. (2018, August). A comparison of transformer and recurrent neural networks on multilingual neural machine translation. In E. M. Bender, L. Derczynski, & P. Isabelle (Eds.), *Proceedings of the 27th international conference on computational linguistics* (pp. 641–652). Association for Computational Linguistics. https://aclanthology.org/C18-1054

Lample, G., Conneau, A., Ranzato, M., Denoyer, L., & Jégou, H. (2018). Word translation without parallel data. *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. https://openreview.net/forum?id=H196sainb

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2020). ALBERT: A lite BERT for self-supervised learning of language representations. *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. https://openreview.net/forum?id=H1eA7AEtvS

Lazaridou, A., Dinu, G., & Baroni, M. (2015). Hubness and pollution: Delving into cross-space mapping for zero-shot learning. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 270–280. https://doi.org/10.3115/v1/P15-1027

Le, H., Vial, L., Frej, J., Segonne, V., Coavoux, M., Lecouteux, B., Allauzen, A., Crabbé, B., Besacier, L., & Schwab, D. (2020). FlauBERT: Unsupervised language model pretraining for French. *Proceedings of the Twelfth Language Resources and Evaluation Conference*, 2479–2490. https://aclanthology.org/2020.lrec-1.302

Lehečka, J., Švec, J., Ircing, P., & Šmídl, L. (2020). Bert-based sentiment analysis using distillation. In L. Espinosa-Anke, C. Martín-Vide, & I. Spasić (Eds.), *Statistical language and speech processing* (pp. 58–70). Springer International Publishing.

Lenc, L., & Hercig, T. (2016). Neural networks for sentiment analysis in czech. *ITAT*, 48–55.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 7871–7880. https://doi.org/10.18653/v1/2020.acl-main.703

Li, Y., Yin, C., Zhong, S.-h., & Pan, X. (2020). Multi-instance multi-label learning networks for aspect-category sentiment analysis. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 3550–3560. https://doi.org/10.18653/v1/2020.emnlp-main.287

Liang, B., Su, H., Gui, L., Cambria, E., & Xu, R. (2022). Aspect-based sentiment analysis via affective knowledge enhanced graph convolutional networks. *Knowledge-Based Systems*, *235*, 107643. https://doi.org/https://doi.org/10.1016/j.knosys.2021.107643

Libovický, J., Rosa, R., Helcl, J., & Popel, M. (2018). Solving three czech nlp tasks with end-to-end neural models. *ITAT*, 138–143.

Liu, B. (2006). *Web data mining: Exploring hyperlinks, contents, and usage data (data-centric systems and applications)*. Springer-Verlag.

Liu, B., et al. (2010). Sentiment analysis and subjectivity. *Handbook of natural language processing*, *2*(2010), 627–666.

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, *5*(1), 1–167.

Liu, Cohn, T., & Baldwin, T. (2018). Recurrent entity networks with delayed memory update for targeted aspect-based sentiment analysis. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 278–283. https://doi.org/10.18653/v1/N18-2045

Liu, H., Tam, D., Muqeeth, M., Mohta, J., Huang, T., Bansal, M., & Raffel, C. A. (2022). Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Advances in neural information processing systems* (pp. 1950–1965, Vol. 35). Curran Associates, Inc.

Liu, J., Teng, Z., Cui, L., Liu, H., & Zhang, Y. (2021). Solving aspect category sentiment analysis as a text generation task. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 4406–4416. https://doi.org/10.18653/v1/2021.emnlp-main.361

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, *55*(9), 1–35. https://doi.org/10.1145/3560815

Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L., & Shazeer, N. (2018). Generating wikipedia by summarizing long sequences. *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. https://openreview.net/forum?id=Hyg0vbWC-

Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M., & Zettlemoyer, L. (2020). Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, *8*, 726–742. https://doi.org/10.1162/tacl_a_00343

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692.*

Loshchilov, I., & Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv: 1711.05101.*

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning word vectors for sentiment analysis. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 142–150. https://www.aclweb.org/anthology/P11-1015

Manning, C., Raghavan, P., & Schütze, H. (2010). Introduction to information retrieval. *Natural Language Engineering*, *16*(1), 100–103.

Mäntylä, M., Adams, B., Destefanis, G., Graziotin, D., & Ortu, M. (2016). Mining valence, arousal, and dominance: Possibilities for detecting burnout and productivity? *Proceedings of the 13th International Conference on Mining Software Repositories*, 247–258.

Mäntylä, M. V., Graziotin, D., & Kuutila, M. (2018). The evolution of sentiment analysis—a review of research topics, venues, and top cited papers. *Computer Science Review*, *27*, 16–32.

Mao, R., Liu, Q., He, K., Li, W., & Cambria, E. (2022). The biases of pre-trained language models: An empirical study on prompt-based sentiment analysis and emotion detection. *IEEE Transactions on Affective Computing*.

Martin, L., Muller, B., Ortiz Suárez, P. J., Dupont, Y., Romary, L., de la Clergerie, É., Seddah, D., & Sagot, B. (2020). CamemBERT: A tasty French language model. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 7203–7219. https://doi.org/10.18653/v1/2020.acl-main.645

Martineau, J., & Finin, T. W. (2009). Delta tfidf: An improved feature space for sentiment analysis. *ICWSM*.

Maynard, D., & Funk, A. (2011). Automatic detection of political opinions in tweets. *Extended Semantic Web Conference*, 88–99.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, *5*(4), 115–133.

Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, *5*(4), 1093–1113.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781.*

Mikolov, T., Le, Q. V., & Sutskever, I. (2013). Exploiting similarities among languages for machine translation. *CoRR*, *abs/1309.4168*. http://arxiv.org/abs/1309.4168

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, 3111–3119.

Min, S., Lyu, X., Holtzman, A., Artetxe, M., Lewis, M., Hajishirzi, H., & Zettlemoyer, L. (2022). Rethinking the role of demonstrations: What makes in-context learning work? *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 11048–11064. https://doi.org/10.18653/v1/2022.emnlp-main.759

Mitchell, M., Aguilar, J., Wilson, T., & Van Durme, B. (2013). Open domain targeted sentiment. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1643–1654.

Mohammad, S. (2018). Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 english words. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 174–184.

Mohammad, S., & Bravo-Marquez, F. (2017). WASSA-2017 shared task on emotion intensity. *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 34–49. https://doi.org/10.18653/v1/W17-5205

Mohammad, S., Bravo-Marquez, F., Salameh, M., & Kiritchenko, S. (2018, June). SemEval-2018 task 1: Affect in tweets. In M. Apidianaki, S. M. Mohammad, J. May, E. Shutova, S. Bethard, & M. Carpuat (Eds.), *Proceedings of the 12th international workshop on semantic evaluation* (pp. 1–17). Association for Computational Linguistics. https://doi.org/10.18653/v1/S18-1001

Mukherjee, S., Mitra, A., Jawahar, G., Agarwal, S., Palangi, H., & Awadallah, A. (2023). Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv*.

Nakov, P., Rosenthal, S., Kozareva, Z., Stoyanov, V., Ritter, A., & Wilson, T. (2013). SemEval-2013 task 2: Sentiment analysis in Twitter. *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, 312–320. https://aclanthology.org/S13-2052

Nandi, R., Maiya, G., Kamath, P., & Shekhar, S. (2021a). An empirical evaluation of word embedding models for subjectivity analysis tasks. *2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, 1–5.

Nandi, R., Maiya, G., Kamath, P., & Shekhar, S. (2021b). An empirical evaluation of word embedding models for subjectivity analysis tasks. *2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, 1–5.

Ng, A. Y., & Jordan, M. I. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In T. G. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Advances in neural information processing systems 14* (pp. 841–848). MIT Press.

OpenAI. (2022, November). *Openai: Introducing chatgpt*. Retrieved July 5, 2023, from https://openai.com/blog/chatgpt

OpenAI. (2023). GPT-4 technical report. *CoRR*, *abs/2303.08774*. https://doi.org/10.48550/ARXIV.2303.08774

Osgood, C. E., Suci, G. J., & Tannenbaum, P. H. (1957). *The measurement of meaning*. University of Illinois press.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, *35*, 27730–27744.

Pak, A., & Paroubek, P. (2010). Twitter as a corpus for sentiment analysis and opinion mining. *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. http://www.lrec-conf.org/proceedings/lrec2010/pdf/385_Paper.pdf

Palmer, M., Gildea, D., & Kingsbury, P. (2005). The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, *31*(1), 71–106. https://doi.org/10.1162/0891201053630264

Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359.

Pang, B., & Lee, L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, 271–278. https://doi.org/10.3115/1218955.1218990

Pang, B., & Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, 115–124. https://doi.org/10.3115/1219840.1219855

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, *2*(1–2), 1–135. https://doi.org/10.1561/1500000011

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? sentiment classification using machine learning techniques. *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, 79–86. https://doi.org/10.3115/1118693.1118704

Papay, S., Klinger, R., & Padó, S. (2022). Constraining linear-chain crfs to regular languages. *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. https://openreview.net/forum?id=jbrgwbv8nD

Park, H., Vyas, Y., & Shah, K. (2022). Efficient classification of long documents using transformers. *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 702–709. https://doi.org/10.18653/v1/2022.acl-short.79

Parrott, W. G. (2001). *Emotions in social psychology: Essential readings*. Psychology Press.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543. https://doi.org/10.3115/v1/D14-1162

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2227–2237. https://doi.org/10.18653/v1/N18-1202

Piskorski, J., Babych, B., Kancheva, Z., Kanishcheva, O., Lebedeva, M., Marcińczuk, M., Nakov, P., Osenova, P., Pivovarova, L., Pollak, S., Přibáň, P., Radev, I., Robnik-Sikonja, M., Starko, V., Steinberger, J., & Yangarber, R. (2021, April). Slav-NER: The 3rd cross-lingual challenge on recognition, normalization, classification, and linking of named entities across Slavic languages. In B. Babych, O. Kanishcheva, P. Nakov, J. Piskorski, L. Pivovarova, V. Starko, J. Steinberger, R. Yangarber, M. Marcińczuk, S. Pollak, P. Přibáň, & M. Robnik-Šikonja (Eds.), *Proceedings of the 8th workshop on balto-slavic natural language processing* (pp. 122–133). Association for Computational Linguistics. https://aclanthology.org/2021.bsnlp-1.15

Piskorski, J., Laskova, L., Marcińczuk, M., Pivovarova, L., Přibáň, P., Steinberger, J., & Yangarber, R. (2019, August). The second cross-lingual challenge on recognition, normalization, classification, and linking of named entities across Slavic languages. In T. Erjavec, M. Marcińczuk, P. Nakov, J. Piskorski, L. Pivovarova, J. Šnajder, J. Steinberger, & R. Yangarber (Eds.), *Proceedings of the 7th workshop on balto-slavic natural language processing* (pp. 63–74). Association for Computational Linguistics. https://doi.org/10.18653/v1/W19-3709

Platt, J. C. (1999). 12 fast training of support vector machines using sequential minimal optimization. *Advances in kernel methods*, 185–208.

Plutchik, R. (1980). A general psychoevolutionary theory of emotion. In *Theories of emotion* (pp. 3–33). Elsevier.

Polignano, M., Basile, P., de Gemmis, M., & Semeraro, G. (2019). A comparison of word-embeddings in emotion detection from text using bilstm, cnn and self-attention. *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization*, 63–68. https://doi.org/10.1145/3314183.3324983

Pontiki, M., Galanis, D., Papageorgiou, H., Androutsopoulos, I., Manandhar, S., AL-Smadi, M., Al-Ayyoub, M., Zhao, Y., Qin, B., De Clercq, O., Hoste, V., Apidianaki, M., Tannier, X., Loukachevitch, N., Kotelnikov, E., Bel, N., Jiménez-Zafra, S. M., & Eryiğit, G. (2016). SemEval-2016 task 5: Aspect based sentiment analysis. *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, 19–30. https://doi.org/10.18653/v1/S16-1002

Pontiki, M., Galanis, D., Papageorgiou, H., Manandhar, S., & Androutsopoulos, I. (2015). SemEval-2015 task 12: Aspect based sentiment analysis. *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, 486–495. https://doi.org/10.18653/v1/S15-2082

Pontiki, M., Galanis, D., Pavlopoulos, J., Papageorgiou, H., Androutsopoulos, I., & Manandhar, S. (2014). SemEval-2014 task 4: Aspect based sentiment analysis. *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, 27–35. https://doi.org/10.3115/v1/S14-2004

Poria, S., Cambria, E., & Gelbukh, A. (2016). Aspect extraction for opinion mining with a deep convolutional neural network. *Know.-Based Syst.*, *108*(100), 42–49. https://doi.org/10.1016/j.knosys.2016.06.009

Pražák, O., Přibáň, P., & Taylor, S. (2020). UWB @ DIACR-Ita: Lexical semantic change detection with CCA and orthogonal transformation. In V. Basile, D. Croce, M. D. Maro, & L. C. Passaro (Eds.), *Proceedings of the seventh evaluation campaign of natural language processing and speech tools for italian. final workshop (EVALITA 2020), online event, december 17th, 2020* (Vol. 2765). CEUR-WS.org. http://ceur-ws.org/Vol-2765/paper110.pdf

Pražák, O., Přibáň, P., Taylor, S., & Sido, J. (2020, December). UWB at SemEval-2020 task 1: Lexical semantic change detection. In A. Herbelot, X. Zhu, A. Palmer, N. Schneider, J. May, & E. Shutova (Eds.), *Proceedings of the fourteenth workshop on semantic evaluation* (pp. 246–254). International Committee for Computational Linguistics. https://doi.org/10.18653/v1/2020.semeval-1.30

Přibáň, P. (2020). Multilingual sentiment analysis. https://dspace5.zcu.cz/handle/11025/42665

Přibáň, P., & Balahur, A. (2023). Comparative analyses of multilingual sentiment analysis systems for news and social media. In A. Gelbukh (Ed.), *Computational linguistics and intelligent text processing* (pp. 260–279). Springer Nature Switzerland.

Přibáň, P., Baloun, J., Martínek, J., Lenc, L., Prantl, M., & Král, P. (2023). Towards automatic medical report classification in czech. *ICAART (3)*, 228–233.

Přibáň, P., Hercig, T., & Lenc, L. (2018). UWB at SemEval-2018 task 1: Emotion intensity detection in tweets. *Proceedings of the 12th International Workshop on Semantic Evaluation*, 133–140. https://doi.org/10.18653/v1/S18-1018

Přibáň, P., Hercig, T., & Steinberger, J. (2019, September). Machine learning approach to fact-checking in West Slavic languages. In R. Mitkov & G. Angelova (Eds.), *Proceedings of the international conference on recent advances in natural language processing (ranlp 2019)* (pp. 973–979). INCOMA Ltd. https://doi.org/10.26615/978-954-452-056-4_113

Přibáň, P., & Martínek, J. (2018). UWB at IEST 2018: Emotion prediction in tweets with bidirectional long short-term memory neural network. *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 224–230. https://doi.org/10.18653/v1/W18-6232

Přibáň, P., Ondřej, P., & Stephen, T. (2021). Uwb@ rushifteval: Measuring semantic difference as per-word variation in aligned semantic spaces. *Komp'yuternaya Lingvistika i Intellektual'nye Tekhnologii: Dialogue conference.—-2021.* https://doi.org/10.28995/2075-7182-2021-20-1188-1196

Přibáň, P., & Pražák, O. (2023, September). Improving aspect-based sentiment with end-to-end semantic role labeling model. In R. Mitkov & G. Angelova (Eds.), *Proceedings of the 14th international conference on recent advances in natural language processing* (pp. 888–897). INCOMA Ltd., Shoumen, Bulgaria. https://aclanthology.org/2023.ranlp-1.96

Přibáň, P., Šmíd, J., Mištera, A., & Král, P. (2022). Linear transformations for cross-lingual sentiment analysis. In P. Sojka, A. Horák, I. Kopeček, & K. Pala (Eds.), *Text, speech, and dialogue* (pp. 125–137). Springer International Publishing.

Přibáň, P., Šmíd, J., Steinberger, J., & Mištera, A. (2024). A comparative study of cross-lingual sentiment analysis. *Expert Systems with Applications*, *247*, 123247. https://doi.org/https://doi.org/10.1016/j.eswa.2024.123247

Přibáň, P., & Steinberger, J. (2021). Are the multilingual models better? improving Czech sentiment with transformers. *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, 1138–1149. https://aclanthology.org/2021.ranlp-1.128

Přibáň, P., & Steinberger, J. (2022). Czech dataset for cross-lingual subjectivity classification. *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, 1381–1391. https://aclanthology.org/2022.lrec-1.148

Přibáň, P., & Taylor, S. (2019, August). ZCU-NLP at MADAR 2019: Recognizing Arabic dialects. In W. El-Hajj, L. H. Belguith, F. Bougares, W. Magdy, I. Zitouni, N. Tomeh, M. El-Haj, & W. Zaghouani (Eds.), *Proceedings of the fourth arabic natural language processing workshop* (pp. 208–213). Association for Computational Linguistics. https://doi.org/10.18653/v1/W19-4623

Ptáček, T., Habernal, I., & Hong, J. (2014). Sarcasm detection on Czech and English twitter. *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 213–223. https://www.aclweb.org/anthology/C14-1022

Qin, C., Zhang, A., Zhang, Z., Chen, J., Yasunaga, M., & Yang, D. (2023). Is chatgpt a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476.*

Rabiner, L. R. (1990). A tutorial on hidden markov models and selected applications in speech recognition. In *Readings in speech recognition* (pp. 267–296). Morgan Kaufmann Publishers Inc.

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. Retrieved July 5, 2023, from https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, *1*(8), 9.

Radovanović, M., Nanopoulos, A., & Ivanović, M. (2010). Hubs in space: Popular nearest neighbors in high-dimensional data. *J. Mach. Learn. Res.*, *11*, 2487–2531.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, *21*(140), 1–67. http://jmlr.org/papers/v21/20-074.html

Řehůřek, R., & Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. *New Challenges For NLP Frameworks*, 45–50. http://is.muni.cz/publication/884893/en

Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 3982–3992. https://doi.org/10.18653/v1/D19-1410

Reitan, J., Faret, J., Gambäck, B., & Bungum, L. (2015). Negation scope detection for twitter sentiment analysis. *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 99–108.

Reyes, A., Rosso, P., & Buscaldi, D. (2012). From humor recognition to irony detection: The figurative language of social media. *Data Knowl. Eng.*, *74*, 1–12. https://doi.org/10.1016/j.datak.2012.02.005

Rietzler, A., Stabinger, S., Opitz, P., & Engl, S. (2020). Adapt or get left behind: Domain adaptation through BERT language model finetuning for aspect-target sentiment classification. *Proceedings of the Twelfth Language Resources and Evaluation Conference*, 4933–4941. https://aclanthology.org/2020.lrec-1.607

Riloff, E., Patwardhan, S., & Wiebe, J. (2006). Feature subsumption for opinion analysis. *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, 440–448. https://www.aclweb.org/anthology/W06-1652

Riloff, E., & Wiebe, J. (2003). Learning extraction patterns for subjective expressions. *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, 105–112. https://www.aclweb.org/anthology/W03-1014

Roberts, K., Roach, M. A., Johnson, J., Guthrie, J., & Harabagiu, S. M. (2012). EmpaTweet: Annotating and detecting emotions on twitter. *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, 3806–3813. http://www.lrec-conf.org/proceedings/lrec2012/pdf/201_Paper.pdf

Rosenthal, S., Farra, N., & Nakov, P. (2017). SemEval-2017 task 4: Sentiment analysis in twitter. *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 502–518. https://doi.org/10.18653/v1/S17-2088

Ruder, S., Vulić, I., & Søgaard, A. (2019). A survey of cross-lingual word embedding models. *J. Artif. Int. Res.*, *65*(1), 569–630. https://doi.org/10.1613/jair.1.11640

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In *Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1: Foundations* (pp. 318–362). MIT Press.

Russell, J. A. (1980). A circumplex model of affect. *Journal of personality and social psychology*, *39*(6), 1161.

Russell, J. A. (2003). Core affect and the psychological construction of emotion. *Psychological review*, *110*(1), 145.

Saeidi, M., Bouchard, G., Liakata, M., & Riedel, S. (2016). SentiHood: Targeted aspect based sentiment analysis dataset for urban neighbourhoods. *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 1546–1556. https://aclanthology.org/C16-1146

Safaya, A., Abdullatif, M., & Yuret, D. (2020). KUISAIL at SemEval-2020 task 12: BERT-CNN for offensive speech identification in social media. *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, 2054–2059. https://doi.org/10.18653/v1/2020.semeval-1.271

Sazzed, S. (2020). Cross-lingual sentiment classification in low-resource Bengali language. *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, 50–60. https://doi.org/10.18653/v1/2020.wnut-1.8

Scherer, K. R., & Wallbott, H. G. (1994). Evidence for universality and cultural variation of differential emotion response patterning. *Journal of personality and social psychology*, *66*(2), 310.

Schlechtweg, D., McGillivray, B., Hengchen, S., Dubossarsky, H., & Tahmasebi, N. (2020, December). SemEval-2020 task 1: Unsupervised lexical semantic change detection. In A. Herbelot, X. Zhu, A. Palmer, N. Schneider, J. May, & E. Shutova (Eds.), *Proceedings of the fourteenth workshop on semantic evaluation* (pp. 1–23). International Committee for Computational Linguistics. https://doi.org/10.18653/v1/2020.semeval-1.1

Schuff, H., Barnes, J., Mohme, J., Padó, S., & Klinger, R. (2017). Annotation, modelling and analysis of fine-grained emotions on a stance and sentiment detection corpus. *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 13–23. https://doi.org/10.18653/v1/W17-5203

Schuster, M., & Nakajima, K. (2012). Japanese and korean voice search. *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 5149–5152.

Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, *45*(11), 2673–2681.

Schweter, S. (2020, April). *Berturk - bert models for turkish* (Version 1.0.0). Zenodo. https://doi.org/10.5281/zenodo.3770924

Sennrich, R., Haddow, B., & Birch, A. (2016). Neural machine translation of rare words with subword units. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1715–1725. https://doi.org/10.18653/v1/P16-1162

Shamma, D. A., Kennedy, L., & Churchill, E. F. (2009). Tweet the debates: Understanding community annotation of uncollected sources. *Proceedings of the first SIGMM workshop on Social media*, 3–10.

Sharma, M. (2020). Polarity detection in a cross-lingual sentiment analysis using spacy. *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 490–496. https://doi.org/10.1109/ICRITO48877.2020.9197829

Shi, P., & Lin, J. (2019). Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.

Shrivastava, K., Kumar, S., & Jain, D. K. (2019). An effective approach for emotion detection in multimedia text data using sequence based convolutional neural network. *Multimedia Tools and Applications*, *78*(20), 29607–29639.

Shu, L., Xu, H., & Liu, B. (2017). Lifelong learning CRF for supervised aspect extraction. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 148–154. https://doi.org/10.18653/v1/P17-2023

Sido, J., Pražák, O., Přibáň, P., Pašek, J., Seják, M., & Konopík, M. (2021). Czert – Czech BERT-like model for language representation. *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, 1326–1338. https://aclanthology.org/2021.ranlp-1.149

Singh, N. K., Tomar, D. S., & Sangaiah, A. K. (2018). Sentiment analysis: A review and comparative analysis over social media. *Journal of Ambient Intelligence and Humanized Computing*, 1–21.

Šmíd, J., & Přibáň, P. (2023, September). Prompt-based approach for Czech sentiment analysis. In R. Mitkov & G. Angelova (Eds.), *Proceedings of the 14th international conference on recent advances in natural language processing* (pp. 1110–1120). INCOMA Ltd., Shoumen, Bulgaria. https://aclanthology.org/2023.ranlp-1.118

Šmíd, J., Přibáň, P., & Pražák, O. (2024). Czech dataset for complex aspect-based sentiment analysis tasks. *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation*. https://home.zcu.cz/~pribanp/LREC-2024/paper.pdf

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., & Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1631–1642. https://www.aclweb.org/anthology/D13-1170

Soleymani, R., Beaulieu, J., & Farret, J. (2021). Experimental comparison of transformers and reformers for text classification. *Sensors & Transducers*, *249*(2), 110–118.

Speriosu, M., Sudan, N., Upadhyay, S., & Baldridge, J. (2011). Twitter polarity classification with label propagation over lexical links and the follower graph. *Proceedings of the First workshop on Unsupervised Learning in NLP*, 53–63.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Steinberger, J., Lenkova, P., Kabadjov, M., Steinberger, R., & Van der Goot, E. (2011). Multilingual entity-centered sentiment analysis evaluated by parallel corpora. *Proceedings* *of the International Conference Recent Advances in Natural Language Processing 2011*, 770–775.

Straka, M. (2018). UDPipe 2.0 prototype at CoNLL 2018 UD shared task. *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, 197–207. https://doi.org/10.18653/v1/K18-2020

Straka, M., Náplava, J., Straková, J., & Samuel, D. (2021). RobeCzech: Czech RoBERTa, a Monolingual Contextualized Language Representation Model. In K. Ekštein, F. Pártl, & M. Konopík (Eds.), *Text, speech, and dialogue* (pp. 197–209). Springer International Publishing.

Straková, J., Straka, M., & Hajic, J. (2014). Open-source tools for morphology, lemmatization, pos tagging and named entity recognition. *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 13–18.

Strapparava, C., & Mihalcea, R. (2007). SemEval-2007 task 14: Affective text. *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, 70–74. https://www.aclweb.org/anthology/S07-1013

Strapparava, C., & Mihalcea, R. (2008). Learning to identify emotions in text. *Proceedings of the 2008 ACM symposium on Applied computing*, 1556–1560.

Sun, C., Huang, L., & Qiu, X. (2019). Utilizing BERT for aspect-based sentiment analysis via constructing auxiliary sentence. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 380–385. https://doi.org/10.18653/v1/N19-1035

Sun, C., Qiu, X., Xu, Y., & Huang, X. (2019). How to fine-tune bert for text classification? *China National Conference on Chinese Computational Linguistics*, 194–206.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems 27* (pp. 3104–3112). Curran Associates, Inc. http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf

Tang, Y., Tran, C., Li, X., Chen, P.-J., Goyal, N., Chaudhary, V., Gu, J., & Fan, A. (2021, August). Multilingual translation from denoising pre-training. In C. Zong, F. Xia, W. Li, & R. Navigli (Eds.), *Findings of the association for computational linguistics: Acl-ijcnlp 2021* (pp. 3450–3466). Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.findings-acl.304

Taylor, S., Přibáň, P., & Prazák, O. (2021). Comparewords: Measuring semantic change in word usage in different corpora. *Softw. Impacts*, *8*, 100067. https://doi.org/10.1016/J.SIMPA.2021.100067

Taylor, W. L. (1953). “cloze procedure”: A new tool for measuring readability. *Journalism quarterly*, *30*(4), 415–433.

Thakkar, G., Mikelic, N., & Marko, T. (2021). Multi-task learning for cross-lingual sentiment analysis. *Proceedings of the 2nd International Workshop on Cross-lingual Event-centric Open Analytics co-located with the 30th The Web Conference (WWW 2021, 2829*, 76–84.

Théophile, B. (2020). French sentiment analysis with bert. %5Curl%7Bhttps://github.com/TheophileBlard/french-sentiment-analysis-with-bert%7D

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., & Lample, G. (2023). Llama: Open and efficient foundation language models.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., … Scialom, T. (2023). Llama 2: Open foundation and fine-tuned chat models.

Van Hee, C., Lefever, E., & Hoste, V. (2018). SemEval-2018 task 3: Irony detection in English tweets. *Proceedings of The 12th International Workshop on Semantic Evaluation*, 39–50. https://doi.org/10.18653/v1/S18-1005

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems 30* (pp. 5998–6008). Curran Associates, Inc. http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

Villena-Román, J. (2013). Tass 2013 — workshop on sentiment analysis at sepln 2013: An overview. *Proceedings of the TASS workshop at SEPLN*, 112–125.

Virtanen, A., Kanerva, J., Ilo, R., Luoma, J., Luotolahti, J., Salakoski, T., Ginter, F., & Pyysalo, S. (2019). Multilingual is not enough: Bert for finnish.

Wan, H., Yang, Y., Du, J., Liu, Y., Qi, K., & Pan, J. Z. (2020). Target-aspect-sentiment joint detection for aspect-based sentiment analysis. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(05), 9122–9129. https://doi.org/10.1609/aaai.v34i05.6447

Wan, X. (2008). Using bilingual knowledge and ensemble techniques for unsupervised Chinese sentiment analysis. *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, 553–561. https://aclanthology.org/D08-1058

Wan, X. (2009). Co-training for cross-lingual sentiment classification. *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 235–243. https://aclanthology.org/P09-1027

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. (2019). Superglue: A stickier benchmark for general-purpose language understanding systems. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 32). Curran Associates, Inc.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. (2018). GLUE: A multi-task benchmark and analysis platform for natural language understanding. *Proceedings of* *the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 353–355. https://doi.org/10.18653/v1/W18-5446

Wang, C., & Banko, M. (2021). Practical transformer-based multilingual text classification. *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*, 121–129. https://doi.org/10.18653/v1/2021.naacl-industry.16

Wang, S., Fang, H., Khabsa, M., Mao, H., & Ma, H. (2021). Entailment as few-shot learner. *arXiv preprint arXiv:2104.14690.*

Wei, J., Wang, X., Schuurmans, D., Bosma, M., ichter brian, b., Xia, F., Chi, E., Le, Q. V., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Advances in neural information processing systems* (pp. 24824–24837, Vol. 35). Curran Associates, Inc.

Weischedel, R., Palmer, M., Marcus, M., Hovy, E., Pradhan, S., Ramshaw, L., Xue, N., Taylor, A., Kaufman, J., Franchini, M., et al. (2013). Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA, 23.*

White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J., & Schmidt, D. C. (2023). A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382.*

Wiebe, J. (2000). Learning subjective adjectives from corpora. *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, 735–740.

Wiebe, J., Bruce, R. F., & O'Hara, T. P. (1999). Development and use of a gold-standard data set for subjectivity classifications. *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*, 246–253.

Wiebe, J., & Riloff, E. (2005). Creating subjective and objective sentence classifiers from unannotated texts. *Proceedings of the 6th International Conference on Computational Linguistics and Intelligent Text Processing*, 486–497. https://doi.org/10.1007/978-3-540-30586-6_53

Wiebe, J., Wilson, T., & Cardie, C. (2005). Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, *39*(2-3), 165–210.

Wiebe, J. M., Bruce, R. F., & O'Hara, T. P. (1999). Development and use of a gold-standard data set for subjectivity classifications. *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, 246–253. https://doi.org/10.3115/1034678.1034721

Winata, G., Wu, S., Kulkarni, M., Solorio, T., & Preotiuc-Pietro, D. (2022). Cross-lingual few-shot learning on unseen languages. *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 777–791. https://aclanthology.org/2022.aacl-main.59

Wu, Z., & Ong, D. C. (2021). Context-guided bert for targeted aspect-based sentiment analysis. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*(16), 14094–14102.

Xiao, M., & Guo, Y. (2014). Distributed word representation learning for cross-lingual dependency parsing. *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, 119–129. https://doi.org/10.3115/v1/W14-1613

Xu, H., Liu, B., Shu, L., & Yu, P. (2019). BERT post-training for review reading comprehension and aspect-based sentiment analysis. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2324–2335. https://doi.org/10.18653/v1/N19-1242

Xu, H., Liu, B., Shu, L., & Yu, P. S. (2018). Double embeddings and CNN-based sequence labeling for aspect extraction. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 592–598. https://doi.org/10.18653/v1/P18-2094

Xu, Y., Cao, H., Du, W., & Wang, W. (2022). A survey of cross-lingual sentiment analysis: Methodologies, models and evaluations. *Data Science and Engineering*, 1–21.

Xue, L., Barua, A., Constant, N., Al-Rfou, R., Narang, S., Kale, M., Roberts, A., & Raffel, C. (2022). ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, *10*, 291–306. https://doi.org/10.1162/tacl_a_00461

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., & Raffel, C. (2021). MT5: A massively multilingual pre-trained text-to-text transformer. *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 483–498. https://doi.org/10.18653/v1/2021.naacl-main.41

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 1–11, Vol. 32). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2019/file/dc6a7e655d7e5840e66733e9ee67cc69-Paper.pdf

Yangarber, R., Piskorski, J., Dmitrieva, A., Marcińczuk, M., Přibáň, P., Rybak, P., & Steinberger, J. (2023, May). Slav-NER: The 4th cross-lingual challenge on recognition, normalization, classification, and linking of named entities across Slavic languages. In J. Piskorski, M. Marcińczuk, P. Nakov, M. Ogrodniczuk, S. Pollak, P. Přibáň, P. Rybak, J. Steinberger, & R. Yangarber (Eds.), *Proceedings of the 9th workshop on slavic natural language processing 2023 (slavicnlp 2023)* (pp. 179–189). Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.bsnlp-1.21

Zeyer, A., Bahar, P., Irie, K., Schlüter, R., & Ney, H. (2019). A comparison of transformer and lstm encoder decoder models for asr. *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 8–15.

Zhang, C., Li, Q., & Song, D. (2019). Aspect-based sentiment classification with aspect-specific graph convolutional networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 4568–4578. https://doi.org/10.18653/v1/D19-1464

Zhang, L., Wang, S., & Liu, B. (2018). Deep learning for sentiment analysis: A survey. *WIREs Data Mining and Knowledge Discovery*, *8*(4), e1253. https://doi.org/10.1002/widm.1253

Zhang, W., Deng, Y., Li, X., Yuan, Y., Bing, L., & Lam, W. (2021). Aspect sentiment quad prediction as paraphrase generation. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 9209–9219. https://doi.org/10.18653/v1/2021.emnlp-main.726

Zhang, W., Deng, Y., Liu, B., Pan, S. J., & Bing, L. (2023). Sentiment analysis in the era of large language models: A reality check. *arXiv preprint arXiv:2305.15005*.

Zhang, W., He, R., Peng, H., Bing, L., & Lam, W. (2021). Cross-lingual aspect-based sentiment analysis with aspect term code-switching. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 9220–9230. https://doi.org/10.18653/v1/2021.emnlp-main.727

Zhang, W., Li, X., Deng, Y., Bing, L., & Lam, W. (2021). Towards generative aspect-based sentiment analysis. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 504–510. https://doi.org/10.18653/v1/2021.acl-short.64

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, 649–657.

Zhao, H., Lu, Z., & Poupart, P. (2015). Self-adaptive hierarchical sentence model. *Proceedings of the 24th International Conference on Artificial Intelligence*, 4069–4076.

Zhong, Q., Ding, L., Liu, J., Du, B., & Tao, D. (2023). Can chatgpt understand too? a comparative study on chatgpt and fine-tuned bert. *arXiv preprint*. https://arxiv.org/abs/2302.10198

Zhou, H., Chen, L., Shi, F., & Huang, D. (2015). Learning bilingual sentiment word embeddings for cross-language sentiment classification. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 430–440. https://doi.org/10.3115/v1/P15-1042

Zhou, P., Shi, W., Tian, J., Qi, Z., Li, B., Hao, H., & Xu, B. (2016). Attention-based bidirectional long short-term memory networks for relation classification. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 207–212. https://doi.org/10.18653/v1/P16-2034

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., & He, Q. (2020). A comprehensive survey on transfer learning. *Proceedings of the IEEE, 109*(1), 43–76.

Zou, W. Y., Socher, R., Cer, D., & Manning, C. D. (2013). Bilingual word embeddings for phrase-based machine translation. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1393–1398. https://aclanthology.org/D13-1141

# Appendix A

In Appendix A.1, we provide monolingual results for sentiment classification experiments and the used hyper-parameters for training the corresponding models. The same hyper-parameters were used for the cross-lingual experiments. In Appendix A.2, we report the detailed results of the cross-lingual sentiment analysis experiments with linear transformations. In Appendix A.4, we report details about fine-tuning and the used hyper-parameters for experiments with the subjectivity datasets. We provide examples of prompts for binary classification in Appendix A.5. Appendix A.6 contains examples of outputs of LLMs.

## A.1 Results with Hyper-parameters

We provide monolingual results with the important hyper-parameters for all models we trained (fine-tuned) in Tables A.1, A.2, A.3 and A.4 for the French Allocine, English IMDB, Czech CSFD and English SST datasets, respectively. These tables contain the same information as Tables 7.3, 7.4, 7.5 and 7.6 in Section 7.5, which describes monolingual results. The models denoted by *CNN* and *LSTM* were trained with in-domain embeddings, while the models with the suffix *-F*, i.e., *CNN-F* and *LSTM-F*, were trained with the original fastText embeddings. For the LSTM and CNN models, there are two results separated by a slash, where the first number represents the accuracy score for the unnormalized embeddings, and the second number represents the score for the normalized version of the word embeddings. The values in parentheses separated by a slash character describe the hyper-parameters in the following order: learning rate, number of epochs and a learning rate scheduler. The learning rate scheduler is either constant (c) or linear (l). For example, the (1e-3 / 9 / l) values mean that the corresponding model was trained with a learning rate of 1e-3 for nine epochs and the learning rate was linearly decreased.

| Model | **Allocine** (French) |
|---|---|
| | Default/Normalized |
| CNN | $95.0^{\pm0.1}$ (1e-3 / 10 / c) / $95.1^{\pm0.1}$ (1e-3 / 8 / c) |
| CNN-F | $94.3^{\pm0.1}$ (1e-3 / 10 / c) / $94.7^{\pm0.2}$ (1e-3 / 5 / c) |
| LSTM | $96.4^{\pm0.1}$ (1e-3 / 10 / c) / $96.4^{\pm0.1}$ (1e-3 / 10 / c) |
| LSTM-F | $95.7^{\pm0.1}$ (1e-3 / 10 / c) / $95.9^{\pm0.1}$ (1e-3 / 7 / c) |
| CamemBERT | $97.5^{\pm0.0}$ (2e-5 / 2 / l) |
| mBERT | $96.2^{\pm0.1}$ (2e-6 / 15 / c) |
| XLM | $96.3^{\pm0.0}$ (2e-6 / 8 / l) |
| XLM-R$_{\text{Base}}$ | $96.9^{\pm0.0}$ (2e-6 / 8 / c) |
| XLM-R$_{\text{Large}}$ | $97.6^{\pm0.0}$ (2e-6 / 6 / l) |

Table A.1: Monolingual accuracy results for the French Allocine dataset (2 classes) with the used hyper-parameters.

| Model | **IMDB** (English) |
|---|---|
| | Default/Normalized |
| CNN | $91.8^{\pm0.1}$ (1e-3 / 7 / l) / $91.6^{\pm0.2}$ (1e-3 / 5 / c) |
| CNN-F | $89.3^{\pm0.6}$ (1e-3 / 10 / l) / $91.1^{\pm0.2}$ (1e-3 / 8 / c) |
| LSTM | $92.5^{\pm0.2}$ (1e-3 / 9 / l) / $92.6^{\pm0.4}$ (1e-3 / 10 / c) |
| LSTM-F | $90.7^{\pm0.7}$ (1e-3 / 10 / c) / $91.5^{\pm0.5}$ (1e-3 / 10 / c) |
| BERT$_{\text{Base-Cased}}$ | $93.7^{\pm0.0}$ (2e-5 / 37 / l) |
| mBERT | $92.4^{\pm0.4}$ (2e-6 / 57 / c) |
| XLM | $86.4^{\pm0.2}$ (2e-6 / 29 / l) |
| XLM-R$_{\text{Base}}$ | $94.5^{\pm0.2}$ (2e-6 / 43 / l) |
| XLM-R$_{\text{Large}}$ | $96.2^{\pm0.1}$ (2e-6 / 37 / l) |

Table A.2: Monolingual accuracy results for the English IMDB dataset (2 classes) with the used hyper-parameters.

| Model | **CSFD** (Czech) | |
|---|---|---|
| | **2 Classes** | **3 Classes** |
| | Default/Normalized | Default/Normalized |
| CNN | $93.9^{\pm0.1}$ (1e-3 / 6 / l) / $93.4^{\pm0.1}$ (1e-3 / 3 / c) | $83.7^{\pm0.1}$ (1e-3 / 5 / l) / $82.9^{\pm0.2}$ (1e-3 / 4 / c) |
| CNN-F | $91.5^{\pm0.2}$ (1e-3 / 10 / c) / $92.6^{\pm0.1}$ (1e-3 / 9 / c) | $80.3^{\pm0.1}$ (1e-3 / 7 / c) / $81.7^{\pm0.2}$ (1e-3 / 7 / c) |
| LSTM | $94.4^{\pm0.2}$ (1e-3 / 8 / c) / $93.9^{\pm0.1}$ (1e-3 / 6 / c) | $84.8^{\pm0.2}$ (1e-3 / 8 / c) / $84.2^{\pm0.1}$ (1e-3 / 9 / c) |
| LSTM-F | $92.1^{\pm0.3}$ (1e-3 / 10 / c) / $92.6^{\pm0.3}$ (1e-3 / 10 / c) | $81.8^{\pm0.3}$ (1e-3 / 9 / c) / $82.8^{\pm0.2}$ (1e-3 / 10 / c) |
| Czert-B | $94.4^{\pm0.1}$ (2e-5 / 15 / l) | $84.9^{\pm0.1}$ (2e-5 / 12 / l) |
| RobeCzech | $95.1^{\pm0.9}$ (2e-5 / 15 / l) | $86.0^{\pm0.2}$ (2e-6 / 13 / c) |
| Czech Electra | $93.2^{\pm0.4}$ (2e-5 / 15 / c) | $81.8^{\pm0.1}$ (2e-5 / 13 / c) |
| mBERT | $93.1^{\pm0.3}$ (2e-6 / 14 / c) | $82.9^{\pm0.1}$ (2e-6 / 13 / l) |
| XLM | $93.9^{\pm0.2}$ (2e-5 / 5 / l) | $83.8^{\pm0.1}$ (2e-5 / 11 / l) |
| XLM-R$_{\text{Base}}$ | $94.3^{\pm0.3}$ (2e-6 / 14 / c) | $85.0^{\pm0.1}$ (2e-6 / 15 / c) |
| XLM-R$_{\text{Large}}$ | $96.0^{\pm0.0}$ (2e-6 / 14 / c) | $87.2^{\pm0.1}$ (2e-6 / 11 / l) |

Table A.3: Monolingual accuracy results for the Czech CSFD dataset with the used hyper-parameters.

| Model | SST (English) | |
|---|---|---|
| | 2 Classes | 3 Classes |
| CNN | $84.4^{\pm0.6}$ (1e-3 / 8 / c) / $84.6^{\pm0.3}$ (1e-3 / 3 / c) | $66.4^{\pm1.1}$ (1e-3 / 4 / c) / $68.5^{\pm0.6}$ (1e-3 / 3 / c) |
| CNN-F | $83.7^{\pm0.2}$ (1e-3 / 8 / c) / $85.4^{\pm0.4}$ (1e-3 / 3 / c) | $66.1^{\pm1.0}$ (1e-3 / 9 / c) / $68.6^{\pm0.8}$ (1e-3 / 6 / c) |
| LSTM | $85.3^{\pm0.4}$ (1e-3 / 9 / c) / $84.5^{\pm1.2}$ (1e-3 / 9 / c) | $69.7^{\pm1.1}$ (1e-3 / 7 / c) / $68.2^{\pm1.7}$ (1e-3 / 10 / c) |
| LSTM-F | $84.3^{\pm0.6}$ (1e-3 / 9 / c) / $85.9^{\pm0.9}$ (1e-3 / 10 / c) | $70.4^{\pm0.7}$ (1e-3 / 9 / c) / $71.3^{\pm1.2}$ (1e-3 / 10 / c) |
| BERT$_{\text{Base-Cased}}$ | $91.0^{\pm0.1}$ (2e-6 / 57 / l) | $71.9^{\pm0.1}$ (2e-6 / 63 / l) |
| mBERT | $85.2^{\pm0.9}$ (2e-6 / 55 / l) | $65.1^{\pm0.4}$ (2.5e-7 / 50 / c) |
| XLM | $89.6^{\pm0.2}$ (2e-6 / 51 / l) | $70.5^{\pm0.4}$ (2e-6 / 4 / c) |
| XLM-R$_{\text{Base}}$ | $90.9^{\pm0.2}$ (2.5e-7 / 77 / c) | $73.5^{\pm0.2}$ (2.5e-7 / 56 / c) |
| XLM-R$_{\text{Large}}$ | $94.6^{\pm0.4}$ (2.5e-7 / 73 / c) | $78.1^{\pm0.5}$ (2.5e-7 / 57 / l) |

Table A.4: Monolingual accuracy results for the English SST dataset with the used hyper-parameters.

## A.2 Complete Cross-lingual Results for Linear Transformations

Here we report the complete results for individual linear transformations for cross-lingual sentiment analysis in Tables A.5, A.6, A.7, A.8, A.9, A.10, A.11, A.12, A.13, A.14, A.15 and A.16. In Section 7.6.1, we provide the averaged results over the individual linear transformations. The notation and formatting of these tables are identical to those in Section 7.6.1. The only difference in notation is that the underlined number always represents the better score from the normalized/unnormalized pair. The best results for each language and model pair are in bold.

| | | Evaluated on **Czech** | | | Evaluated on **English** | | |
|---|---|---|---|---|---|---|---|
| **Norm.** | **Method** | **Monoling.** | **EN-s ⇒CS-t** | **CS-t ⇒EN-s** | **Monoling.** | **CS-s ⇒EN-t** | **EN-t ⇒CS-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| | | | | **CNN** | | | |
| - | MSE | 93.9/91.5 | 88.2$^{\pm0.3}$/75.7$^{\pm1.4}$ | 87.5$^{\pm1.8}$/80.3$^{\pm1.4}$ | 91.8/89.3 | **84.4**$^{\pm0.4}$/78.7$^{\pm0.4}$ | 64.7$^{\pm0.5}$/70.3$^{\pm0.8}$ |
| | Orto | | 88.5$^{\pm0.1}$/78.9$^{\pm0.9}$ | **89.2**$^{\pm0.1}$/78.1$^{\pm0.9}$ | | 84.0$^{\pm0.1}$/79.7$^{\pm0.1}$ | 81.3$^{\pm0.3}$/79.3$^{\pm0.8}$ |
| | CCA | | 88.4$^{\pm0.1}$/76.3$^{\pm1.1}$ | 88.2$^{\pm0.1}$/79.2$^{\pm0.6}$ | | 83.9$^{\pm0.1}$/77.4$^{\pm0.5}$ | 80.2$^{\pm0.1}$/75.0$^{\pm0.6}$ |
| | Rank | | 85.7$^{\pm0.3}$/78.9$^{\pm0.7}$ | 88.0$^{\pm0.8}$/76.7$^{\pm0.6}$ | | 83.4$^{\pm0.2}$/77.6$^{\pm0.7}$ | 82.9$^{\pm0.3}$/73.5$^{\pm0.7}$ |
| | Or-Ra | | 83.3$^{\pm0.6}$/76.9$^{\pm1.7}$ | **89.2**$^{\pm0.1}$/79.2$^{\pm1.0}$ | | 79.6$^{\pm0.5}$/78.4$^{\pm0.5}$ | 82.3$^{\pm0.4}$/75.0$^{\pm0.8}$ |
| | Avg. | | 86.8/77.3 | 88.4/78.7 | | 83.1/78.4 | 78.3/74.6 |
| B | MSE | 93.4/92.6 | 87.6$^{\pm0.2}$/85.0$^{\pm0.3}$ | 87.9$^{\pm0.2}$/85.2$^{\pm0.4}$ | 91.6/91.1 | 85.4$^{\pm0.9}$/84.2$^{\pm0.5}$ | 81.4$^{\pm0.3}$/81.3$^{\pm0.3}$ |
| | Orto | | 88.2$^{\pm0.1}$/85.8$^{\pm0.1}$ | 87.3$^{\pm0.4}$/86.1$^{\pm0.1}$ | | 85.2$^{\pm0.2}$/82.2$^{\pm1.3}$ | 81.0$^{\pm2.9}$/79.8$^{\pm2.7}$ |
| | CCA | | 88.3$^{\pm0.1}$/85.9$^{\pm0.2}$ | 86.9$^{\pm0.2}$/86.0$^{\pm0.1}$ | | 85.2$^{\pm0.1}$/83.9$^{\pm0.1}$ | 83.6$^{\pm0.2}$/80.5$^{\pm2.1}$ |
| | Rank | | 87.8$^{\pm0.2}$/85.5$^{\pm0.2}$ | 87.9$^{\pm0.1}$/85.3$^{\pm0.2}$ | | 85.7$^{\pm0.2}$/84.7$^{\pm0.1}$ | 83.8$^{\pm0.1}$/84.0$^{\pm0.0}$ |
| | Or-Ra | | 87.6$^{\pm0.4}$/86.3$^{\pm0.2}$ | **88.4**$^{\pm0.1}$/85.7$^{\pm0.2}$ | | **85.9**$^{\pm0.1}$/79.6$^{\pm0.4}$ | 82.9$^{\pm0.8}$/82.5$^{\pm0.5}$ |
| | Avg. | | 87.9/85.7 | 87.7/85.7 | | 85.5/82.9 | 82.5/81.6 |
| | | | | **LSTM** | | | |
| - | MSE | 94.4/92.1 | 84.9$^{\pm0.7}$/80.6$^{\pm1.3}$ | 85.9$^{\pm2.6}$/79.3$^{\pm4.3}$ | 92.5/90.7 | **85.5**$^{\pm0.7}$/83.5$^{\pm1.9}$ | 67.7$^{\pm3.7}$/75.2$^{\pm2.9}$ |
| | Orto | | 87.6$^{\pm0.4}$/80.2$^{\pm2.3}$ | **87.9**$^{\pm0.5}$/80.2$^{\pm3.0}$ | | 73.6$^{\pm1.1}$/79.8$^{\pm1.5}$ | 74.9$^{\pm3.8}$/83.3$^{\pm1.2}$ |
| | CCA | | 86.6$^{\pm1.9}$/82.9$^{\pm0.7}$ | 87.4$^{\pm0.3}$/82.7$^{\pm0.5}$ | | 83.6$^{\pm0.8}$/66.9$^{\pm3.1}$ | 81.8$^{\pm1.5}$/82.6$^{\pm0.6}$ |
| | Rank | | 84.1$^{\pm1.2}$/75.4$^{\pm0.5}$ | 86.0$^{\pm1.5}$/82.6$^{\pm1.1}$ | | 74.1$^{\pm4.5}$/69.6$^{\pm2.8}$ | 83.8$^{\pm0.8}$/83.5$^{\pm0.5}$ |
| | Or-Ra | | 86.2$^{\pm0.4}$/73.2$^{\pm1.5}$ | 86.8$^{\pm0.7}$/82.9$^{\pm2.0}$ | | 82.0$^{\pm3.4}$/71.6$^{\pm3.4}$ | 84.9$^{\pm0.7}$/83.8$^{\pm0.8}$ |
| | Avg. | | 85.9/78.5 | 86.8/81.5 | | 79.8/74.3 | 78.6/81.7 |
| B | MSE | 93.9/92.6 | 86.7$^{\pm0.8}$/85.4$^{\pm0.6}$ | 87.2$^{\pm3.0}$/72.0$^{\pm7.3}$ | 92.6/91.5 | 85.1$^{\pm0.8}$/83.6$^{\pm1.3}$ | 76.0$^{\pm3.7}$/78.0$^{\pm4.6}$ |
| | Orto | | 85.1$^{\pm0.8}$/79.2$^{\pm9.0}$ | 84.9$^{\pm3.3}$/84.8$^{\pm0.7}$ | | 77.3$^{\pm3.8}$/85.3$^{\pm0.7}$ | 76.3$^{\pm3.4}$/85.1$^{\pm0.6}$ |
| | CCA | | 87.0$^{\pm1.3}$/83.8$^{\pm1.7}$ | 87.2$^{\pm0.6}$/83.7$^{\pm2.8}$ | | **86.0**$^{\pm1.9}$/82.6$^{\pm3.0}$ | 81.3$^{\pm3.5}$/78.0$^{\pm2.7}$ |
| | Rank | | 86.2$^{\pm0.9}$/81.4$^{\pm2.4}$ | **88.5**$^{\pm1.1}$/84.4$^{\pm0.6}$ | | 83.1$^{\pm1.7}$/73.5$^{\pm6.5}$ | 81.1$^{\pm1.4}$/83.1$^{\pm0.6}$ |
| | Or-Ra | | 85.2$^{\pm1.8}$/76.3$^{\pm4.3}$ | **88.5**$^{\pm1.4}$/85.1$^{\pm0.5}$ | | 84.2$^{\pm1.8}$/68.4$^{\pm3.9}$ | 79.6$^{\pm2.0}$/85.5$^{\pm0.7}$ |
| | Avg. | | 86.0/81.2 | 87.3/82.0 | | 83.1/78.7 | 78.9/81.9 |

Table A.5: Cross-lingual accuracy results for linear transformations obtained on the binary IMDB-CSFD (English-Czech) dataset pair. Normalization was applied only before the transformation.

| | | Evaluated on **Czech** | | | Evaluated on **English** | | |
|---|---|---|---|---|---|---|---|
| **Norm.** | **Method** | **Monoling.** | **EN-s ⇒CS-t** | **CS-t ⇒EN-s** | **Monoling.** | **CS-s ⇒EN-t** | **EN-t ⇒CS-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| | | | | **CNN** | | | |
| - | MSE | 93.9/91.5 | 88.2$^{\pm0.3}$/75.7$^{\pm1.4}$ | 87.5$^{\pm1.8}$/80.3$^{\pm1.4}$ | 91.8/89.3 | **84.4**$^{\pm0.4}$/78.7$^{\pm0.4}$ | 64.7$^{\pm0.5}$/70.3$^{\pm0.8}$ |
| | Orto | | 88.5$^{\pm0.1}$/78.9$^{\pm0.9}$ | **89.2**$^{\pm0.1}$/78.1$^{\pm0.9}$ | | 84.0$^{\pm0.1}$/79.7$^{\pm0.1}$ | 81.3$^{\pm0.3}$/79.3$^{\pm0.8}$ |
| | CCA | | 88.4$^{\pm0.1}$/76.3$^{\pm1.1}$ | 88.2$^{\pm0.1}$/79.2$^{\pm0.6}$ | | 83.9$^{\pm0.1}$/77.4$^{\pm0.5}$ | 80.2$^{\pm0.1}$/75.0$^{\pm0.6}$ |
| | Rank | | 85.7$^{\pm0.3}$/78.9$^{\pm0.7}$ | 88.0$^{\pm0.8}$/76.7$^{\pm0.6}$ | | 83.4$^{\pm0.2}$/77.6$^{\pm0.7}$ | 82.9$^{\pm0.3}$/73.5$^{\pm0.7}$ |
| | Or-Ra | | 83.3$^{\pm0.6}$/76.9$^{\pm1.7}$ | **89.2**$^{\pm0.1}$/79.2$^{\pm1.0}$ | | 79.6$^{\pm0.5}$/78.4$^{\pm0.5}$ | 82.3$^{\pm0.4}$/75.0$^{\pm0.8}$ |
| | Avg. | | 86.8/77.3 | 88.4/78.7 | | 83.1/78.4 | 78.3/74.6 |
| B,A | MSE | 93.4/92.6 | **88.7**$^{\pm0.2}$/86.0$^{\pm0.1}$ | 86.8$^{\pm0.1}$/86.1$^{\pm0.1}$ | 91.6/91.1 | 84.3$^{\pm0.2}$/84.5$^{\pm0.1}$ | 80.9$^{\pm0.3}$/82.7$^{\pm0.4}$ |
| | Orto | | 88.0$^{\pm0.2}$/86.1$^{\pm0.1}$ | 87.2$^{\pm0.4}$/87.2$^{\pm0.4}$ | | **85.3**$^{\pm0.3}$/82.1$^{\pm1.9}$ | 82.0$^{\pm0.9}$/84.0$^{\pm0.3}$ |
| | CCA | | 88.2$^{\pm0.1}$/84.7$^{\pm0.4}$ | 86.5$^{\pm0.3}$/85.9$^{\pm0.1}$ | | 85.0$^{\pm0.1}$/84.3$^{\pm0.1}$ | 83.4$^{\pm0.2}$/83.4$^{\pm0.8}$ |
| | Rank | | 88.0$^{\pm0.2}$/85.7$^{\pm0.1}$ | 87.2$^{\pm0.1}$/85.5$^{\pm0.2}$ | | 84.9$^{\pm0.2}$/84.5$^{\pm0.1}$ | 83.9$^{\pm0.4}$/83.5$^{\pm0.8}$ |
| | Or-Ra | | 87.8$^{\pm0.2}$/86.2$^{\pm0.2}$ | 88.3$^{\pm0.2}$/83.1$^{\pm0.4}$ | | 85.2$^{\pm0.1}$/83.6$^{\pm0.8}$ | 83.6$^{\pm0.3}$/84.9$^{\pm0.4}$ |
| | Avg. | | 88.1/85.7 | 87.2/85.6 | | 84.9/83.8 | 82.8/83.7 |
| | | | | **LSTM** | | | |
| - | MSE | 94.4/92.1 | 84.9$^{\pm0.7}$/80.6$^{\pm1.3}$ | 85.9$^{\pm2.6}$/79.3$^{\pm4.3}$ | 92.5/90.7 | **85.5**$^{\pm0.7}$/83.5$^{\pm1.9}$ | 67.7$^{\pm3.7}$/75.2$^{\pm2.9}$ |
| | Orto | | 87.6$^{\pm0.4}$/80.2$^{\pm2.3}$ | **87.9**$^{\pm0.5}$/80.2$^{\pm3.0}$ | | 73.6$^{\pm1.1}$/79.8$^{\pm1.5}$ | 74.9$^{\pm3.8}$/83.3$^{\pm1.2}$ |
| | CCA | | 86.6$^{\pm1.9}$/82.9$^{\pm0.7}$ | 87.4$^{\pm0.3}$/82.7$^{\pm0.5}$ | | 83.6$^{\pm0.8}$/66.9$^{\pm3.1}$ | 81.8$^{\pm1.5}$/82.6$^{\pm0.6}$ |
| | Rank | | 84.1$^{\pm1.2}$/75.4$^{\pm0.5}$ | 86.0$^{\pm1.5}$/82.6$^{\pm1.1}$ | | 74.1$^{\pm4.5}$/69.6$^{\pm2.8}$ | 83.8$^{\pm0.8}$/83.5$^{\pm0.5}$ |
| | Or-Ra | | 86.2$^{\pm0.4}$/73.2$^{\pm1.5}$ | 86.8$^{\pm0.7}$/82.9$^{\pm2.0}$ | | 82.0$^{\pm3.4}$/71.6$^{\pm3.4}$ | 84.9$^{\pm0.7}$/83.8$^{\pm0.8}$ |
| | Avg. | | 85.9/78.5 | 86.8/81.5 | | 79.8/74.3 | 78.6/81.7 |
| B,A | MSE | 93.9/92.6 | 86.2$^{\pm0.6}$/83.9$^{\pm2.1}$ | **89.1**$^{\pm0.3}$/83.5$^{\pm1.5}$ | 92.6/91.5 | 85.4$^{\pm1.8}$/78.4$^{\pm3.3}$ | 81.3$^{\pm1.1}$/83.7$^{\pm1.6}$ |
| | Orto | | 86.6$^{\pm0.9}$/83.8$^{\pm0.9}$ | 80.9$^{\pm5.5}$/79.1$^{\pm3.1}$ | | 72.0$^{\pm6.1}$/84.7$^{\pm0.9}$ | 78.5$^{\pm2.2}$/**86.2**$^{\pm1.0}$ |
| | CCA | | 86.5$^{\pm1.0}$/84.0$^{\pm1.2}$ | 85.9$^{\pm1.4}$/83.3$^{\pm1.1}$ | | 84.2$^{\pm3.4}$/83.0$^{\pm2.5}$ | 84.1$^{\pm2.2}$/85.1$^{\pm0.4}$ |
| | Rank | | 87.2$^{\pm0.6}$/81.8$^{\pm1.5}$ | 87.9$^{\pm0.8}$/82.5$^{\pm2.3}$ | | 84.2$^{\pm2.0}$/76.3$^{\pm3.6}$ | 83.5$^{\pm1.4}$/84.8$^{\pm0.6}$ |
| | Or-Ra | | 85.8$^{\pm1.4}$/82.5$^{\pm3.4}$ | 86.7$^{\pm1.8}$/82.7$^{\pm2.0}$ | | 76.7$^{\pm3.5}$/83.6$^{\pm3.8}$ | 79.7$^{\pm2.1}$/86.0$^{\pm0.7}$ |
| | Avg. | | 86.5/83.2 | 86.1/82.2 | | 80.5/81.2 | 81.4/85.2 |

Table A.6: Cross-lingual accuracy results for linear transformations obtained on the binary IMDB-CSFD (English-Czech) dataset pair. Normalization was applied before and after the transformation.

| | | Evaluated on **Czech** | | | Evaluated on **English** | | |
|---|---|---|---|---|---|---|---|
| **Norm.** | **Method** | **Monoling.** | **EN-s ⇒CS-t** | **CS-t ⇒EN-s** | **Monoling.** | **CS-s ⇒EN-t** | **EN-t ⇒CS-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| | | | | **CNN** | | | |
| - | MSE | 93.9/91.5 | $\mathbf{86.3}^{\pm0.2}/74.5^{\pm0.5}$ | $86.0^{\pm1.4}/78.1^{\pm1.0}$ | 84.4/83.7 | $73.4^{\pm0.4}/70.3^{\pm0.9}$ | $64.7^{\pm2.5}/70.4^{\pm0.3}$ |
| | Orto | | $85.4^{\pm0.1}/77.1^{\pm0.5}$ | $84.9^{\pm1.1}/66.5^{\pm2.1}$ | | $\mathbf{77.8}^{\pm0.2}/76.0^{\pm0.2}$ | $75.3^{\pm0.5}/74.3^{\pm0.8}$ |
| | CCA | | $85.4^{\pm0.1}/76.3^{\pm0.9}$ | $85.5^{\pm0.4}/72.7^{\pm2.5}$ | | $77.6^{\pm0.2}/74.7^{\pm0.4}$ | $74.6^{\pm0.5}/73.0^{\pm0.4}$ |
| | Rank | | $85.9^{\pm0.3}/69.3^{\pm1.2}$ | $83.4^{\pm0.9}/74.8^{\pm0.5}$ | | $77.0^{\pm0.4}/73.2^{\pm0.2}$ | $77.4^{\pm0.4}/75.2^{\pm0.4}$ |
| | Or-Ra | | $82.4^{\pm0.9}/66.5^{\pm2.6}$ | $85.6^{\pm0.8}/75.4^{\pm0.3}$ | | $76.2^{\pm0.4}/75.5^{\pm0.3}$ | $77.4^{\pm0.3}/77.2^{\pm0.3}$ |
| | Avg. | | 85.1/72.7 | 85.1/73.5 | | 76.4/73.9 | 73.9/74.0 |
| B | MSE | 93.4/92.6 | $85.4^{\pm0.2}/81.3^{\pm0.8}$ | $\mathbf{86.0}^{\pm0.5}/80.1^{\pm1.0}$ | 84.6/85.4 | $74.9^{\pm0.3}/77.5^{\pm0.3}$ | $77.8^{\pm0.1}/78.4^{\pm0.4}$ |
| | Orto | | $83.0^{\pm1.3}/81.6^{\pm0.8}$ | $84.0^{\pm1.6}/82.0^{\pm0.1}$ | | $77.6^{\pm0.7}/75.4^{\pm0.5}$ | $74.7^{\pm1.1}/75.7^{\pm0.9}$ |
| | CCA | | $85.9^{\pm0.2}/83.0^{\pm0.5}$ | $83.0^{\pm0.6}/79.4^{\pm1.1}$ | | $77.8^{\pm0.2}/78.5^{\pm0.4}$ | $75.3^{\pm0.4}/77.3^{\pm0.6}$ |
| | Rank | | $85.5^{\pm0.3}/78.1^{\pm0.8}$ | $84.5^{\pm0.6}/81.7^{\pm0.6}$ | | $78.6^{\pm0.3}/78.4^{\pm0.3}$ | $75.3^{\pm0.6}/76.2^{\pm0.6}$ |
| | Or-Ra | | $85.2^{\pm0.7}/82.6^{\pm0.5}$ | $85.8^{\pm0.5}/82.7^{\pm0.2}$ | | $\mathbf{78.7}^{\pm0.2}/75.7^{\pm0.4}$ | $76.2^{\pm0.9}/77.6^{\pm0.4}$ |
| | Avg. | | 85.0/81.3 | 84.7/81.2 | | 77.5/77.1 | 75.9/77.0 |
| | | | | **LSTM** | | | |
| - | MSE | 94.4/92.1 | $\mathbf{85.3}^{\pm0.4}/73.0^{\pm0.4}$ | $82.0^{\pm2.2}/69.5^{\pm2.0}$ | 85.3/84.3 | $76.1^{\pm0.4}/78.4^{\pm0.4}$ | $72.3^{\pm2.9}/70.6^{\pm3.0}$ |
| | Orto | | $80.4^{\pm1.9}/75.5^{\pm1.1}$ | $80.1^{\pm0.9}/76.7^{\pm1.3}$ | | $72.6^{\pm1.3}/78.4^{\pm0.5}$ | $75.6^{\pm1.8}/78.7^{\pm0.5}$ |
| | CCA | | $83.0^{\pm1.3}/72.7^{\pm1.7}$ | $82.6^{\pm0.8}/72.9^{\pm1.9}$ | | $76.5^{\pm1.6}/76.9^{\pm1.5}$ | $75.0^{\pm0.6}/76.5^{\pm1.7}$ |
| | Rank | | $84.1^{\pm0.7}/71.7^{\pm1.0}$ | $76.1^{\pm0.6}/73.3^{\pm1.0}$ | | $72.9^{\pm3.1}/76.2^{\pm0.7}$ | $77.5^{\pm1.3}/79.1^{\pm0.6}$ |
| | Or-Ra | | $83.0^{\pm0.7}/73.8^{\pm1.8}$ | $82.2^{\pm1.8}/78.3^{\pm2.0}$ | | $74.7^{\pm1.5}/76.1^{\pm1.8}$ | $75.9^{\pm2.1}/\mathbf{79.5}^{\pm0.4}$ |
| | Avg. | | 83.2/73.3 | 80.6/74.1 | | 74.6/77.2 | 75.3/76.9 |
| B | MSE | 93.9/92.6 | $83.8^{\pm0.4}/81.1^{\pm0.6}$ | $85.3^{\pm0.9}/77.7^{\pm3.1}$ | 84.5/85.9 | $77.7^{\pm0.8}/78.2^{\pm1.5}$ | $74.9^{\pm1.1}/75.5^{\pm2.0}$ |
| | Orto | | $74.2^{\pm3.9}/80.2^{\pm1.3}$ | $78.4^{\pm3.7}/80.0^{\pm0.5}$ | | $76.9^{\pm0.8}/79.1^{\pm0.3}$ | $73.2^{\pm0.9}/76.9^{\pm1.4}$ |
| | CCA | | $82.2^{\pm1.4}/79.8^{\pm2.8}$ | $74.5^{\pm3.5}/78.4^{\pm1.2}$ | | $77.6^{\pm1.2}/77.1^{\pm1.1}$ | $74.4^{\pm1.4}/77.8^{\pm0.8}$ |
| | Rank | | $83.2^{\pm2.5}/79.4^{\pm1.5}$ | $84.2^{\pm0.6}/82.0^{\pm0.6}$ | | $76.8^{\pm1.0}/76.1^{\pm0.7}$ | $75.6^{\pm1.1}/79.9^{\pm0.3}$ |
| | Or-Ra | | $79.1^{\pm2.5}/74.7^{\pm3.1}$ | $\mathbf{85.8}^{\pm0.5}/81.3^{\pm0.8}$ | | $74.7^{\pm2.2}/70.5^{\pm3.6}$ | $75.5^{\pm0.9}/\mathbf{80.3}^{\pm0.4}$ |
| | Avg. | | 80.5/79.0 | 81.6/79.9 | | 76.7/76.2 | 74.7/78.1 |

Table A.7: Cross-lingual accuracy results for linear transformations obtained on the binary SST-CSFD (English-Czech) dataset pair. Normalization was applied only before the transformation.

| Norm. | Method | Evaluated on **Czech** | | | Evaluated on **English** | | |
|---|---|---|---|---|---|---|---|
| | | **Monoling.** | **EN-s ⇒CS-t** | **CS-t ⇒EN-s** | **Monoling.** | **CS-s ⇒EN-t** | **EN-t ⇒CS-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| | | | **CNN** | | | | |
| - | MSE | 93.9/91.5 | **86.3**$^{\pm0.2}$/74.5$^{\pm0.5}$ | 86.0$^{\pm1.4}$/78.1$^{\pm1.0}$ | 84.4/83.7 | 73.4$^{\pm0.4}$/70.3$^{\pm0.9}$ | 64.7$^{\pm2.5}$/70.4$^{\pm0.3}$ |
| | Orto | | 85.4$^{\pm0.1}$/77.1$^{\pm0.5}$ | 84.9$^{\pm1.1}$/66.5$^{\pm2.1}$ | | **77.8**$^{\pm0.2}$/76.0$^{\pm0.2}$ | 75.3$^{\pm0.5}$/74.3$^{\pm0.8}$ |
| | CCA | | 85.4$^{\pm0.1}$/76.3$^{\pm0.9}$ | 85.5$^{\pm0.4}$/72.7$^{\pm2.5}$ | | 77.6$^{\pm0.2}$/74.7$^{\pm0.4}$ | 74.6$^{\pm0.5}$/73.0$^{\pm0.4}$ |
| | Rank | | 85.9$^{\pm0.3}$/69.3$^{\pm1.2}$ | 83.4$^{\pm0.9}$/74.8$^{\pm0.5}$ | | 77.0$^{\pm0.4}$/73.2$^{\pm0.2}$ | 77.4$^{\pm0.4}$/75.2$^{\pm0.4}$ |
| | Or-Ra | | 82.4$^{\pm0.9}$/66.5$^{\pm2.6}$ | 85.6$^{\pm0.8}$/75.4$^{\pm0.3}$ | | 76.2$^{\pm0.4}$/75.5$^{\pm0.3}$ | 77.4$^{\pm0.3}$/77.2$^{\pm0.3}$ |
| | Avg. | | 85.1/72.7 | 85.1/73.5 | | 76.4/73.9 | 73.9/74.0 |
| B,A | MSE | 93.4/92.6 | 84.9$^{\pm0.5}$/79.9$^{\pm0.8}$ | 84.3$^{\pm0.4}$/79.8$^{\pm1.7}$ | 84.6/85.4 | 77.4$^{\pm0.3}$/**79.2**$^{\pm0.1}$ | 77.8$^{\pm0.2}$/78.2$^{\pm0.5}$ |
| | Orto | | 84.2$^{\pm0.9}$/82.2$^{\pm0.2}$ | 81.4$^{\pm0.9}$/81.6$^{\pm0.5}$ | | 77.5$^{\pm0.4}$/77.5$^{\pm0.4}$ | 76.1$^{\pm0.3}$/77.7$^{\pm0.4}$ |
| | CCA | | 85.5$^{\pm0.2}$/82.4$^{\pm0.1}$ | 84.8$^{\pm0.6}$/78.6$^{\pm1.3}$ | | 77.8$^{\pm0.1}$/**79.2**$^{\pm0.3}$ | 75.6$^{\pm0.2}$/77.2$^{\pm0.2}$ |
| | Rank | | **85.9**$^{\pm0.5}$/82.6$^{\pm0.3}$ | 82.9$^{\pm0.9}$/81.6$^{\pm0.2}$ | | 78.8$^{\pm0.3}$/79.1$^{\pm0.2}$ | 76.7$^{\pm0.4}$/76.8$^{\pm0.5}$ |
| | Or-Ra | | **85.9**$^{\pm0.3}$/82.5$^{\pm0.3}$ | 84.3$^{\pm0.6}$/81.0$^{\pm0.7}$ | | 78.8$^{\pm0.4}$/78.8$^{\pm0.3}$ | 77.5$^{\pm0.6}$/78.6$^{\pm0.2}$ |
| | Avg. | | 85.3/81.9 | 83.5/80.5 | | 78.1/78.8 | 76.7/77.7 |
| | | | **LSTM** | | | | |
| - | MSE | 94.4/92.1 | **85.3**$^{\pm0.4}$/73.0$^{\pm0.4}$ | 82.0$^{\pm2.2}$/69.5$^{\pm2.0}$ | 85.3/84.3 | 76.1$^{\pm0.4}$/78.4$^{\pm0.4}$ | 72.3$^{\pm2.9}$/70.6$^{\pm3.0}$ |
| | Orto | | 80.4$^{\pm1.9}$/75.5$^{\pm1.1}$ | 80.1$^{\pm0.9}$/76.7$^{\pm1.3}$ | | 72.6$^{\pm1.3}$/78.4$^{\pm0.5}$ | 75.6$^{\pm1.8}$/78.7$^{\pm0.5}$ |
| | CCA | | 83.0$^{\pm1.3}$/72.7$^{\pm1.7}$ | 82.6$^{\pm0.8}$/72.9$^{\pm1.9}$ | | 76.5$^{\pm1.6}$/76.9$^{\pm1.5}$ | 75.0$^{\pm0.6}$/76.5$^{\pm1.7}$ |
| | Rank | | 84.1$^{\pm0.7}$/71.7$^{\pm1.0}$ | 76.1$^{\pm0.6}$/73.3$^{\pm1.0}$ | | 72.9$^{\pm3.1}$/76.2$^{\pm0.7}$ | 77.5$^{\pm1.3}$/79.1$^{\pm0.6}$ |
| | Or-Ra | | 83.0$^{\pm0.7}$/73.8$^{\pm1.8}$ | 82.2$^{\pm1.8}$/78.3$^{\pm2.0}$ | | 74.7$^{\pm1.5}$/76.1$^{\pm1.8}$ | 75.9$^{\pm2.1}$/**79.5**$^{\pm0.4}$ |
| | Avg. | | 83.2/73.3 | 80.6/74.1 | | 74.6/77.2 | 75.3/76.9 |
| B,A | MSE | 93.9/92.6 | 79.6$^{\pm3.0}$/77.7$^{\pm2.6}$ | **86.7**$^{\pm0.9}$/80.3$^{\pm1.0}$ | 84.5/85.9 | 76.0$^{\pm1.1}$/77.2$^{\pm2.3}$ | 78.2$^{\pm0.4}$/77.4$^{\pm1.6}$ |
| | Orto | | 79.6$^{\pm3.7}$/79.7$^{\pm2.6}$ | 80.1$^{\pm4.0}$/78.6$^{\pm4.3}$ | | 78.3$^{\pm1.1}$/**79.6**$^{\pm1.2}$ | 73.2$^{\pm3.2}$/79.1$^{\pm0.7}$ |
| | CCA | | 83.6$^{\pm1.5}$/77.5$^{\pm5.1}$ | 80.3$^{\pm2.5}$/80.7$^{\pm0.9}$ | | 77.8$^{\pm0.7}$/77.5$^{\pm2.1}$ | 75.9$^{\pm0.6}$/77.3$^{\pm1.2}$ |
| | Rank | | 84.5$^{\pm1.0}$/79.2$^{\pm3.1}$ | 84.3$^{\pm0.7}$/80.9$^{\pm1.0}$ | | 76.5$^{\pm0.6}$/73.7$^{\pm3.0}$ | 78.0$^{\pm0.7}$/77.4$^{\pm1.3}$ |
| | Or-Ra | | 82.8$^{\pm1.4}$/79.7$^{\pm1.5}$ | 81.1$^{\pm2.7}$/77.5$^{\pm2.0}$ | | 76.7$^{\pm0.9}$/79.1$^{\pm1.2}$ | 78.5$^{\pm0.4}$/78.5$^{\pm0.4}$ |
| | Avg. | | 82.0/78.8 | 82.5/79.6 | | 77.1/77.4 | 76.8/77.9 |

Table A.8: Cross-lingual accuracy results for linear transformations obtained on the binary SST-CSFD (English-Czech) dataset pair. Normalization was applied before and after the transformation.

| | | Evaluated on **Czech** | | | Evaluated on **English** | | |
|---|---|---|---|---|---|---|---|
| **Norm.** | **Method** | **Monoling.** | **EN-s ⇒CS-t** | **CS-t ⇒EN-s** | **Monoling.** | **CS-s ⇒EN-t** | **EN-t ⇒CS-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| | | | | **CNN** | | | |
| - | MSE | 83.7/80.3 | 57.8$^{\pm0.0}$/43.8$^{\pm0.0}$ | 56.6$^{\pm5.7}$/44.4$^{\pm1.2}$ | 66.4/66.1 | 45.0$^{\pm0.0}$/45.7$^{\pm0.0}$ | 34.6$^{\pm0.0}$/53.3$^{\pm0.0}$ |
| | Orto | | 55.7$^{\pm1.9}$/49.8$^{\pm2.1}$ | 53.9$^{\pm4.3}$/47.2$^{\pm0.4}$ | | 48.1$^{\pm0.1}$/50.8$^{\pm0.7}$ | 48.1$^{\pm0.0}$/57.0$^{\pm0.0}$ |
| | CCA | | **58.2**$^{\pm1.7}$/49.5$^{\pm1.0}$ | 55.3$^{\pm1.4}$/47.7$^{\pm1.3}$ | | 47.2$^{\pm0.5}$/48.0$^{\pm0.5}$ | 48.5$^{\pm0.0}$/54.3$^{\pm0.0}$ |
| | Rank | | 55.8$^{\pm0.0}$/47.4$^{\pm0.0}$ | 55.8$^{\pm0.0}$/35.8$^{\pm0.0}$ | | 47.4$^{\pm0.7}$/50.6$^{\pm1.0}$ | 59.3$^{\pm2.8}$/46.9$^{\pm0.2}$ |
| | Or-Ra | | 51.1$^{\pm0.0}$/47.3$^{\pm0.0}$ | 56.9$^{\pm2.0}$/49.2$^{\pm1.0}$ | | 43.4$^{\pm1.0}$/47.5$^{\pm1.0}$ | **61.4**$^{\pm1.1}$/51.2$^{\pm1.5}$ |
| | Avg. | | 55.7/47.6 | 55.7/44.9 | | 46.2/48.5 | 50.4/52.5 |
| B | MSE | 82.9/81.7 | 57.3$^{\pm0.9}$/55.0$^{\pm0.7}$ | 55.7$^{\pm1.4}$/52.7$^{\pm2.3}$ | 68.5/68.6 | 49.8$^{\pm0.2}$/49.6$^{\pm0.3}$ | 50.3$^{\pm0.3}$/**55.0**$^{\pm0.5}$ |
| | Orto | | 56.4$^{\pm0.9}$/55.0$^{\pm0.3}$ | 56.8$^{\pm1.4}$/52.6$^{\pm0.2}$ | | 51.2$^{\pm1.1}$/48.6$^{\pm1.3}$ | 52.7$^{\pm0.2}$/53.9$^{\pm0.3}$ |
| | CCA | | **59.7**$^{\pm0.5}$/52.8$^{\pm0.5}$ | 57.3$^{\pm1.4}$/53.1$^{\pm0.6}$ | | 48.3$^{\pm0.5}$/50.7$^{\pm0.5}$ | 50.2$^{\pm0.3}$/53.4$^{\pm0.3}$ |
| | Rank | | 58.4$^{\pm1.4}$/52.3$^{\pm0.5}$ | 56.2$^{\pm2.1}$/54.5$^{\pm0.4}$ | | 47.7$^{\pm0.3}$/50.7$^{\pm0.6}$ | 52.4$^{\pm0.3}$/52.1$^{\pm0.2}$ |
| | Or-Ra | | 54.0$^{\pm1.8}$/47.7$^{\pm2.7}$ | **59.7**$^{\pm1.3}$/55.4$^{\pm0.3}$ | | 47.6$^{\pm0.3}$/50.8$^{\pm0.4}$ | 51.6$^{\pm0.0}$/50.6$^{\pm0.0}$ |
| | Avg. | | 57.2/52.6 | 57.1/53.7 | | 48.9/50.1 | 51.4/53.0 |
| | | | | **LSTM** | | | |
| - | MSE | 84.8/81.8 | 54.0$^{\pm0.0}$/53.4$^{\pm0.0}$ | 47.5$^{\pm0.0}$/37.5$^{\pm0.0}$ | 69.7/70.4 | 48.0$^{\pm0.6}$/47.4$^{\pm1.1}$ | 39.5$^{\pm0.9}$/48.3$^{\pm0.5}$ |
| | Orto | | 53.2$^{\pm0.7}$/46.6$^{\pm0.5}$ | 51.4$^{\pm0.0}$/44.4$^{\pm0.0}$ | | 46.6$^{\pm0.6}$/49.9$^{\pm1.5}$ | 45.5$^{\pm0.0}$/52.2$^{\pm0.0}$ |
| | CCA | | 51.4$^{\pm0.0}$/50.6$^{\pm0.0}$ | 49.9$^{\pm0.0}$/50.7$^{\pm0.0}$ | | 46.1$^{\pm0.0}$/41.1$^{\pm0.0}$ | 44.9$^{\pm0.0}$/46.1$^{\pm0.0}$ |
| | Rank | | 55.2$^{\pm0.4}$/47.2$^{\pm0.8}$ | **55.3**$^{\pm2.0}$/32.9$^{\pm1.4}$ | | 41.8$^{\pm4.0}$/44.9$^{\pm1.6}$ | **59.4**$^{\pm0.0}$/51.7$^{\pm0.0}$ |
| | Or-Ra | | 54.3$^{\pm0.0}$/42.5$^{\pm0.0}$ | 54.2$^{\pm2.3}$/34.7$^{\pm2.0}$ | | 41.1$^{\pm5.0}$/42.9$^{\pm2.7}$ | 55.4$^{\pm0.0}$/53.3$^{\pm0.0}$ |
| | Avg. | | 53.6/48.1 | 51.7/40.0 | | 44.7/45.2 | 48.9/50.3 |
| B | MSE | 84.2/82.8 | **55.7**$^{\pm0.4}$/**55.7**$^{\pm0.8}$ | 54.4$^{\pm2.6}$/54.4$^{\pm1.2}$ | 68.2/71.3 | 51.4$^{\pm0.0}$/48.4$^{\pm0.0}$ | 46.8$^{\pm0.0}$/51.1$^{\pm0.0}$ |
| | Orto | | 53.5$^{\pm2.7}$/52.8$^{\pm1.8}$ | 43.4$^{\pm4.3}$/53.9$^{\pm1.8}$ | | **56.5**$^{\pm0.7}$/49.7$^{\pm1.1}$ | 50.1$^{\pm1.6}$/47.5$^{\pm1.3}$ |
| | CCA | | 53.6$^{\pm2.7}$/53.9$^{\pm1.6}$ | 53.5$^{\pm0.7}$/51.4$^{\pm2.4}$ | | 53.8$^{\pm0.0}$/46.5$^{\pm0.0}$ | 49.3$^{\pm1.1}$/50.2$^{\pm1.2}$ |
| | Rank | | 54.2$^{\pm3.9}$/52.5$^{\pm1.2}$ | 52.5$^{\pm1.0}$/53.7$^{\pm0.9}$ | | 48.5$^{\pm1.2}$/40.1$^{\pm1.3}$ | 52.0$^{\pm0.6}$/55.5$^{\pm0.7}$ |
| | Or-Ra | | 47.7$^{\pm3.0}$/52.6$^{\pm1.2}$ | 54.2$^{\pm1.3}$/54.2$^{\pm0.7}$ | | 45.8$^{\pm1.4}$/43.1$^{\pm1.6}$ | 54.3$^{\pm0.7}$/50.0$^{\pm0.6}$ |
| | Avg. | | 52.9/53.5 | 51.6/53.5 | | 51.2/45.6 | 50.5/50.9 |

Table A.9: Cross-lingual accuracy results for linear transformations obtained on the three class SST-CSFD (English-Czech) dataset pair. Normalization was applied only before the transformation.

|  |  | Evaluated on **Czech** |  |  | Evaluated on **English** |  |  |
|---|---|---|---|---|---|---|---|
| **Norm.** | **Method** | **Monoling.** | **EN-s ⇒CS-t** | **CS-t ⇒EN-s** | **Monoling.** | **CS-s ⇒EN-t** | **EN-t ⇒CS-s** |
|  |  |  | in-domain/fastText | in-domain/fastText |  | in-domain/fastText | in-domain/fastText |
|  |  |  |  | **CNN** |  |  |  |
| - | MSE | 83.7/80.3 | 57.8$^{\pm0.0}$/43.8$^{\pm0.0}$ | 56.6$^{\pm5.7}$/44.4$^{\pm1.2}$ | 66.4/66.1 | 45.0$^{\pm0.0}$/45.7$^{\pm0.0}$ | 34.6$^{\pm0.0}$/53.3$^{\pm0.0}$ |
|  | Orto |  | 55.7$^{\pm1.9}$/49.8$^{\pm2.1}$ | 53.9$^{\pm4.3}$/47.2$^{\pm0.4}$ |  | 48.1$^{\pm0.1}$/50.8$^{\pm0.7}$ | 48.1$^{\pm0.0}$/57.0$^{\pm0.0}$ |
|  | CCA |  | **58.2**$^{\pm1.7}$/49.5$^{\pm1.0}$ | 55.3$^{\pm1.4}$/47.7$^{\pm1.3}$ |  | 47.2$^{\pm0.5}$/48.0$^{\pm0.5}$ | 48.5$^{\pm0.0}$/54.3$^{\pm0.0}$ |
|  | Rank |  | 55.8$^{\pm0.0}$/47.4$^{\pm0.0}$ | 55.8$^{\pm0.0}$/35.8$^{\pm0.0}$ |  | 47.4$^{\pm0.7}$/50.6$^{\pm1.0}$ | 59.3$^{\pm2.8}$/46.9$^{\pm0.2}$ |
|  | Or-Ra |  | 51.1$^{\pm0.0}$/47.3$^{\pm0.0}$ | 56.9$^{\pm2.0}$/49.2$^{\pm1.0}$ |  | 43.4$^{\pm1.0}$/47.5$^{\pm1.0}$ | **61.4**$^{\pm1.1}$/51.2$^{\pm1.5}$ |
|  | Avg. |  | 55.7/47.6 | 55.7/44.9 |  | 46.2/48.5 | 50.4/52.5 |
| B,A | MSE | 82.9/81.7 | **58.7**$^{\pm1.3}$/54.3$^{\pm0.7}$ | 56.9$^{\pm0.2}$/54.0$^{\pm0.6}$ | 68.5/68.6 | 46.5$^{\pm0.3}$/53.9$^{\pm0.3}$ | 52.1$^{\pm0.4}$/**56.2**$^{\pm0.2}$ |
|  | Orto |  | 55.7$^{\pm1.6}$/54.6$^{\pm0.4}$ | 54.7$^{\pm0.7}$/55.2$^{\pm0.3}$ |  | 54.3$^{\pm0.3}$/54.7$^{\pm0.6}$ | 53.5$^{\pm0.2}$/55.8$^{\pm0.5}$ |
|  | CCA |  | 58.0$^{\pm0.5}$/52.5$^{\pm0.7}$ | 55.6$^{\pm1.3}$/49.4$^{\pm1.0}$ |  | 46.0$^{\pm1.7}$/51.9$^{\pm1.2}$ | 47.5$^{\pm1.4}$/49.2$^{\pm1.2}$ |
|  | Rank |  | 57.9$^{\pm0.4}$/51.8$^{\pm0.4}$ | 55.3$^{\pm0.7}$/54.9$^{\pm0.7}$ |  | 50.8$^{\pm0.5}$/55.9$^{\pm0.3}$ | 50.6$^{\pm0.1}$/53.9$^{\pm0.5}$ |
|  | Or-Ra |  | 57.0$^{\pm0.3}$/55.3$^{\pm0.2}$ | 55.9$^{\pm0.7}$/55.6$^{\pm0.6}$ |  | 51.1$^{\pm0.2}$/54.1$^{\pm0.4}$ | 53.1$^{\pm0.2}$/52.1$^{\pm0.6}$ |
|  | Avg. |  | 57.5/53.7 | 55.7/53.8 |  | 49.7/54.1 | 51.4/53.4 |
|  |  |  |  | **LSTM** |  |  |  |
| - | MSE | 84.8/81.8 | 54.0$^{\pm0.0}$/53.4$^{\pm0.0}$ | 47.5$^{\pm0.0}$/37.5$^{\pm0.0}$ | 69.7/70.4 | 48.0$^{\pm0.6}$/47.4$^{\pm1.1}$ | 39.5$^{\pm0.9}$/48.3$^{\pm0.5}$ |
|  | Orto |  | 53.2$^{\pm0.7}$/46.6$^{\pm0.5}$ | 51.4$^{\pm0.0}$/44.4$^{\pm0.0}$ |  | 46.6$^{\pm0.6}$/49.9$^{\pm1.5}$ | 45.5$^{\pm0.0}$/52.2$^{\pm0.0}$ |
|  | CCA |  | 51.4$^{\pm0.0}$/50.6$^{\pm0.0}$ | 49.9$^{\pm0.0}$/50.7$^{\pm0.0}$ |  | 46.1$^{\pm0.0}$/41.1$^{\pm0.0}$ | 44.9$^{\pm0.0}$/46.1$^{\pm0.0}$ |
|  | Rank |  | 55.2$^{\pm0.4}$/47.2$^{\pm0.8}$ | **55.3**$^{\pm2.0}$/32.9$^{\pm1.4}$ |  | 41.8$^{\pm4.0}$/44.9$^{\pm1.6}$ | **59.4**$^{\pm0.0}$/51.7$^{\pm0.0}$ |
|  | Or-Ra |  | 54.3$^{\pm0.0}$/42.5$^{\pm0.0}$ | 54.2$^{\pm2.3}$/34.7$^{\pm2.0}$ |  | 41.1$^{\pm5.0}$/42.9$^{\pm2.7}$ | 55.4$^{\pm0.0}$/53.3$^{\pm0.0}$ |
|  | Avg. |  | 53.6/48.1 | 51.7/40.0 |  | 44.7/45.2 | 48.9/50.3 |
| B,A | MSE | 84.2/82.8 | 52.6$^{\pm2.3}$/52.1$^{\pm1.6}$ | **57.2**$^{\pm0.4}$/53.8$^{\pm0.7}$ | 68.2/71.3 | 40.9$^{\pm1.5}$/47.2$^{\pm2.4}$ | **57.5**$^{\pm1.3}$/54.0$^{\pm1.7}$ |
|  | Orto |  | 51.0$^{\pm1.7}$/53.9$^{\pm0.6}$ | 43.9$^{\pm2.7}$/52.5$^{\pm1.5}$ |  | 57.1$^{\pm1.1}$/44.1$^{\pm1.6}$ | 52.8$^{\pm1.5}$/53.0$^{\pm1.2}$ |
|  | CCA |  | 53.8$^{\pm1.3}$/53.6$^{\pm1.0}$ | 46.8$^{\pm1.9}$/48.0$^{\pm1.9}$ |  | 52.3$^{\pm1.0}$/53.6$^{\pm1.2}$ | 49.4$^{\pm1.2}$/52.0$^{\pm1.1}$ |
|  | Rank |  | 52.8$^{\pm2.4}$/54.5$^{\pm0.4}$ | 52.9$^{\pm1.5}$/54.6$^{\pm0.8}$ |  | 50.6$^{\pm1.3}$/50.9$^{\pm2.0}$ | 48.4$^{\pm0.8}$/49.4$^{\pm1.5}$ |
|  | Or-Ra |  | 52.6$^{\pm2.0}$/54.1$^{\pm1.0}$ | 55.3$^{\pm1.1}$/52.3$^{\pm1.9}$ |  | 46.5$^{\pm1.0}$/43.6$^{\pm2.7}$ | 53.6$^{\pm0.6}$/49.8$^{\pm1.5}$ |
|  | Avg. |  | 52.6/53.6 | 51.2/52.2 |  | 49.5/47.9 | 52.3/51.6 |

Table A.10: Cross-lingual accuracy results for linear transformations obtained on the three class SST-CSFD (English-Czech) dataset pair. Normalization was applied before and after the transformation.

| | | Evaluated on **French** | | | Evaluated on **English** | | |
|---|---|---|---|---|---|---|---|
| **Norm.** | **Method** | **Monoling.** | **EN-s ⇒FR-t** | **FR-t ⇒EN-s** | **Monoling.** | **FR-s ⇒EN-t** | **EN-t ⇒FR-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| | | | | **CNN** | | | |
| - | MSE | 95.0/94.3 | 87.5$^{\pm0.6}$/78.0$^{\pm0.8}$ | 74.3$^{\pm4.3}$/73.5$^{\pm3.4}$ | 91.8/89.3 | 86.2$^{\pm0.1}$/78.2$^{\pm0.2}$ | 56.1$^{\pm1.5}$/73.5$^{\pm1.6}$ |
| | Orto | | **90.4**$^{\pm0.1}$/81.0$^{\pm0.7}$ | 89.4$^{\pm0.1}$/79.6$^{\pm0.9}$ | | 86.0$^{\pm0.1}$/81.3$^{\pm0.4}$ | **87.0**$^{\pm0.1}$/81.0$^{\pm0.6}$ |
| | CCA | | 89.9$^{\pm0.1}$/81.0$^{\pm0.5}$ | 89.1$^{\pm0.1}$/81.9$^{\pm0.1}$ | | 84.6$^{\pm0.3}$/80.2$^{\pm0.4}$ | 85.3$^{\pm0.3}$/79.2$^{\pm0.4}$ |
| | Rank | | 88.5$^{\pm0.7}$/68.2$^{\pm1.6}$ | 88.7$^{\pm0.1}$/80.9$^{\pm0.5}$ | | 83.6$^{\pm0.2}$/74.5$^{\pm0.6}$ | 85.3$^{\pm0.5}$/74.9$^{\pm0.9}$ |
| | Or-Ra | | 89.2$^{\pm0.3}$/75.9$^{\pm0.8}$ | 89.4$^{\pm0.0}$/80.8$^{\pm0.6}$ | | 81.0$^{\pm0.8}$/78.3$^{\pm1.3}$ | 86.3$^{\pm0.2}$/76.2$^{\pm0.7}$ |
| | Avg. | | 89.1/76.8 | 86.2/79.3 | | 84.3/78.5 | 80.0/77.0 |
| B | MSE | 95.1/94.7 | **90.4**$^{\pm0.1}$/86.7$^{\pm0.5}$ | 85.0$^{\pm0.8}$/82.9$^{\pm0.5}$ | 91.6/91.1 | 86.4$^{\pm0.5}$/85.1$^{\pm0.6}$ | 76.1$^{\pm0.8}$/81.2$^{\pm0.7}$ |
| | Orto | | 89.9$^{\pm0.1}$/86.6$^{\pm0.1}$ | 89.7$^{\pm0.1}$/87.1$^{\pm0.1}$ | | 85.9$^{\pm0.3}$/84.8$^{\pm0.6}$ | 86.3$^{\pm0.3}$/83.9$^{\pm0.4}$ |
| | CCA | | 89.4$^{\pm0.1}$/87.1$^{\pm0.1}$ | 89.2$^{\pm0.1}$/86.0$^{\pm0.2}$ | | 86.9$^{\pm0.1}$/85.6$^{\pm0.4}$ | 84.0$^{\pm0.4}$/85.4$^{\pm0.3}$ |
| | Rank | | 90.2$^{\pm0.2}$/83.2$^{\pm1.2}$ | 88.9$^{\pm0.0}$/87.1$^{\pm0.1}$ | | **87.1**$^{\pm0.1}$/85.3$^{\pm0.3}$ | 85.1$^{\pm0.3}$/81.9$^{\pm0.8}$ |
| | Or-Ra | | **90.4**$^{\pm0.1}$/84.1$^{\pm0.6}$ | 89.3$^{\pm0.1}$/87.3$^{\pm0.1}$ | | 86.2$^{\pm0.2}$/84.8$^{\pm0.4}$ | 86.8$^{\pm0.2}$/82.7$^{\pm0.3}$ |
| | Avg. | | 90.1/85.5 | 88.4/86.1 | | 86.5/85.1 | 83.7/83.0 |
| | | | | **LSTM** | | | |
| - | MSE | 96.4/95.7 | 84.9$^{\pm1.2}$/86.0$^{\pm1.5}$ | 85.6$^{\pm1.4}$/67.2$^{\pm0.9}$ | 92.5/90.7 | 86.9$^{\pm0.5}$/83.2$^{\pm2.4}$ | 79.3$^{\pm2.6}$/78.4$^{\pm1.1}$ |
| | Orto | | 91.5$^{\pm0.5}$/79.4$^{\pm2.8}$ | 88.8$^{\pm1.0}$/86.3$^{\pm1.2}$ | | 82.7$^{\pm3.4}$/86.1$^{\pm1.4}$ | 88.2$^{\pm0.8}$/83.4$^{\pm3.3}$ |
| | CCA | | **91.9**$^{\pm0.2}$/76.7$^{\pm9.2}$ | 90.7$^{\pm0.4}$/86.6$^{\pm0.9}$ | | 85.2$^{\pm0.9}$/82.2$^{\pm0.2}$ | **88.9**$^{\pm0.4}$/85.1$^{\pm0.9}$ |
| | Rank | | 89.7$^{\pm1.6}$/84.5$^{\pm4.2}$ | 88.7$^{\pm1.3}$/84.8$^{\pm1.6}$ | | 62.3$^{\pm6.6}$/83.1$^{\pm2.2}$ | 88.6$^{\pm0.7}$/87.3$^{\pm1.4}$ |
| | Or-Ra | | 90.6$^{\pm1.1}$/79.2$^{\pm4.1}$ | 90.2$^{\pm1.3}$/87.9$^{\pm1.0}$ | | 79.4$^{\pm4.4}$/80.8$^{\pm3.2}$ | 88.5$^{\pm0.2}$/83.3$^{\pm0.4}$ |
| | Avg. | | 89.7/81.2 | 88.8/82.6 | | 79.3/83.1 | 86.7/83.5 |
| B | MSE | 96.4/95.9 | **91.5**$^{\pm0.3}$/89.3$^{\pm0.4}$ | 85.9$^{\pm3.0}$/72.7$^{\pm4.1}$ | 92.6/91.5 | 88.3$^{\pm0.7}$/88.3$^{\pm0.9}$ | 84.3$^{\pm2.9}$/84.0$^{\pm2.4}$ |
| | Orto | | 90.9$^{\pm0.6}$/81.5$^{\pm4.2}$ | 91.0$^{\pm0.6}$/89.3$^{\pm0.4}$ | | 89.1$^{\pm0.5}$/86.8$^{\pm0.3}$ | 89.3$^{\pm0.4}$/87.2$^{\pm1.2}$ |
| | CCA | | 90.3$^{\pm0.5}$/88.5$^{\pm1.0}$ | 90.5$^{\pm0.5}$/78.4$^{\pm3.3}$ | | 88.4$^{\pm1.5}$/88.4$^{\pm0.7}$ | 89.7$^{\pm0.9}$/88.6$^{\pm0.9}$ |
| | Rank | | 89.4$^{\pm1.2}$/83.6$^{\pm1.3}$ | 91.1$^{\pm0.3}$/89.8$^{\pm0.4}$ | | 79.9$^{\pm3.6}$/86.2$^{\pm1.1}$ | 87.4$^{\pm0.9}$/85.8$^{\pm2.2}$ |
| | Or-Ra | | 90.2$^{\pm1.9}$/52.8$^{\pm3.0}$ | 90.9$^{\pm1.0}$/89.6$^{\pm0.5}$ | | 82.3$^{\pm3.4}$/78.1$^{\pm5.7}$ | **90.0**$^{\pm0.4}$/82.3$^{\pm1.8}$ |
| | Avg. | | 90.5/79.1 | 89.9/84.0 | | 85.6/85.6 | 88.1/85.6 |

Table A.11: Cross-lingual accuracy results for linear transformations obtained on the binary IMDB-Allocine (English-French) dataset pair. Normalization was applied only before the transformation.

| Norm. | Method | Evaluated on **French** | | | Evaluated on **English** | | |
|---|---|---|---|---|---|---|---|
| | | **Monoling.** | **EN-s ⇒FR-t** | **FR-t ⇒EN-s** | **Monoling.** | **FR-s ⇒EN-t** | **EN-t ⇒FR-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| | | | | **CNN** | | | |
| - | MSE | 95.0/94.3 | $87.5^{\pm0.6}/78.0^{\pm0.8}$ | $74.3^{\pm4.3}/73.5^{\pm3.4}$ | 91.8/89.3 | $86.2^{\pm0.1}/78.2^{\pm0.2}$ | $56.1^{\pm1.5}/73.5^{\pm1.6}$ |
| | Orto | | $\underline{\mathbf{90.4}}^{\pm0.1}/81.0^{\pm0.7}$ | $\underline{89.4}^{\pm0.1}/79.6^{\pm0.9}$ | | $\underline{86.0}^{\pm0.1}/81.3^{\pm0.4}$ | $\mathbf{87.0}^{\pm0.1}/81.0^{\pm0.6}$ |
| | CCA | | $89.9^{\pm0.1}/81.0^{\pm0.5}$ | $\underline{89.1}^{\pm0.1}/81.9^{\pm0.1}$ | | $84.6^{\pm0.3}/80.2^{\pm0.4}$ | $85.3^{\pm0.3}/79.2^{\pm0.4}$ |
| | Rank | | $88.5^{\pm0.7}/68.2^{\pm1.6}$ | $88.7^{\pm0.1}/80.9^{\pm0.5}$ | | $83.6^{\pm0.2}/74.5^{\pm0.6}$ | $85.3^{\pm0.5}/74.9^{\pm0.9}$ |
| | Or-Ra | | $89.2^{\pm0.3}/75.9^{\pm0.8}$ | $89.4^{\pm0.0}/80.8^{\pm0.6}$ | | $81.0^{\pm0.8}/78.3^{\pm1.3}$ | $86.3^{\pm0.2}/76.2^{\pm0.7}$ |
| | Avg. | | 89.1/76.8 | 86.2/79.3 | | 84.3/78.5 | 80.0/77.0 |
| B,A | MSE | 95.1/94.7 | $\underline{\mathbf{91.2}}^{\pm0.1}/\underline{86.0}^{\pm0.9}$ | $\underline{87.6}^{\pm0.1}/\underline{87.2}^{\pm0.2}$ | 91.6/91.1 | $\underline{86.6}^{\pm0.1}/\underline{86.4}^{\pm1.0}$ | $\underline{83.9}^{\pm0.3}/\underline{86.4}^{\pm0.1}$ |
| | Orto | | $90.3^{\pm0.1}/\underline{87.4}^{\pm0.1}$ | $88.9^{\pm0.1}/\underline{86.7}^{\pm0.2}$ | | $80.7^{\pm2.8}/\underline{85.7}^{\pm1.1}$ | $\underline{\mathbf{88.1}}^{\pm0.2}/\underline{86.9}^{\pm0.2}$ |
| | CCA | | $\underline{90.0}^{\pm0.2}/\underline{86.2}^{\pm0.5}$ | $88.9^{\pm0.2}/\underline{87.2}^{\pm0.1}$ | | $\underline{84.9}^{\pm0.3}/\underline{87.3}^{\pm0.1}$ | $\underline{87.2}^{\pm0.1}/\underline{82.5}^{\pm0.5}$ |
| | Rank | | $\underline{90.5}^{\pm0.2}/\underline{85.3}^{\pm0.4}$ | $\underline{89.0}^{\pm0.1}/\underline{87.8}^{\pm0.2}$ | | $\underline{83.8}^{\pm1.2}/\underline{86.5}^{\pm0.2}$ | $\underline{86.5}^{\pm0.2}/\underline{86.4}^{\pm0.1}$ |
| | Or-Ra | | $\underline{90.6}^{\pm0.1}/\underline{87.2}^{\pm0.2}$ | $\underline{89.6}^{\pm0.1}/\underline{86.8}^{\pm0.2}$ | | $\underline{82.9}^{\pm1.3}/\underline{86.2}^{\pm0.3}$ | $\underline{88.0}^{\pm0.2}/\underline{86.1}^{\pm0.5}$ |
| | Avg. | | $\underline{90.5}/\underline{86.4}$ | $\underline{88.8}/\underline{87.1}$ | | $\underline{83.8}/\underline{86.4}$ | $\underline{86.7}/\underline{85.7}$ |
| | | | | **LSTM** | | | |
| - | MSE | 96.4/95.7 | $84.9^{\pm1.2}/86.0^{\pm1.5}$ | $85.6^{\pm1.4}/67.2^{\pm0.9}$ | 92.5/90.7 | $86.9^{\pm0.5}/83.2^{\pm2.4}$ | $79.3^{\pm2.6}/78.4^{\pm1.1}$ |
| | Orto | | $91.5^{\pm0.5}/79.4^{\pm2.8}$ | $\underline{88.8}^{\pm1.0}/86.3^{\pm1.2}$ | | $82.7^{\pm3.4}/86.1^{\pm1.4}$ | $88.2^{\pm0.8}/83.4^{\pm3.3}$ |
| | CCA | | $\underline{\mathbf{91.9}}^{\pm0.2}/76.7^{\pm9.2}$ | $\underline{90.7}^{\pm0.4}/\underline{86.6}^{\pm0.9}$ | | $85.2^{\pm0.9}/82.2^{\pm0.2}$ | $\mathbf{88.9}^{\pm0.4}/\underline{85.1}^{\pm0.9}$ |
| | Rank | | $\underline{89.7}^{\pm1.6}/\underline{84.5}^{\pm4.2}$ | $88.7^{\pm1.3}/84.8^{\pm1.6}$ | | $62.3^{\pm6.6}/83.1^{\pm2.2}$ | $88.6^{\pm0.7}/87.3^{\pm1.4}$ |
| | Or-Ra | | $90.6^{\pm1.1}/79.2^{\pm4.1}$ | $\underline{90.2}^{\pm1.3}/\underline{87.9}^{\pm1.0}$ | | $79.4^{\pm4.4}/\underline{80.8}^{\pm3.2}$ | $88.5^{\pm0.2}/\underline{83.3}^{\pm0.4}$ |
| | Avg. | | $89.7/\underline{81.2}$ | 88.8/82.6 | | 79.3/83.1 | 86.7/83.5 |
| B,A | MSE | 96.4/95.9 | $\underline{91.4}^{\pm0.9}/\underline{87.4}^{\pm2.8}$ | $\underline{89.3}^{\pm0.7}/\underline{88.2}^{\pm1.1}$ | 92.6/91.5 | $\underline{88.3}^{\pm0.7}/\underline{88.3}^{\pm0.9}$ | $\underline{88.3}^{\pm0.6}/\underline{88.6}^{\pm0.5}$ |
| | Orto | | $\underline{\mathbf{92.1}}^{\pm0.3}/\underline{89.1}^{\pm1.4}$ | $86.7^{\pm4.4}/\underline{88.6}^{\pm1.3}$ | | $\underline{89.1}^{\pm0.5}/\underline{86.8}^{\pm0.3}$ | $\underline{89.4}^{\pm0.4}/\underline{87.2}^{\pm3.3}$ |
| | CCA | | $91.6^{\pm0.7}/\underline{85.8}^{\pm2.7}$ | $87.4^{\pm1.8}/86.0^{\pm5.5}$ | | $\underline{88.4}^{\pm1.5}/\underline{88.4}^{\pm0.7}$ | $\underline{\mathbf{90.1}}^{\pm0.5}/83.7^{\pm3.2}$ |
| | Rank | | $89.6^{\pm1.4}/83.6^{\pm1.2}$ | $\underline{89.3}^{\pm1.5}/\underline{87.0}^{\pm3.8}$ | | $\underline{79.9}^{\pm3.6}/\underline{86.2}^{\pm1.1}$ | $\underline{89.4}^{\pm0.3}/87.3^{\pm1.1}$ |
| | Or-Ra | | $\underline{91.1}^{\pm0.7}/\underline{84.0}^{\pm4.7}$ | $89.3^{\pm2.0}/87.8^{\pm1.5}$ | | $\underline{82.3}^{\pm3.4}/78.1^{\pm5.7}$ | $\underline{89.5}^{\pm0.4}/83.2^{\pm4.5}$ |
| | Avg. | | $\underline{91.2}/86.0$ | $\underline{88.4}/\underline{87.5}$ | | $\underline{81.2}/\underline{88.4}$ | $\underline{89.3}/\underline{86.0}$ |

Table A.12: Cross-lingual accuracy results for linear transformations obtained on the binary IMDB-Allocine (English-French) dataset pair. Normalization was applied before and after the transformation.

| | | Evaluated on **French** | | | Evaluated on **English** | | |
|---|---|---|---|---|---|---|---|
| **Norm.** | **Method** | **Monoling.** | **EN-s ⇒FR-t** | **FR-t ⇒EN-s** | **Monoling.** | **FR-s ⇒EN-t** | **EN-t ⇒FR-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| | | | | **CNN** | | | |
| - | MSE | 95.0/94.3 | 86.7$^{\pm0.9}$/67.9$^{\pm1.4}$ | 84.5$^{\pm0.2}$/68.4$^{\pm0.6}$ | 84.4/83.7 | 79.6$^{\pm0.2}$/79.2$^{\pm0.3}$ | 50.8$^{\pm0.6}$/72.8$^{\pm1.4}$ |
| | Orto | | 87.1$^{\pm1.0}$/76.5$^{\pm1.0}$ | 82.4$^{\pm2.7}$/77.9$^{\pm1.4}$ | | 78.9$^{\pm0.4}$/80.0$^{\pm0.3}$ | 80.1$^{\pm0.4}$/79.5$^{\pm0.3}$ |
| | CCA | | **87.8**$^{\pm0.1}$/61.7$^{\pm1.4}$ | 84.2$^{\pm0.6}$/66.4$^{\pm0.7}$ | | 77.7$^{\pm0.3}$/78.9$^{\pm0.3}$ | 79.2$^{\pm0.4}$/78.2$^{\pm0.3}$ |
| | Rank | | 86.5$^{\pm1.3}$/71.4$^{\pm1.6}$ | 82.9$^{\pm0.9}$/78.6$^{\pm0.8}$ | | 75.4$^{\pm0.9}$/74.4$^{\pm1.1}$ | **82.0**$^{\pm0.3}$/77.8$^{\pm0.7}$ |
| | Or-Ra | | 87.4$^{\pm1.0}$/79.0$^{\pm0.5}$ | 85.9$^{\pm1.5}$/79.3$^{\pm0.6}$ | | 73.4$^{\pm1.0}$/77.8$^{\pm0.6}$ | 81.1$^{\pm0.6}$/77.1$^{\pm0.7}$ |
| | Avg. | | 87.1/71.3 | 84.0/74.1 | | 77.0/78.1 | 74.6/77.1 |
| B | MSE | 95.1/94.7 | **89.6**$^{\pm0.2}$/86.7$^{\pm0.2}$ | 86.7$^{\pm0.6}$/82.6$^{\pm1.4}$ | 84.6/85.4 | 79.8$^{\pm0.2}$/80.9$^{\pm0.2}$ | 77.0$^{\pm1.1}$/77.8$^{\pm0.3}$ |
| | Orto | | 89.1$^{\pm0.3}$/81.6$^{\pm2.1}$ | 86.3$^{\pm0.3}$/84.4$^{\pm0.2}$ | | 79.4$^{\pm0.2}$/80.0$^{\pm0.1}$ | 80.3$^{\pm0.2}$/79.6$^{\pm0.3}$ |
| | CCA | | 89.0$^{\pm0.1}$/86.1$^{\pm0.4}$ | 86.2$^{\pm0.5}$/84.8$^{\pm0.1}$ | | 80.3$^{\pm0.3}$/79.7$^{\pm0.3}$ | 79.3$^{\pm0.2}$/79.6$^{\pm0.2}$ |
| | Rank | | 89.3$^{\pm0.1}$/86.4$^{\pm0.2}$ | 85.8$^{\pm0.3}$/83.6$^{\pm0.3}$ | | 80.5$^{\pm0.2}$/79.0$^{\pm0.3}$ | 81.2$^{\pm0.1}$/78.5$^{\pm0.5}$ |
| | Or-Ra | | 88.2$^{\pm1.0}$/81.5$^{\pm1.2}$ | 86.1$^{\pm1.1}$/84.1$^{\pm1.2}$ | | 79.7$^{\pm0.4}$/79.4$^{\pm0.4}$ | **81.7**$^{\pm0.2}$/78.3$^{\pm0.2}$ |
| | Avg. | | 89.0/84.5 | 86.2/83.9 | | 79.9/79.8 | 79.9/78.8 |
| | | | | **LSTM** | | | |
| - | MSE | 96.4/95.7 | 80.0$^{\pm3.3}$/79.8$^{\pm0.8}$ | 83.1$^{\pm1.0}$/68.1$^{\pm1.9}$ | 85.3/84.3 | 79.2$^{\pm0.8}$/78.7$^{\pm0.8}$ | 76.3$^{\pm1.7}$/76.6$^{\pm0.7}$ |
| | Orto | | **87.6**$^{\pm0.5}$/71.9$^{\pm1.0}$ | 84.8$^{\pm3.7}$/78.1$^{\pm2.2}$ | | 79.8$^{\pm1.5}$/81.2$^{\pm0.7}$ | 81.8$^{\pm0.4}$/78.9$^{\pm0.3}$ |
| | CCA | | 87.0$^{\pm0.4}$/76.4$^{\pm0.5}$ | 85.3$^{\pm1.0}$/77.0$^{\pm4.8}$ | | 79.9$^{\pm0.4}$/78.6$^{\pm0.6}$ | 81.8$^{\pm0.7}$/79.0$^{\pm1.5}$ |
| | Rank | | 86.5$^{\pm1.1}$/73.1$^{\pm0.9}$ | 84.2$^{\pm0.5}$/81.7$^{\pm0.7}$ | | 69.8$^{\pm2.3}$/77.4$^{\pm0.2}$ | **82.5**$^{\pm0.4}$/79.0$^{\pm0.4}$ |
| | Or-Ra | | 85.8$^{\pm1.8}$/77.5$^{\pm1.8}$ | 85.6$^{\pm0.8}$/76.5$^{\pm0.5}$ | | 74.7$^{\pm3.3}$/79.8$^{\pm1.0}$ | 82.1$^{\pm0.4}$/79.6$^{\pm0.5}$ |
| | Avg. | | 85.4/75.7 | 84.6/76.3 | | 76.7/79.1 | 80.9/78.6 |
| B | MSE | 96.4/95.9 | 83.8$^{\pm1.4}$/80.4$^{\pm1.3}$ | 81.4$^{\pm2.3}$/78.0$^{\pm4.6}$ | 84.5/85.9 | 79.4$^{\pm0.7}$/81.0$^{\pm0.6}$ | 80.3$^{\pm2.4}$/78.0$^{\pm3.4}$ |
| | Orto | | **86.8**$^{\pm0.4}$/81.2$^{\pm1.2}$ | 86.0$^{\pm0.8}$/82.6$^{\pm2.8}$ | | 81.1$^{\pm1.2}$/**83.0**$^{\pm0.5}$ | 82.6$^{\pm0.8}$/81.9$^{\pm0.8}$ |
| | CCA | | 86.2$^{\pm0.3}$/78.1$^{\pm1.1}$ | 83.6$^{\pm1.6}$/83.3$^{\pm0.7}$ | | 82.0$^{\pm0.5}$/82.4$^{\pm0.6}$ | 82.7$^{\pm0.8}$/81.5$^{\pm1.0}$ |
| | Rank | | 84.4$^{\pm4.2}$/81.8$^{\pm2.8}$ | 85.3$^{\pm1.1}$/82.2$^{\pm1.2}$ | | 79.6$^{\pm0.9}$/78.9$^{\pm2.8}$ | 82.4$^{\pm0.5}$/79.1$^{\pm1.5}$ |
| | Or-Ra | | 85.3$^{\pm1.0}$/78.5$^{\pm2.6}$ | 82.2$^{\pm1.2}$/80.7$^{\pm0.9}$ | | 80.4$^{\pm0.9}$/80.8$^{\pm1.2}$ | 82.0$^{\pm0.4}$/79.4$^{\pm0.5}$ |
| | Avg. | | 85.3/80.0 | 83.7/81.4 | | 80.5/81.2 | 82.0/80.0 |

Table A.13: Cross-lingual accuracy results for linear transformations obtained on the binary SST-Allocine (English-French) dataset pair. Normalization was applied only before the transformation.

| | | Evaluated on **French** | | | Evaluated on **English** | | |
|---|---|---|---|---|---|---|---|
| **Norm.** | **Method** | **Monoling.** | **EN-s ⇒FR-t** | **FR-t ⇒EN-s** | **Monoling.** | **FR-s ⇒EN-t** | **EN-t ⇒FR-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| | | | | **CNN** | | | |
| - | MSE | 95.0/94.3 | $86.7^{\pm0.9}/67.9^{\pm1.4}$ | $84.5^{\pm0.2}/68.4^{\pm0.6}$ | 84.4/83.7 | $79.6^{\pm0.2}/79.2^{\pm0.3}$ | $50.8^{\pm0.6}/72.8^{\pm1.4}$ |
| | Orto | | $87.1^{\pm1.0}/76.5^{\pm1.0}$ | $82.4^{\pm2.7}/77.9^{\pm1.4}$ | | $78.9^{\pm0.4}/80.0^{\pm0.3}$ | $80.1^{\pm0.4}/79.5^{\pm0.3}$ |
| | CCA | | $\mathbf{87.8}^{\pm0.1}/61.7^{\pm1.4}$ | $84.2^{\pm0.6}/66.4^{\pm0.7}$ | | $77.7^{\pm0.3}/78.9^{\pm0.3}$ | $79.2^{\pm0.4}/78.2^{\pm0.3}$ |
| | Rank | | $86.5^{\pm1.3}/71.4^{\pm1.6}$ | $82.9^{\pm0.9}/78.6^{\pm0.8}$ | | $75.4^{\pm0.9}/74.4^{\pm1.1}$ | $\underline{\mathbf{82.0}}^{\pm0.3}/77.8^{\pm0.7}$ |
| | Or-Ra | | $87.4^{\pm1.0}/79.0^{\pm0.5}$ | $\underline{85.9}^{\pm1.5}/79.3^{\pm0.6}$ | | $73.4^{\pm1.0}/77.8^{\pm0.6}$ | $81.1^{\pm0.6}/77.1^{\pm0.7}$ |
| | Avg. | | 87.1/71.3 | 84.0/74.1 | | 77.0/78.1 | 74.6/77.1 |
| B,A | MSE | 95.1/94.7 | $\underline{89.2}^{\pm0.2}/\underline{84.3}^{\pm1.4}$ | $\underline{87.2}^{\pm0.2}/\underline{85.6}^{\pm0.4}$ | 84.6/85.4 | $79.6^{\pm0.2}/\underline{80.5}^{\pm0.2}$ | $\underline{80.8}^{\pm0.2}/\underline{80.5}^{\pm0.1}$ |
| | Orto | | $\underline{89.4}^{\pm0.2}/\underline{86.8}^{\pm0.2}$ | $\underline{85.8}^{\pm0.5}/\underline{85.2}^{\pm0.2}$ | | $\underline{79.0}^{\pm0.1}/\underline{80.6}^{\pm0.4}$ | $\underline{80.8}^{\pm0.3}/\underline{80.5}^{\pm0.2}$ |
| | CCA | | $\underline{88.7}^{\pm0.6}/\underline{85.1}^{\pm0.9}$ | $\underline{86.3}^{\pm0.2}/\underline{83.2}^{\pm0.4}$ | | $\underline{80.0}^{\pm0.3}/\underline{80.5}^{\pm0.3}$ | $\underline{79.9}^{\pm0.3}/\underline{80.6}^{\pm0.1}$ |
| | Rank | | $\underline{89.3}^{\pm0.2}/\underline{86.1}^{\pm0.4}$ | $\underline{84.3}^{\pm0.3}/\underline{84.5}^{\pm0.3}$ | | $\underline{79.7}^{\pm0.5}/\underline{79.8}^{\pm0.2}$ | $81.5^{\pm0.3}/\underline{79.3}^{\pm0.3}$ |
| | Or-Ra | | $\underline{\mathbf{89.6}}^{\pm0.2}/\underline{85.5}^{\pm0.6}$ | $85.5^{\pm0.5}/\underline{83.8}^{\pm0.4}$ | | $\underline{79.2}^{\pm0.3}/\underline{80.4}^{\pm0.3}$ | $\underline{\mathbf{81.7}}^{\pm0.3}/\underline{80.1}^{\pm0.2}$ |
| | Avg. | | $\underline{89.2}/\underline{85.6}$ | $\underline{85.8}/\underline{84.5}$ | | $\underline{79.5}/\underline{80.4}$ | $\underline{80.9}/\underline{80.2}$ |
| | | | | **LSTM** | | | |
| - | MSE | 96.4/95.7 | $80.0^{\pm3.3}/79.8^{\pm0.8}$ | $83.1^{\pm1.0}/68.1^{\pm1.9}$ | 85.3/84.3 | $79.2^{\pm0.8}/78.7^{\pm0.8}$ | $76.3^{\pm1.7}/76.6^{\pm0.7}$ |
| | Orto | | $\underline{\mathbf{87.6}}^{\pm0.5}/71.9^{\pm1.0}$ | $\underline{84.8}^{\pm3.7}/78.1^{\pm2.2}$ | | $79.8^{\pm1.5}/\underline{81.2}^{\pm0.7}$ | $81.8^{\pm0.4}/78.9^{\pm0.3}$ |
| | CCA | | $\underline{87.0}^{\pm0.4}/76.4^{\pm0.5}$ | $\underline{85.3}^{\pm1.0}/77.0^{\pm4.8}$ | | $79.9^{\pm0.4}/78.6^{\pm0.6}$ | $81.8^{\pm0.7}/79.0^{\pm1.5}$ |
| | Rank | | $\underline{86.5}^{\pm1.1}/73.1^{\pm0.9}$ | $\underline{84.2}^{\pm0.5}/81.7^{\pm0.7}$ | | $69.8^{\pm2.3}/77.4^{\pm0.2}$ | $\underline{\mathbf{82.5}}^{\pm0.4}/79.0^{\pm0.4}$ |
| | Or-Ra | | $85.8^{\pm1.8}/77.5^{\pm1.8}$ | $\underline{85.6}^{\pm0.8}/76.5^{\pm0.5}$ | | $74.7^{\pm3.3}/79.8^{\pm1.0}$ | $82.1^{\pm0.4}/\underline{79.6}^{\pm0.5}$ |
| | Avg. | | 85.4/75.7 | $\underline{84.6}/76.3$ | | 76.7/79.1 | 80.9/78.6 |
| B,A | MSE | 96.4/95.9 | $\underline{86.1}^{\pm0.6}/\underline{80.6}^{\pm2.4}$ | $\underline{84.5}^{\pm1.2}/\underline{78.8}^{\pm2.5}$ | 84.5/85.9 | $\underline{81.5}^{\pm0.7}/\underline{81.6}^{\pm0.6}$ | $\underline{82.5}^{\pm0.3}/\underline{80.2}^{\pm0.3}$ |
| | Orto | | $86.3^{\pm0.6}/\underline{80.6}^{\pm3.0}$ | $83.9^{\pm2.6}/\underline{79.8}^{\pm1.8}$ | | $\underline{81.4}^{\pm0.7}/81.0^{\pm1.0}$ | $\underline{82.6}^{\pm0.8}/\underline{80.7}^{\pm1.1}$ |
| | CCA | | $\mathbf{86.7}^{\pm0.9}/\underline{84.0}^{\pm0.9}$ | $84.3^{\pm1.2}/\underline{82.5}^{\pm1.6}$ | | $\underline{80.6}^{\pm0.9}/\underline{81.9}^{\pm0.9}$ | $\underline{81.9}^{\pm0.9}/\underline{82.5}^{\pm1.0}$ |
| | Rank | | $85.9^{\pm0.5}/\underline{83.0}^{\pm0.1}$ | $83.6^{\pm0.5}/\underline{82.3}^{\pm1.5}$ | | $\underline{82.2}^{\pm0.7}/\underline{82.1}^{\pm0.6}$ | $82.0^{\pm0.7}/\underline{\mathbf{83.0}}^{\pm0.8}$ |
| | Or-Ra | | $\underline{85.9}^{\pm1.1}/\underline{78.9}^{\pm4.5}$ | $82.4^{\pm2.7}/\underline{81.0}^{\pm2.5}$ | | $\underline{77.7}^{\pm1.8}/\underline{80.2}^{\pm1.4}$ | $\underline{82.3}^{\pm0.5}/78.0^{\pm1.0}$ |
| | Avg. | | $\underline{86.2}/\underline{81.4}$ | $83.7/\underline{80.9}$ | | $\underline{80.7}/\underline{81.4}$ | $\underline{82.3}/\underline{80.9}$ |

Table A.14: Cross-lingual accuracy results for linear transformations obtained on the binary SST-Allocine (English-French) dataset pair. Normalization was applied before and after the transformation.

| | | Evaluated on **Czech** | | | Evaluated on **French** | | |
|---|---|---|---|---|---|---|---|
| **Norm.** | **Method** | **Monoling.** | **FR-s ⇒CS-t** | **CS-t ⇒FR-s** | **Monoling.** | **CS-s ⇒FR-t** | **FR-t ⇒CS-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| | | | | **CNN** | | | |
| - | MSE | 93.9/91.5 | $85.4^{\pm0.1}/76.0^{\pm0.4}$ | $56.0^{\pm1.8}/68.5^{\pm2.9}$ | 95.0/94.3 | $75.8^{\pm0.9}/65.3^{\pm0.7}$ | $58.4^{\pm2.5}/70.1^{\pm1.5}$ |
| | Orto | | $\underline{86.0}^{\pm0.2}/\underline{78.1}^{\pm0.4}$ | $\underline{86.3}^{\pm0.2}/78.5^{\pm0.3}$ | | $\underline{84.6}^{\pm0.2}/\underline{80.8}^{\pm0.2}$ | $84.0^{\pm0.3}/\underline{78.4}^{\pm0.5}$ |
| | CCA | | $83.7^{\pm0.3}/75.9^{\pm0.4}$ | $\underline{83.9}^{\pm0.2}/\underline{72.5}^{\pm0.4}$ | | $\underline{84.7}^{\pm0.3}/79.8^{\pm0.3}$ | $76.9^{\pm0.5}/\underline{73.7}^{\pm0.5}$ |
| | Rank | | $81.7^{\pm0.9}/75.1^{\pm1.1}$ | $86.2^{\pm0.3}/68.8^{\pm0.4}$ | | $82.7^{\pm0.5}/77.0^{\pm0.8}$ | $\underline{84.7}^{\pm0.1}/71.6^{\pm0.8}$ |
| | Or-Ra | | $82.7^{\pm0.7}/72.6^{\pm1.4}$ | $\underline{\mathbf{87.0}}^{\pm0.2}/75.1^{\pm0.1}$ | | $83.7^{\pm0.7}/\underline{74.9}^{\pm1.3}$ | $\underline{\mathbf{85.3}}^{\pm0.2}/\underline{80.3}^{\pm0.2}$ |
| | Avg. | | 83.9/75.5 | 79.9/72.7 | | 82.3/75.6 | $77.9/\underline{74.8}$ |
| B | MSE | 93.4/92.6 | $\underline{86.0}^{\pm0.2}/\underline{81.4}^{\pm0.6}$ | $\underline{82.2}^{\pm0.9}/\underline{79.2}^{\pm0.6}$ | 95.1/94.7 | $\underline{81.2}^{\pm1.1}/\underline{81.4}^{\pm0.7}$ | $\underline{85.4}^{\pm0.1}/\underline{83.7}^{\pm0.2}$ |
| | Orto | | $85.4^{\pm0.2}/76.8^{\pm0.9}$ | $85.6^{\pm0.3}/\underline{82.4}^{\pm0.8}$ | | $84.5^{\pm0.9}/77.9^{\pm0.7}$ | $\underline{84.9}^{\pm0.6}/74.4^{\pm1.6}$ |
| | CCA | | $\underline{84.8}^{\pm0.1}/\underline{80.2}^{\pm0.4}$ | $83.7^{\pm0.9}/68.5^{\pm1.8}$ | | $84.6^{\pm0.4}/\underline{84.0}^{\pm0.4}$ | $\underline{77.3}^{\pm2.0}/58.8^{\pm1.0}$ |
| | Rank | | $\underline{84.7}^{\pm0.2}/\underline{81.6}^{\pm0.4}$ | $\underline{86.5}^{\pm0.1}/\underline{81.8}^{\pm0.1}$ | | $\underline{\mathbf{86.1}}^{\pm0.6}/\underline{80.6}^{\pm1.4}$ | $84.4^{\pm0.3}/\underline{73.3}^{\pm0.8}$ |
| | Or-Ra | | $\underline{83.2}^{\pm0.4}/\underline{83.0}^{\pm0.3}$ | $\mathbf{86.8}^{\pm0.2}/\underline{75.2}^{\pm0.7}$ | | $\underline{83.8}^{\pm1.5}/73.4^{\pm2.8}$ | $84.1^{\pm1.1}/80.2^{\pm0.3}$ |
| | Avg. | | $\underline{84.8}/\underline{80.6}$ | $\underline{85.0}/\underline{77.4}$ | | $\underline{84.0}/\underline{79.5}$ | $\underline{83.2}/74.1$ |
| | | | | **LSTM** | | | |
| - | MSE | 94.4/92.1 | $85.6^{\pm0.6}/82.9^{\pm0.7}$ | $84.8^{\pm2.4}/74.5^{\pm3.6}$ | 96.4/95.7 | $81.8^{\pm1.9}/\underline{76.6}^{\pm1.7}$ | $60.6^{\pm2.6}/\underline{67.7}^{\pm4.0}$ |
| | Orto | | $\underline{87.6}^{\pm0.5}/80.3^{\pm0.6}$ | $\mathbf{88.0}^{\pm0.7}/81.5^{\pm0.6}$ | | $73.2^{\pm0.9}/76.0^{\pm4.4}$ | $\underline{73.5}^{\pm1.2}/71.7^{\pm2.9}$ |
| | CCA | | $\underline{87.4}^{\pm0.4}/79.3^{\pm1.1}$ | $87.3^{\pm0.5}/79.3^{\pm0.9}$ | | $80.0^{\pm0.4}/\underline{81.7}^{\pm1.0}$ | $69.1^{\pm1.1}/75.8^{\pm1.2}$ |
| | Rank | | $80.0^{\pm3.6}/82.7^{\pm0.7}$ | $87.8^{\pm0.4}/76.3^{\pm0.4}$ | | $72.8^{\pm5.0}/\underline{74.9}^{\pm3.4}$ | $\underline{85.4}^{\pm0.8}/78.5^{\pm0.9}$ |
| | Or-Ra | | $85.2^{\pm0.7}/79.4^{\pm0.8}$ | $87.6^{\pm0.6}/\underline{81.0}^{\pm0.8}$ | | $81.6^{\pm5.5}/\underline{77.6}^{\pm2.9}$ | $\underline{\mathbf{85.8}}^{\pm1.0}/\underline{84.4}^{\pm0.6}$ |
| | Avg. | | 85.2/80.9 | 87.1/78.5 | | $77.9/\underline{77.4}$ | 74.9/75.6 |
| B | MSE | 93.9/92.6 | $\underline{86.1}^{\pm0.8}/\underline{84.6}^{\pm0.5}$ | $\underline{85.4}^{\pm0.8}/\underline{78.8}^{\pm2.3}$ | 96.4/95.9 | $\underline{84.6}^{\pm0.6}/74.7^{\pm2.1}$ | $\underline{63.0}^{\pm5.7}/56.3^{\pm2.4}$ |
| | Orto | | $86.9^{\pm0.4}/\underline{83.4}^{\pm1.1}$ | $\underline{88.2}^{\pm0.8}/81.5^{\pm1.4}$ | | $\underline{76.3}^{\pm1.4}/\underline{83.9}^{\pm2.3}$ | $69.2^{\pm3.3}/\underline{80.7}^{\pm3.6}$ |
| | CCA | | $86.2^{\pm0.8}/\underline{84.4}^{\pm0.7}$ | $\underline{87.5}^{\pm1.2}/\underline{81.1}^{\pm2.0}$ | | $\underline{85.2}^{\pm1.8}/80.0^{\pm3.0}$ | $\underline{76.1}^{\pm3.5}/\underline{77.2}^{\pm2.1}$ |
| | Rank | | $\underline{87.9}^{\pm0.4}/\underline{83.2}^{\pm1.9}$ | $\mathbf{88.7}^{\pm0.3}/\underline{83.5}^{\pm1.0}$ | | $\underline{77.8}^{\pm4.5}/64.9^{\pm3.6}$ | $84.7^{\pm1.0}/\underline{82.9}^{\pm1.3}$ |
| | Or-Ra | | $\underline{87.0}^{\pm0.6}/\underline{84.3}^{\pm1.6}$ | $\underline{87.7}^{\pm0.9}/80.7^{\pm2.3}$ | | $\underline{82.4}^{\pm3.1}/72.1^{\pm2.3}$ | $\mathbf{85.6}^{\pm1.3}/84.1^{\pm1.3}$ |
| | Avg. | | $\underline{86.8}/\underline{84.0}$ | $\underline{87.5}/\underline{81.1}$ | | $\underline{81.3}/75.1$ | $\underline{75.7}/\underline{76.2}$ |

Table A.15: Cross-lingual accuracy results for linear transformations obtained on the binary CSFD-Allocine (Czech-French) dataset pair. Normalization was applied only before the transformation.

| | | Evaluated on **Czech** | | | Evaluated on **French** | | |
|---|---|---|---|---|---|---|---|
| **Norm.** | **Method** | **Monoling.** | **FR-s ⇒CS-t** | **CS-t ⇒FR-s** | **Monoling.** | **CS-s ⇒FR-t** | **FR-t ⇒CS-s** |
| | | | in-domain/fastText | in-domain/fastText | | in-domain/fastText | in-domain/fastText |
| | | | | **CNN** | | | |
| - | MSE | 93.9/91.5 | 85.4$^{\pm0.1}$/76.0$^{\pm0.4}$ | 56.0$^{\pm1.8}$/68.5$^{\pm2.9}$ | 95.0/94.3 | 75.8$^{\pm0.9}$/65.3$^{\pm0.7}$ | 58.4$^{\pm2.5}$/70.1$^{\pm1.5}$ |
| | Orto | | 86.0$^{\pm0.2}$/78.1$^{\pm0.4}$ | 86.3$^{\pm0.2}$/78.5$^{\pm0.3}$ | | 84.6$^{\pm0.2}$/80.8$^{\pm0.2}$ | 84.0$^{\pm0.3}$/78.4$^{\pm0.5}$ |
| | CCA | | 83.7$^{\pm0.3}$/75.9$^{\pm0.4}$ | 83.9$^{\pm0.2}$/72.5$^{\pm0.4}$ | | 84.7$^{\pm0.3}$/79.8$^{\pm0.3}$ | 76.9$^{\pm0.5}$/73.7$^{\pm0.5}$ |
| | Rank | | 81.7$^{\pm0.9}$/75.1$^{\pm1.1}$ | 86.2$^{\pm0.3}$/68.8$^{\pm0.4}$ | | 82.7$^{\pm0.5}$/77.0$^{\pm0.8}$ | 84.7$^{\pm0.1}$/71.6$^{\pm0.8}$ |
| | Or-Ra | | 82.7$^{\pm0.7}$/72.6$^{\pm1.4}$ | **87.0**$^{\pm0.2}$/75.1$^{\pm0.1}$ | | 83.7$^{\pm0.7}$/74.9$^{\pm1.3}$ | **85.3**$^{\pm0.2}$/80.3$^{\pm0.2}$ |
| | Avg. | | 83.9/75.5 | 79.9/72.7 | | 82.3/75.6 | 77.9/74.8 |
| B,A | MSE | 93.4/92.6 | 84.8$^{\pm0.2}$/82.5$^{\pm0.0}$ | 84.8$^{\pm0.1}$/84.2$^{\pm0.1}$ | 95.1/94.7 | 85.1$^{\pm1.1}$/83.6$^{\pm0.4}$ | 83.3$^{\pm0.7}$/79.9$^{\pm1.5}$ |
| | Orto | | 85.2$^{\pm0.1}$/83.4$^{\pm0.1}$ | 84.5$^{\pm0.3}$/82.8$^{\pm0.1}$ | | 85.6$^{\pm0.8}$/83.9$^{\pm0.7}$ | 85.7$^{\pm1.5}$/75.4$^{\pm2.7}$ |
| | CCA | | 85.2$^{\pm0.1}$/82.9$^{\pm0.2}$ | 83.2$^{\pm0.2}$/82.8$^{\pm0.1}$ | | **85.9**$^{\pm0.8}$/80.9$^{\pm3.2}$ | 82.7$^{\pm0.6}$/81.3$^{\pm1.1}$ |
| | Rank | | 85.6$^{\pm0.1}$/83.9$^{\pm0.1}$ | **86.4**$^{\pm0.2}$/81.6$^{\pm0.1}$ | | 85.4$^{\pm1.3}$/80.5$^{\pm1.4}$ | 84.1$^{\pm1.0}$/78.6$^{\pm1.0}$ |
| | Or-Ra | | 85.0$^{\pm0.2}$/83.9$^{\pm0.2}$ | **86.4**$^{\pm0.2}$/81.4$^{\pm0.3}$ | | 83.2$^{\pm1.6}$/83.3$^{\pm0.3}$ | 84.6$^{\pm0.7}$/77.2$^{\pm2.2}$ |
| | Avg. | | 85.2/83.3 | 85.1/82.6 | | 85.0/82.4 | 84.1/78.5 |
| | | | | **LSTM** | | | |
| - | MSE | 94.4/92.1 | 85.6$^{\pm0.6}$/82.9$^{\pm0.7}$ | 84.8$^{\pm2.4}$/74.5$^{\pm3.6}$ | 96.4/95.7 | 81.8$^{\pm1.9}$/76.6$^{\pm1.7}$ | 60.6$^{\pm2.6}$/67.7$^{\pm4.0}$ |
| | Orto | | 87.6$^{\pm0.5}$/80.3$^{\pm0.6}$ | **88.0**$^{\pm0.7}$/81.5$^{\pm0.6}$ | | 73.2$^{\pm0.9}$/76.0$^{\pm4.4}$ | 73.5$^{\pm1.2}$/71.7$^{\pm2.9}$ |
| | CCA | | 87.4$^{\pm0.4}$/79.3$^{\pm1.1}$ | 87.3$^{\pm0.5}$/79.3$^{\pm0.9}$ | | 80.0$^{\pm0.4}$/81.7$^{\pm1.0}$ | 69.1$^{\pm1.1}$/75.8$^{\pm1.2}$ |
| | Rank | | 80.0$^{\pm3.6}$/82.7$^{\pm0.7}$ | 87.8$^{\pm0.4}$/76.3$^{\pm0.4}$ | | 72.8$^{\pm5.0}$/74.9$^{\pm3.4}$ | 85.4$^{\pm0.8}$/78.5$^{\pm0.9}$ |
| | Or-Ra | | 85.2$^{\pm0.7}$/79.4$^{\pm0.8}$ | 87.6$^{\pm0.6}$/81.0$^{\pm0.8}$ | | 81.6$^{\pm5.5}$/77.6$^{\pm2.9}$ | **85.8**$^{\pm1.0}$/84.4$^{\pm0.6}$ |
| | Avg. | | 85.2/80.9 | 87.1/78.5 | | 77.9/77.4 | 74.9/75.6 |
| B,A | MSE | 93.9/92.6 | 86.6$^{\pm2.1}$/83.4$^{\pm1.4}$ | **88.9**$^{\pm0.2}$/85.1$^{\pm0.7}$ | 96.4/95.9 | 83.0$^{\pm3.5}$/71.4$^{\pm5.6}$ | 77.1$^{\pm3.0}$/83.0$^{\pm2.1}$ |
| | Orto | | 88.2$^{\pm0.8}$/84.1$^{\pm0.6}$ | 87.4$^{\pm1.1}$/84.8$^{\pm0.7}$ | | 77.6$^{\pm2.8}$/76.3$^{\pm4.3}$ | 82.7$^{\pm1.9}$/83.1$^{\pm1.8}$ |
| | CCA | | 87.9$^{\pm0.5}$/84.6$^{\pm1.3}$ | 87.1$^{\pm0.5}$/84.8$^{\pm0.5}$ | | **83.9**$^{\pm1.8}$/76.2$^{\pm2.8}$ | 76.4$^{\pm2.1}$/76.4$^{\pm4.0}$ |
| | Rank | | 88.2$^{\pm0.4}$/81.6$^{\pm2.4}$ | 87.8$^{\pm0.6}$/84.3$^{\pm0.5}$ | | 81.6$^{\pm3.0}$/72.2$^{\pm8.0}$ | 83.5$^{\pm1.9}$/83.1$^{\pm1.9}$ |
| | Or-Ra | | 86.3$^{\pm0.7}$/85.2$^{\pm0.8}$ | 87.9$^{\pm0.5}$/85.2$^{\pm1.1}$ | | 77.8$^{\pm2.5}$/82.8$^{\pm3.0}$ | 82.9$^{\pm0.8}$/79.2$^{\pm3.0}$ |
| | Avg. | | 87.4/83.8 | 87.8/84.8 | | 80.8/75.8 | 80.5/81.0 |

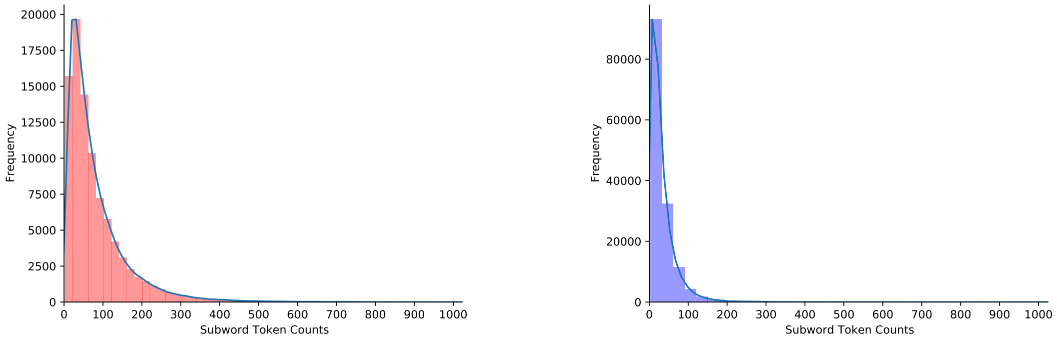
Table A.16: Cross-lingual accuracy results for linear transformations obtained on the binary CSFD-Allocine (Czech-French) dataset pair. Normalization was applied before and after the transformation.

## A.3 Czech Monolingual Results and Hyper-parameters

For fine-tuning of the Transformer-based models in Chapter 8 in Section 8.1, we use the same modification (Loshchilov & Hutter, 2017) of the Adam (Kingma & Ba, 2015) optimizer with default weight decay set to 1e-2. We use different learning rates and a number of epochs for each combination of the models and datasets, see Table A.17. We use either constant linear rate or linear learning rate decay without learning rate warm-up. We use default values of all other hyper-parameters.

| Model | 3 Classes | | | 2 Classes | | |
|---|---|---|---|---|---|---|
| | CSFD | FB | Mallcz | CSFD | FB | Mallcz |
| Log. reg. (ours) | 79.6 | 67.9 | 76.7 | 91.4 | 88.1 | 89.0 |
| LSTM (ours) | $79.9^{\pm0.2}$ (5e-4 / 2)* | $72.9^{\pm0.5}$ (5e-4 / 5)* | $73.4^{\pm0.1}$ (5e-4 / 10) ‡ | $91.8^{\pm0.1}$ (5e-4 / 2)* | $90.1^{\pm0.2}$ (5e-4 / 5)* | $88.0^{\pm0.2}$ (5e-4 / 2)‡ |
| Czert-A | $79.9^{\pm0.6}$ (2e-6 / 8) | $73.1^{\pm0.6}$ (2e-5 / 8) | $76.8^{\pm0.4}$ (2e-5 / 12) | $91.8^{\pm0.8}$ (2e-5 / 8) | $91.3^{\pm0.2}$ (2e-5 / 15)† | $91.2^{\pm0.3}$ (2e-5 / 14) |
| Czert-B | $84.9^{\pm0.1}$ (2e-5 / 12) | $76.9^{\pm0.4}$ (2e-6 / 5)† | $79.4^{\pm0.2}$ (2e-5 / 15) | $94.4^{\pm0.1}$ (2e-5 / 15) | $94.0^{\pm0.3}$ (2e-5 / 2) | $92.9^{\pm0.2}$ (2e-5 / 15) |
| mBERT | $82.9^{\pm0.1}$ (2e-6 / 13) | $71.6^{\pm0.1}$ (2e-6 / 13)† | $70.8^{\pm5.7}$ (2e-5 / 10) | $93.1^{\pm0.3}$ (2e-6 / 14)† | $88.8^{\pm0.4}$ (2e-5 / 8) | $72.8^{\pm3.1}$ (2e-5 / 1) |
| SlavicBERT | $82.6^{\pm0.1}$ (2e-6 / 12) | $73.9^{\pm0.5}$ (2e-5 / 4) | $75.3^{\pm2.5}$ (2e-5 / 10) | $93.5^{\pm0.3}$ (2e-6 / 15)† | $89.8^{\pm0.4}$ (2e-5 / 9)† | $91.0^{\pm0.2}$ (2e-6 / 14)† |
| RandomALBERT | $75.8^{\pm0.2}$ (2e-6 / 14) | $62.5^{\pm0.5}$ (2e-6 / 14)† | $64.8^{\pm0.3}$ (2e-6 / 15)† | $90.0^{\pm0.2}$ (2e-6 / 14)† | $81.7^{\pm0.6}$ (2e-6 / 15)† | $85.4^{\pm0.1}$ (2e-6 / 14)† |
| XLM-R$_{\text{Base}}$ | $85.0^{\pm0.1}$ (2e-6 / 15)† | $77.8^{\pm0.5}$ (2e-6 / 7)† | $75.4^{\pm0.1}$ (2e-6 / 15)† | $94.3^{\pm0.3}$ (2e-6 /14) † | $93.3^{\pm0.7}$ (2e-6 / 5)† | $92.6^{\pm0.1}$ (2e-6 / 12)† |
| XLM-R$_{\text{Large}}$ | **87.2**$^{\pm0.1}$ (2e-6 / 11 ) | **81.7**$^{\pm0.6}$ (2e-6 / 5)† | **79.8**$^{\pm0.2}$ (2e-6 / 24)† | **96.0**$^{\pm0.0}$ (2e-6 / 14)† | **96.1**$^{\pm0.0}$ (2e-6 / 15) | **94.4**$^{\pm0.0}$ (2e-6 / 15)† |
| XLM | $83.8^{\pm0.1}$ (2e-5 / 11) | $71.5^{\pm1.6}$ (2e-6 / 9)† | $77.6^{\pm0.1}$ (2e-6 / 14)† | $93.9^{\pm0.2}$ (2e-5 / 5) | $89.9^{\pm0.3}$ (2e-6 / 15)† | $92.0^{\pm0.2}$ (2e-6 / 16)† |

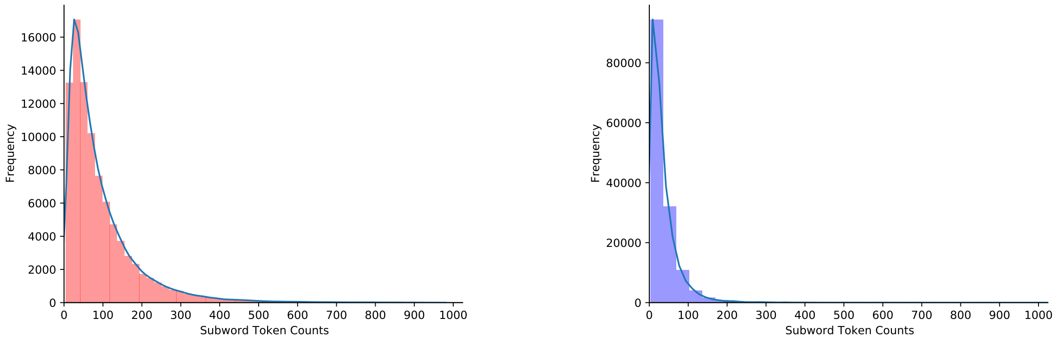
Table A.17: The final monolingual results as macro $F_1$ score and hyper-parameters for all three Czech polarity datasets on two and three classes. For experiments with neural networks performed by us, we present the results with a 95% confidence interval. For each result, we state the used learning rate and the number of epochs used for the training. The † symbol denotes that the result was obtained with a constant learning rate, $*$ denotes the cosine learning rate decay, ‡ denotes exponential learning rate decay; otherwise, the linear learning rate decay was used.



(a) CSFD – Czert-B

(b) Mallcz – Czert-B

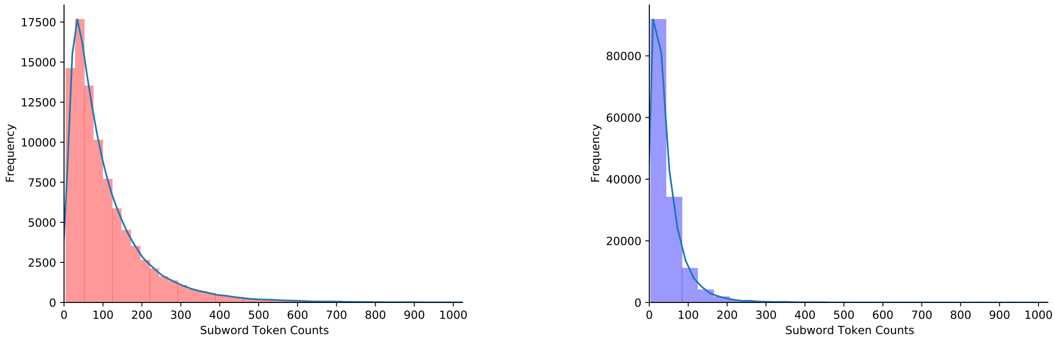
Figure A.1: Subword token histograms for the CSFD and Mallcz datasets for the `Czert-B` model.



(a) CSFD – XLM-R$_{\text{Base}}$ and XLM-R$_{\text{Large}}$

(b) Mallcz – XLM-R$_{\text{Base}}$ and XLM-R$_{\text{Large}}$

Figure A.2: Subword token histograms for the CSFD and Mallcz datasets for the `XLM-R`$_{\texttt{Base}}$ and `XLM-R`$_{\texttt{Large}}$ models.



(a) CSFD – mBERT

(b) Mallcz – mBERT

Figure A.3: Subword token histograms for the CSFD and Mallcz datasets for the `mBERT` model.

## A.4 Details and Hyper-parameters of Subjectivity Experiments

We fine-tune all parameters of the model, including the added classification layers. We run the experiments for at most ten epochs with the linear learning rate decay (without learning rate warm-up) with the initial learning rates ranging from 2e-7 to 2e-4. The 2e-4 learning rate was used only for the Czech Electra model, when used with other models, the models started to diverge. The batch size is set to 32 and the max sequence length of the input is 200 since we classify sentences and the vast majority of them fit into this length. During fine-tuning, we tried a variety of hyper-parameters, we use the Adam (Kingma & Ba, 2015) optimizer with default parameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$) and the cross-entropy loss function. We randomly shuffle training data before each epoch. In Tables A.18, A.21, A.22, A.19 and A.20 we report results with the used initial learning rate and a number of epochs in parentheses. The first number in brackets is the initial learning rate and the second is the number of epochs for fine-tuning.

| Model | Subj-CS (cs-train) | Subj-CS-L (cs-L-train) |
|---|---|---|
| | cs-test | cs-test |
| Czech Electra | 91.9 ± 0.3 (2e-4 / 4) | 91.2 ± 0.1 (2e-5 / 7) |
| Czert-B | 92.9 ± 0.2 (2e-5 / 3) | 91.8 ± 0.1* (2e-6 / 7) |
| RobeCzech | 93.3 ± 0.2* (2e-5 / 7) | 91.6 ± 0.1 (2e-6 / 2) |
| mBERT | 91.2 ± 0.2 (2e-5 / 3) | 91.1 ± 0.1 (2e-6 / 5) |
| XLM-R$_{\text{Large}}$ | **93.6 ± 0.1** (2e-5 / 4) | **92.0 ± 0.1** (2e-6 / 9) |

Table A.18: Results with model hyper-parameters for Czech monolingual experiments reported as average accuracy for the testing `cs-test` data part. The * symbol denotes results containing intersection in confidence interval with the best model.

| Model | CS → EN (cs-train) | | CS → EN (cs-L-train) | | Monolingual (en-train) |
|---|---|---|---|---|---|
| | cs-dev | en-test | en-dev | en-test | en-test |
| mBERT | 92.1 ± 0.4 | 89.0 ± 0.9 (2e-5 / 3) | 85.8 ± 0.9 | 85.5 ± 0.9 (2e-6 / 1) | 95.9 ± 0.1 (2e-5 / 10) |
| XLM-R$_{\text{Large}}$ | 94.4 ± 0.4 | **92.9 ± 0.4** (2e-5 / 4) | 93.4 ± 0.2 | **91.0 ± 0.3** (2e-7 / 1) | 97.3 ± 0.1 (2e-6 / 10) |

Table A.19: Accuracy results with model hyper-parameters for cross-lingual experiments from Czech to English along with the results for models trained on monolingual data.

| Model | en-test | en-10-fold |
|---|---|---|
| BERT | 96.6 ± 0.2 (2e-5 / 3) | 96.9 ± 0.3 (2e-5 / 9) |
| mBERT | 95.9 ± 0.1 (2e-5 / 10) | 96.0 ± 0.2 (2e-5 / 5) |
| XLM-R$_{\text{Large}}$ | **97.3 ± 0.1** (2e-6 / 10) | **97.3 ± 0.2** (2e-5 / 4) |

Table A.21: Results with model hyper-parameters for English monolingual experiments reported as average accuracy for the testing `en-test` and `en-10-fold` data parts.

| Model | EN → CS | | Monoling. (cs-train) |
|---|---|---|---|
| | en-dev | cs-test | cs-test |
| mBERT | 95.4 ± 0.2 | 86.2 ± 0.3 (2e-5 / 10) | 91.2 ± 0.2 (2e-5 / 3) |
| XLM-R$_{\text{Large}}$ | 97.6 ± 0.2 | **90.8 ± 0.3** (2e-6 / 10) | 93.6 ± 0.1 (2e-5 / 4) |

Table A.22: Accuracy results with model hyper-parameters for cross-lingual experiments from English to Czech along with the results for models trained on monolingual data.

| | Joint (cs-train + en-train) | | Monolingual (cs-train) | Monolingual (en-train) |
|---|---|---|---|---|
| Model | cs-test | en-test | cs-test | en-test |
| mBERT | 91.1 ± 0.2 | 95.7 ± 0.2 (2e-5 / 3) | 91.2 ± 0.2 (2e-5 / 3) | 95.9 ± 0.1 (2e-5 / 10) |
| XLM-R$_{\text{Large}}$ | **93.9** ± **0.2** | **96.9** ± **0.1** (2e-6 / 10) | 93.6 ± 0.1 (2e-5 / 4) | 97.3 ± 0.1 (2e-6 / 10) |

Table A.20: Accuracy results with hyper-parameters for models jointly trained on English and Czech data along with the results for models trained on monolingual data.

## A.5 Prompts for Binary Classification

Figures A.4, A.5 and A.6 show examples of prompts for the following review: *"The movie was fantastic!!!"*.

**Basic prompt for binary classification**

You are a sentiment classifier, classify the following review as "positive" or "negative".
Answer in one word only.

The review:

The movie was fantastic!!!

Figure A.4: Example of the basic prompt for binary classification.

**Advanced prompt for binary classification**

You are a Movie and TV Show Review Sentiment Analyzer. You will be given a text of a movie or TV show review, please analyze its content and determine the most appropriate category from the following list. The categories are divided based on the type of sentiment:

Category 1 - positive: Includes reviews that are satisfied with the movie or TV show.
Category 2 - negative: Includes reviews that are dissatisfied with the movie or TV show.
The text for analysis will be marked with four slashes, i.e., ////.

Step 1:#### Judge the overall mood of the text and determine which category the text most likely belongs to.
Step 2:#### Focus more closely on the keywords used in the text. Check if the keywords suggest a specific category. For instance, if the text extensively praises the movie or TV show, you should choose "Positive". If the text criticizes the movie or TV show, choose "Negative".
Step 3:#### Determine the final category based on the highest probability.

Use the following format:
Step 1:#### <rationale for Step 1>
Step 2:#### <rationale for Step 2>
Step 3:#### <rationale for Step 3>

User's answer:#### <the evaluated sentiment itself>

Ensure that you are inserting #### to separate each step.

////The movie was fantastic!!!////

Figure A.5: Example of the advanced prompt for binary classification.

**In-context prompt for binary classification**

You are a sentiment classifier. You will be given a review, please classify the review as "positive" or "negative". Answer in one word only. As an example, you will obtain examples of the reviews and the desired output.

The examples:
Review:"An opportunity missed." sentiment:negative
Review:"The most consistently funny of the Austin Powers films." sentiment:positive
Review:"Very good movie from my favourite director." sentiment:positive
Review:"Even as I valiantly struggled to remain interested, or at least conscious":negative

Ensure that the output is only one word, i.e., one of the sentiment classes.

The review:

The movie was fantastic!!!

Figure A.6: Example of the in-context prompt for binary classification.

## A.6 Examples of LLMs Outputs

We provide examples of ChatGPT and Llama 2 model's outputs for the advanced prompt in Figures A.7, A.8, A.9, A.10, A.11 and A.12.

**Review**: *A masterpiece four years in the making.*
**Label**: positive

**Model output:**

Step 1:####
Based on the given text, it seems to express a positive sentiment towards the movie or TV show.

Step 2:####
The text includes the phrase "A masterpiece four years in the making," which suggests a highly positive sentiment towards the movie or TV show.

Step 3:####
Considering the positive sentiment expressed in the text, the final category is "Positive."

User's answer:#### Positive

(a) ChatGPT output.

**Review**: *A masterpiece four years in the making.*
**Label**: positive

**Model output:**

Step 1:#### The overall mood of the text is very positive, suggesting a high level of satisfaction and admiration for the movie or TV show.

Step 2:#### Keywords such as "masterpiece" and "four years in the making" imply a sense of accomplishment and dedication, further reinforcing the positive tone.

Step 3:#### Based on the strong language used in the text, it is likely that the reviewer has nothing but praise for the movie or TV show, making the final category "Positive."

User's answer:#### Positive

(b) Llama 2 output.

Figure A.7: Example of LLMs outputs for the advanced prompt for English review annotated as *positive*.

**Review**: *This film, starring Anthony Hopkins and Chris Rock, is your typical 'fish out of water' story.*
**Label**: neutral

**Model output:**

Step 1:#### In Step 1, the overall mood of the text seems to be neutral. The reviewer describes the movie as a typical "fish out of water" story, which suggests a mixed or indifferent sentiment.
Step 2:#### In Step 2, the keywords used in the text do not strongly suggest a specific sentiment. The description of the film as a "typical 'fish out of water' story" can be interpreted as neither positive nor negative.
Step 3:#### Based on the analysis in Step 1 and Step 2, the final category for this review is neutral. The text does not express a clear positive or negative sentiment, indicating a neutral response to the movie.

User's answer:#### Neutral

(a) ChatGPT output.

**Review**: *This film, starring Anthony Hopkins and Chris Rock, is your typical 'fish out of water' story.*
**Label**: neutral

**Model output:**

Step 1:#### The overall mood of the text seems to be neutral, as it describes the movie as a "typical" fish-out-of-water story without expressing any strong emotions or opinions.

Step 2:#### Upon closer inspection, the use of the phrase "fish out of water" suggests a lighthearted and comedic tone, which points towards a Positive category. However, the fact that the review doesn't provide any specific details or examples that support this categorization leaves some room for doubt.

Step 3:#### Considering the lack of concrete evidence leaning towards either side, I would classify this review as Neutral.

User's answer:#### Neutral

(b) Llama 2 output.

Figure A.8: Example of LLMs outputs for the advanced prompt for English review annotated as *neutral*.

**Review**: *Efekty mluví za sebe a příběch? Nic takového se zde nevyskytuje!*
**Translation (en)**: *The effects speak for themselves and the story? There is nothing like that here!*
**Label**: negative

**Model output:**

Step 1: The text contains negative words such as "Nic takového se zde nevyskytuje" (Nothing like that is here) which suggest a negative sentiment.
Step 2: The keywords used in the text indicate dissatisfaction with the movie or TV show, as the text mentions that the effects are not impressive and there is no interesting story.
Step 3: Based on the negative sentiment expressed in the text, the most appropriate category is Category 3 - negative.

User's answer: Category 3 - negative

(a) ChatGPT output.

**Review**: *Efekty mluví za sebe a příběch? Nic takového se zde nevyskytuje!*
**Translation (en)**: *The effects speak for themselves and the story? There is nothing like that here!*
**Label**: negative

**Model output:**

Step 1:#### The overall mood of the text is negative.

Step 2:#### The reviewer uses phrases such as "Efekty mluví za sebe" (effects speak for themselves) and "nic takového se zde nevyskytuje" (nothing like this can be found here), indicating disappointment and frustration with the lack of substance in the storytelling.

Step 3:#### Based on the language used, it appears that the reviewer was expecting a more engaging and meaningful experience but was left unsatisfied. Therefore, the final category is Negative.

User's answer:#### Negative

(b) Llama 2 output.

Figure A.9: Example of LLMs outputs for the advanced prompt for Czech review annotated as *negative*.

**Review**: *I po letech aktuální sonda do hamburgerového světa - zábavné, pravdivé a děsivé.*
**Translation (en)**: *A timely probe into the hamburger world, even after all these years - funny, true and scary.*
**Label**: positive

**Model output:**

Step 1: The overall mood of the text seems to be positive because the review mentions that the movie is "zábavné" (fun) and "pravdivé" (truthful).

Step 2: The review uses positive words such as "zábavné" (fun), "pravdivé" (truthful), and "hamburgerového světa" (hamburger world) to describe the movie.

Step 3: Based on the positive words used in the review, the final category is "positive".

User's answer: Positive

(a) ChatGPT output.

**Review**: *I po letech aktuální sonda do hamburgerového světa - zábavné, pravdivé a děsivé.*
**Translation (en)**: *A timely probe into the hamburger world, even after all these years - funny, true and scary.*
**Label**: positive

**Model output:**

Step 1:#### The overall mood of the text is positive. The reviewer uses words like "zábavné" (fun), "pravdivé" (true), and "děsivé" (scary) to describe their experience, indicating that they enjoyed the movie or TV show.

Step 2:#### The keywords used in the text, such as "zábavné" and "pravdivé", suggest a positive category. However, the presence of the word "děsivé" could also indicate some negative elements in the movie or TV show.

Step 3:#### Based on the highest probability, the final category is Positive. The reviewer's use of positive adjectives outweighs the potential negative connotations of the word "děsivé", suggesting that they generally had a positive experience with the movie or TV show.

User's answer:#### Positive

(b) Llama 2 output.

Figure A.10: Example of LLMs outputs for the advanced prompt for Czech review annotated as *positive*.

**Review**: *Un scénario béton...avec pleins de rebondissements...un film à voir absolument.*
**Translation (en)**: *A solid script...with lots of twists and turns...a must-see film.*
**Label**: positive

**Model output:**

Step 1:#### The text expresses positive sentiments towards the movie.
Step 2:#### The keywords used in the text such as "scénario béton" (solid screenplay), "pleins de rebondissements" (lots of twists), and "un film à voir absolument" (a movie to absolutely see) suggest positive sentiments.
Step 3:#### Based on the positive overall mood and the keywords used, the final category is positive.

User's answer:#### Positive

(a) ChatGPT output.

**Review**: *Un scénario béton...avec pleins de rebondissements...un film à voir absolument.*
**Translation (en)**: *A solid script...with lots of twists and turns...a must-see film.*
**Label**: positive

**Model output:**

Step 1:#### The overall mood of the text is positive. The use of the phrase "à voir absolument" (a must-see) suggests that the reviewer has a strong positive opinion of the movie.

Step 2:#### The keywords used in the text, such as "scénario béton" (concrete scenario) and "pleins de rebondissements" (lots of twists), imply that the reviewer enjoyed the movie's storyline and found it engaging.

Step 3:#### Based on the positive language used in the text and the emphasis on the movie's entertainment value, it is likely that the reviewer would give the movie a positive rating.

User's answer:#### Positive

(b) Llama 2 output.

Figure A.11: Example of LLMs outputs for the advanced prompt for French review annotated as *positive*.

**Review**: *Humour potache façon teenage movie, scènes de baston/hip hop pas crédibles, personnages creux. Aucun interêt.*
**Translation (en)**: *Teenage-movie-style humor, implausible fight/hip hop scenes, hollow characters. No interest whatsoever.*
**Label**: negative

**Model output:**

Step 1:#### The overall mood of the text seems to be negative. The reviewer mentions that the humor is puerile, the fight scenes and hip hop scenes are not believable, and the characters are shallow. They also state that there is no interest in the movie.
Step 2:#### The keywords used in the text suggest a negative sentiment. The reviewer mentions "humour potache" (puerile humor), "scènes de baston/hip hop pas crédibles" (unbelievable fight scenes/hip hop scenes), and "personnages creux" (shallow characters).
Step 3:#### Based on the negative mood and the negative keywords used in the text, the final category is negative.

User's answer:#### Negative

(a) ChatGPT output.

**Review**: *Humour potache façon teenage movie, scènes de baston/hip hop pas crédibles, personnages creux. Aucun interêt.*
**Translation (en)**: *Teenage-movie-style humor, implausible fight/hip hop scenes, hollow characters. No interest whatsoever.*
**Label**: negative

**Model output:**

Step 1:#### The overall mood of the text is negative.

Step 2:#### The keywords used in the text suggest a negative category. Words such as "potache," "baston/hip hop pas crédibles," and "personnages creux" convey a sense of disappointment and unrealism.

Step 3:#### Based on the highest probability, the final category is Negative.

User's answer:#### Negative

(b) Llama 2 output.

Figure A.12: Example of LLMs outputs for the advanced prompt for French review annotated as *negative*.