Tools for Semi-automatic Preparation of Training Data for OCR

Ladislav Lenc[1,2], Jiří Martínek[1,2], Pavel Král[1,2]

[1] Dept. of Computer Science & Engineering
Faculty of Applied Sciences
University of West Bohemia
Plzeň, Czech Republic
[2] NTIS - New Technologies for the Information Society
Faculty of Applied Sciences
University of West Bohemia
Plzeň, Czech Republic
{llenc,jimar,pkral}@kiv.zcu.cz

Abstract. This work aims at data preparation for OCR systems based on recurrent neural networks. Precisely annotated data are necessary for training a network as well as for evaluation of OCR methods. It is possible to synthesize the data, however such data are not that realistic as the real ones. Manual annotation is thus still needed in many cases, especially in the case of historical documents we are focusing on. Although there are several complex systems for historical document processing, to the best of our knowledge, a simple annotation tool for OCR data is completely missing. Therefore, we propose and implement a set of tools utilizing artificial intelligence that simplify the annotation process. These tools create ground truths for line images that are used for training of nowadays OCR systems. Another contribution of this paper is making these tools freely available for research purposes.

Keywords: CNN · Historical Documents · LSTM · Neural Networks · OCR

1 Introduction

Digitization of historical documents is a very important task for preserving our cultural heritage. Nowadays, a significant number of such documents is already scanned and saved into archival databases and portals. The next step is making such documents accessible to researchers and to the public. The accessibility is very important and it requires further processing of the scanned documents. A crucial task is converting document images to plain text by the means of optical character recognition (OCR) or handwritten text recognition (HTR). This paper is focused on OCR of printed documents.

We concentrate on historical German newspapers from the end of the nineteenth century. The documents are printed in the obsolete Fraktur script. The

scans have several quality issues such as noise, skew, warping and even missing parts of some characters. The documents also have quite a variable page layout. All above mentioned aspects make the OCR of such documents challenging.

Current trend in the OCR field is utilizing neural networks that process whole text lines. Recurrent neural networks (RNN) are usually utilized for this task [3]. A great benefit of RNN based approaches is that the segmentation to characters is not necessary. Some approaches also combine RNN and convolutional neural networks (CNN) [14, 13]. In such approaches CNN serves as a feature extractor for the recurrent layers. It helps mainly in cases when the data are not well aligned and contain noise.

The segmentation free approaches need a significant amount of annotated line images to be able to train the models. The images can be synthesized using available fonts. However, for capturing all aspects of the real data synthetic data are not sufficient. Therefore, some amount of manually annotated data is still necessary. Moreover, the real data must be also used for system evaluation.

Although there are several complex systems for historical document processing, to the best of our knowledge, a simple single–purpose system to facilitate data annotation is completely missing. Therefore, the main goal of this paper consists in developing a set of simple tools dedicated to OCR data annotation. We would like to save human resources as much as possible. Therefore, we integrate several components of artificial intelligence into these tools to minimize manual annotation work. Their main purpose is to create annotation for line based OCR systems, however one of the tools allows also creating character based annotation. Another contribution of this paper consists in public availability of these tools for research purposes[3]. The implemented tools will be used for training and evaluation of an OCR system developed mainly for historical German newspapers. However, they can be utilized for arbitrary data.

2 Related Work

This section first describes the methods currently used in the OCR field. Then, we summarize available tools and methods for image ground truth creation.

Labeling unsegmented sequences was made possible by using connectionist temporal classification (CTC) loss [7]. CTC loss allowed using a single network for the whole labeling process. It gives the probability distribution over all possible label sequences conditioned on the input sequence. The network uses an objective function that maximizes the probabilities of correct labelings. It is differentiable and therefore a standard backpropagation can be used. This approach was first used for speech recognition. It's utilization directly for text recognition was proposed in [9]. It was aimed mainly at handwritten data. However, it is also suitable for printed OCR.

Breuel et. al [3] proposed an efficient OCR system based on LSTM. It is part of the open–source OCR system OCRopus [2]. The method utilizes a 1D bidirectional LSTM network and uses a text line normalization. The normalization

[3] `http://home.zcu.cz/~pkral/sw/`

step is built upon a dictionary of connected component shapes associated with baseline and x–height information. The dictionary is pre–computed on external annotated data. The baseline and x–height lines are mapped to two straight lines using spline interpolation. The system is applied on printed English and Fraktur texts. It reaches 0.6% CER on English and values from 0.16% to 0.82% on Fraktur depending on the quality of scans.

An approach combining CNN and RNN is proposed in [13]. The system utilizes CNN for feature extraction. LSTM is then used for sequence modeling. The model is evaluated on handwritten as well as printed data. It is shown that such a model using 1D LSTM performs well for both data types. This work also presents weighted finite state transducer (WFST) that supplies a language model to the decoding procedure.

A recursive–recurrent network combined with an attention mechanism was proposed in [12]. This work aims at a photo OCR that reads scene texts in natural images. It is focused on the unconstrained scene text recognition. The system performs directly an image to sequence learning. The network contains 8 convolutional layers with varying number of kernels. RNN with 1024 units serves as the underlying character language model. The attention mechanism is used for better image feature usage. The reported accuracy improvement on the Street view text (SVT) database is 9%.

Van Beusekom et al. [15] proposed an approach for automated ground truth generation. The method finds a robust and accurate alignment of a scanned image and corresponding electronic document. It involves printing of the electronic document and scanning it again. Ground truths on pixel level are prepared. It is tested on the UW3 dataset. The estimated ground truths are compared with the real ones and the resulting accuracy is less than one pixel difference.

A simple approach for ground truth creation is proposed in [6]. Aletheia [5], which is a powerful image ground–truthing tool is used for this task. Aletheia automatically detects objects on four levels: regions, text lines, words and glyphs. The outlines of the objects can be adjusted by users. The user also specifies the segmentation of words to single glyphs.

Another tool is DivaDia [4]. It is used for semi–automatic document image analysis. The system proposes page segmentations and the user then must approve it. The ground truths are stored in xml compatible with the PAGE format.

TrueViz [11] is a tool which allows ground truth creating and visualizing. It is written in Java. The data is saved in xml format. It allows multi–lingual text editing. It is available free of charge for research purposes.

3 Tools for Semi-automatic Annotation

This section describes the proposed set of tools as well as the whole data annotation process. The processing pipeline is depicted in Figure 1. Text regions are first extracted from the input images. The regions are then further segmented into single text lines. A character segmenter estimates segmentation to individual characters, which is then corrected by a human annotator. The extracted

character images are then used for synthetic data preparation. These synthetic lines are then utilized as training data for the CRNN classifier. Line annotator tool used the trained model and its output are the annotated lines.

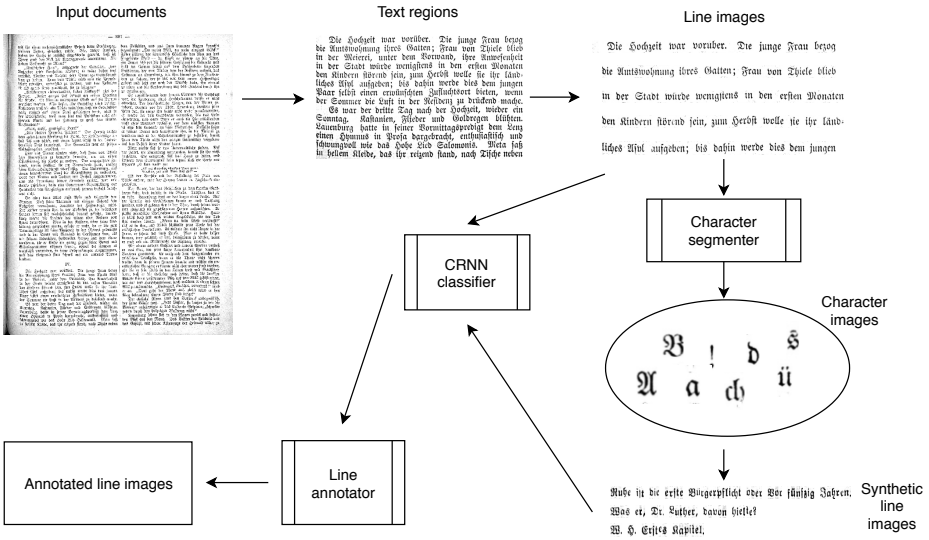

Fig. 1. Processing pipeline of the proposed data preparation procedure.

3.1 Line Image Extraction

This task is solved by automatic tools provided by open–source libraries. The input images are first binarized and deskewed using OCRopus [2]. The next step is extraction of text blocks. Leptonica library [1] is used for this task. Based on the positions of the blocks we determine the reading order. Then we apply a simple algorithm based on projection profiles which finds the text lines. The last step is extracting and post–processing the line images.

3.2 Character Segmentation

The first proposed tool is used for segmenting the text lines into individual characters. It additionally saves the annotated line images as well as the character separator positions. The input of this tool is a set of line images extracted in the previous step. The proposed character segmenter takes a line image and proposes a segmentation to characters. The algorithm is based on projection profiles.

The input image is first inverted and thresholded. Then we calculate the vertical projection profile of the image. This process is illustrated in Figure 2.

Fig. 2. Original text line and its vertical projection profile.

The white peaks indicate presence of characters. The values lower than a specified threshold are considered to be gaps separating the characters. The proposed segmentation of the example image above is shown in Figure 3. This figure shows that several ending character borders are similar to the starting ones, which is depicted with one single separator.

Fig. 3. Proposed segmentation of an example line image.

The example shows several segmentation errors that occur typically in letters *m, n* or *u*. Another issue are some pairs of characters that are impossible to split using projection profile method. Examples are *ch*, *tz* and *ck*. The user interface allows manual correction of incorrectly segmented characters. The tool has options for merging or splitting incorrect segmentations. It is also possible to shift character borders.

Annotation is done by typing the appropriate letter on the keyboard. After pressing a key a next character candidate is selected and it is thus possible to write fluently. Some special characters has dedicated buttons that simplify the typing. There are buttons for the above mentioned hard to segment pairs such as *ch*. An extra button is available also for character *short s*. The user interface is shown in Figure 4

The output of this tool is a set of annotated line images. We also extract individual character images that are saved to directories. We thus can obtain several examples of each character by processing a relatively small amount of data. We also save the positions of character separators that may be useful for successive tasks.

3.3 CRNN Classifier

We utilize a CNN–LSTM line image classifier. It is capable of classifying unsegmented line images and retrieving character sequences. The architecture we use

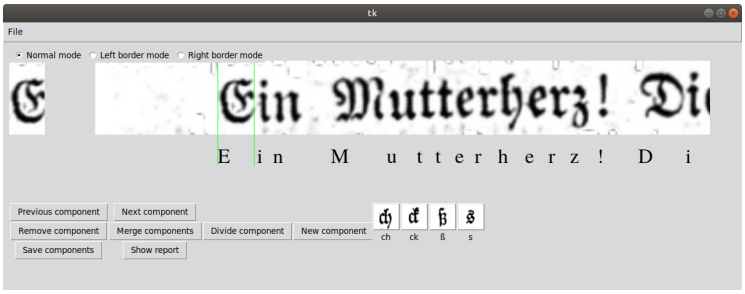

Fig. 4. Line segmenter GUI.

is a simplified version of a network proposed in [14]. Following Graves et al. [8] we use a bidirectional LSTM architecture with CTC loss function [7].

The input of our network is binarized line images. We resize the images so that its height is 40 pixels. The width is set to the maximum image width occurring in the training set. We keep the aspect ratio of the images and pad the rest of the image with white space.

CNN is utilized for extraction of features, which are fed to the LSTM network [10].

The output of the bidirectional LSTM layer is given to a set of dense layers with a softmax activation function. Its output represents a probability distribution of characters per each time frame. Let $\mathcal{A}$ be a set of symbols, which classifier recognizes ($|\mathcal{A}| = 83$). Then the $p_t^{a_i}$ is the probability of observing character a_i at given time t. In the each time t the sum of the probabilities of all symbols is equal to 1.

$$\sum_{i=1}^{|\mathcal{A}|} p_t^{a_i} = 1 \tag{1}$$

The most probable symbol $\hat{a}_t$ of each time frame t is determined as:

$$\hat{a}_t = \underset{a_i \in \mathcal{A}}{\operatorname{argmax}}\, p_t^{a_i} \tag{2}$$

The last part of the classifier is a transcription layer, which decodes the predictions for each frame into the output sequence. To be able to distinguish each individual character the blank–symbol (-) is present. It is also necessary to deduplicate the sequences of the same symbols. The architecture of the classifier is depicted in Figure 5.

Our network has two convolutional layers with 40 kernels of size 3×3. Each of them is followed by Max–pooling layer. The output of the convolutional layers is reshaped and connected to a fully–connected layer with 128 neurons. It is an input of recurrent layers. We utilize two bidirectional LSTM layers with 256 units.

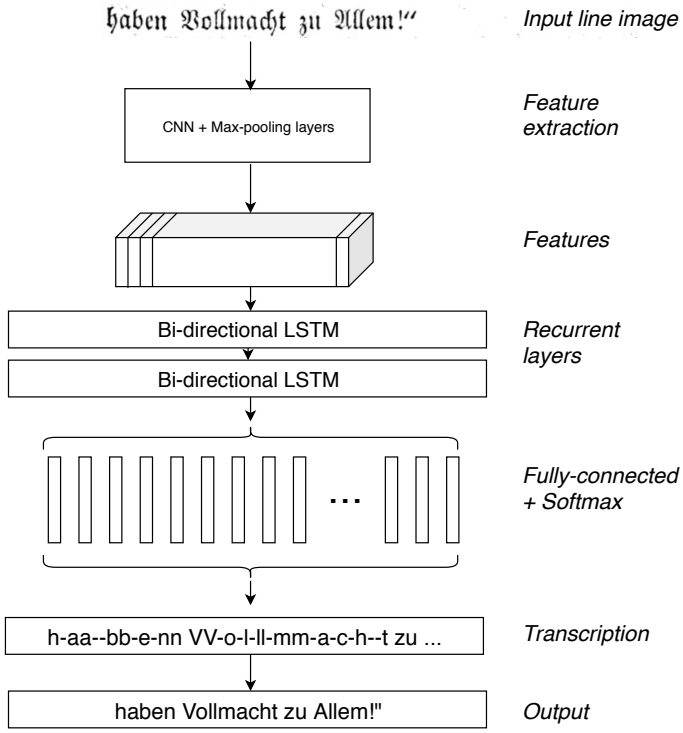

Fig. 5. Architecture of the CRNN classifier.

3.4 Classifier Training

The model is trained on data prepared from the character images. We denote such data as synthetic. The character images extracted from the initial set of lines are used for composition of line images. The texts are taken from old German archives to ensure that the language corresponds with the processed documents. Line images are then composed from the images. We use a simple approach that adds a gap of a fixed size between the characters. A wider gap is used between words. We used 25,000 line images for the initial training.

An example of such synthetic data is shown in Figure 6.

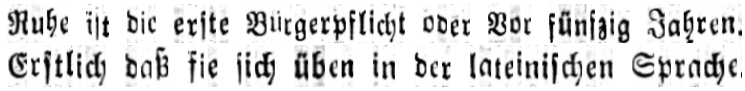

Fig. 6. Two examples of the synthetic line images.

The model is trained using stochastic gradient descent (SGD) algorithm. The initial learning rate is set to 0.005 and learning rate decay is applied. We also clip the gradients to 5. We applied early stopping based on the behavior of character error rate (CER) and validation loss. Figure 7 shows the progress of training on synthetic data for 25 epochs. One page of real data is used for validation. The performance is measured in terms of CER on the validation data. We also show training and validation loss. The curves indicate that training the model longer than for 5 epochs is not beneficial.

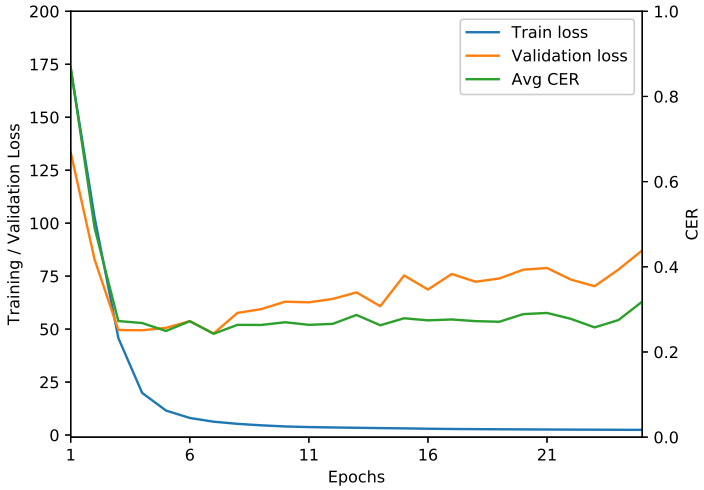

Fig. 7. Training progress on synthetic data.

To fine–tune the model we use the real annotated line images. The set comprises of 137 line images. Figure 8 depicts training loss, validation loss and CER for additional training. Based on the previous experiment we chose a model pre–trained on synthetic data for 5 epochs.

3.5 Line Annotator

This tool utilizes the trained model to predict the transcription of a line image. It is then checked by a human annotator and corrected if needed. It uses a simple GUI as shown in Figure 9.

To be able to evaluate the performance of the underlying model we measure how many corrections must be done by the user. We report Levenshtein edit distance, which indicates the number of insertions, deletions and replacements.

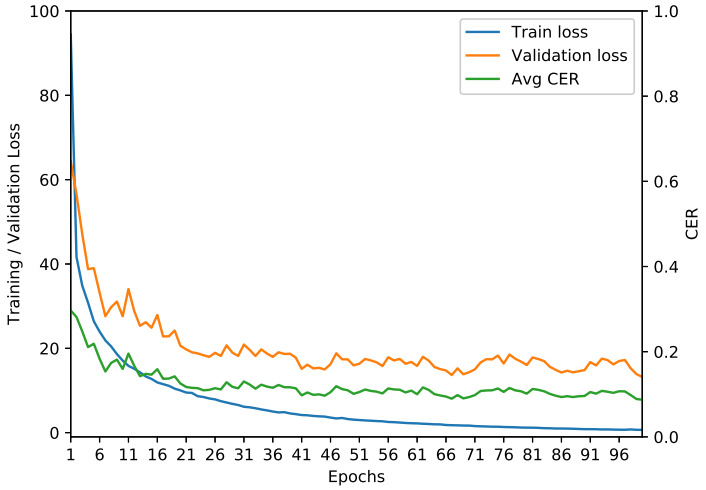

Fig. 8. Progress of the additional training on real data.

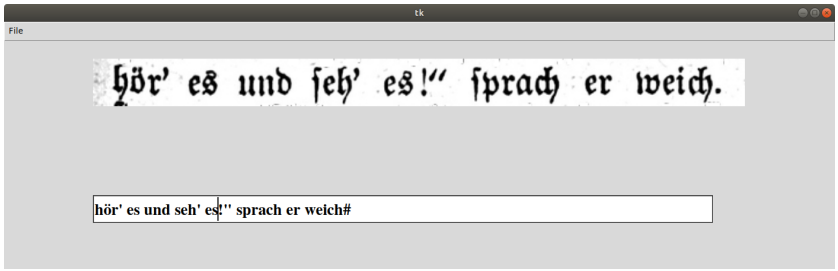

Fig. 9. Line annotation tool.

As additional information we report also CER, word error rate (WER) and normalized edit distance. The normalization is based on the ground truth sentence length. Table 1 summarizes the results of the classifier on nine additional pages that were annotated using this tool.

We can state that the tool has acceptable performance. The average number of corrections is 4 per line image.

4 Conclusion and Future Work

This work has presented a set of tools for OCR data annotation at a low cost. They will be used for preparation of training data for OCR systems. It is focused

Table 1. Evaluation of the CRNN classifier.

Metric	Value
Average edit distance	4.1
Normalized edit distance	0.088
Average WER	0.386
Average CER	0.087

mainly on historical German newspapers. The data are processed in the frame of a project that makes archival documents more accessible.

The tools are semi–automatic and require only small amount of human supervision. The main part of the whole process is the recurrent neural network model that recognizes text lines. The model is trained on data obtained from the character segmenter tool. It is utilized in the line annotator tool where it serves as recognizer of presented line images. The results obtained using model trained on a minimal set of data are satisfactory and allow very fast processing with minimal human labor.

A possibility for further improvement is a pre–processing of the images that ensures better quality of the extracted line images. It should help mainly on warped and noisy lines. A way to improve the line annotator is incremental training of the underlying model. Supplying more lines would help to create a more robust model.

Acknowledgement

This work has been partly supported from ERDF "Research and Development of Intelligent Components of Advanced Technologies for the Pilsen Metropolitan Area (InteCom)" (no.: CZ.02.1.01/0.0/0.0/17_048/0007267) and by Grant No. SGS-2019-018 Processing of heterogeneous data and its specialized applications.

References

1. Bloomberg, D.: Leptonica.[online](2010),[cit. 2010-04-25] (2010)
2. Breuel, T.M.: The ocropus open source ocr system. In: Document Recognition and Retrieval XV. vol. 6815, p. 68150F. International Society for Optics and Photonics (2008)
3. Breuel, T.M., Ul-Hasan, A., Al-Azawi, M.A., Shafait, F.: High-performance ocr for printed english and fraktur using lstm networks. In: Document Analysis and Recognition (ICDAR), 2013 12th International Conference on. pp. 683–687. IEEE (2013)
4. Chen, K., Seuret, M., Wei, H., Liwicki, M., Hennebert, J., Ingold, R.: Ground truth model, tool, and dataset for layout analysis of historical documents. In: Document Recognition and Retrieval XXII. vol. 9402, p. 940204. International Society for Optics and Photonics (2015)

5. Clausner, C., Pletschacher, S., Antonacopoulos, A.: Aletheia-an advanced document layout and text ground-truthing system for production environments. In: Document Analysis and Recognition (ICDAR), 2011 International Conference on. pp. 48–52. IEEE (2011)
6. Clausner, C., Pletschacher, S., Antonacopoulos, A.: Efficient ocr training data generation with aletheia. Proceedings of the International Association for Pattern Recognition (IAPR), Tours, France pp. 7–10 (2014)
7. Graves, A., Fernández, S., Gomez, F., Schmidhuber, J.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: Proceedings of the 23rd international conference on Machine learning. pp. 369–376. ACM (2006)
8. Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A novel connectionist system for unconstrained handwriting recognition. IEEE transactions on pattern analysis and machine intelligence **31**(5), 855–868 (2009)
9. Graves, A., Schmidhuber, J.: Offline handwriting recognition with multidimensional recurrent neural networks. In: Advances in neural information processing systems. pp. 545–552 (2009)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
11. Kanungo, T., Lee, C.H., Czorapinski, J., Bella, I.: Trueviz: a groundtruth/metadata editing and visualizing toolkit for ocr. In: Document Recognition and Retrieval VIII. vol. 4307, pp. 1–13. International Society for Optics and Photonics (2000)
12. Lee, C.Y., Osindero, S.: Recursive recurrent nets with attention modeling for ocr in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2231–2239 (2016)
13. Rawls, S., Cao, H., Kumar, S., Natarajan, P.: Combining convolutional neural networks and lstms for segmentation-free ocr. In: Document Analysis and Recognition (ICDAR), 2017 14th IAPR International Conference on. vol. 1, pp. 155–160. IEEE (2017)
14. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE transactions on pattern analysis and machine intelligence **39**(11), 2298–2304 (2017)
15. Van Beusekom, J., Shafait, F., Breuel, T.M.: Automated ocr ground truth generation. In: Document Analysis Systems, 2008. DAS'08. The Eighth IAPR International Workshop on. pp. 111–117. IEEE (2008)