A Two-Step Approach for Automatic OCR Post-Correction

Robin Schaefer
Staatsbibliothek zu Berlin -
Preußischer Kulturbesitz
10785 Berlin, Germany
robin.schaefer
@sbb.spk-berlin.de

Clemens Neudecker
Staatsbibliothek zu Berlin -
Preußischer Kulturbesitz
10785 Berlin, Germany
clemens.neudecker
@sbb.spk-berlin.de

Abstract

The quality of Optical Character Recognition (OCR) is a key factor in the digitisation of historical documents. OCR errors are a major obstacle for downstream tasks and have hindered advances in the usage of the digitised documents. In this paper we present a two-step approach to automatic OCR post-correction. The first component is responsible for detecting erroneous sequences in a set of OCRed texts, while the second is designed for correcting OCR errors in them. We show that applying the preceding detection model reduces both the character error rate (CER) compared to a simple one-step correction model and the amount of falsely changed correct characters.

1 Introduction

The digitisation of historical documents is a central objective for libraries, archives and museums. Using OCR, the digitised documents can be converted to electronic full text. Important advances have been achieved in OCR over recent years, mainly through the use of neural networks (Reul et al., 2018). However, in the case of historical documents there still remains a substantial amount of OCR errors that hinders the usage of the digitised documents and requires efficient methods for OCR post-correction. By defining OCR post-correction as a translation problem, current approaches are often based on neural networks (Mokhtar et al., 2018), especially sequence-to-sequence models that are inspired by successes achieved in neural machine translation (NMT) (Bahdanau et al., 2014; Luong et al., 2015). While these models perform well for data sets with a high amount of OCR errors, post-correction becomes more difficult when the CER is less severe (Amrhein and Clematide, 2018).

In this paper we present an alternative approach to OCR post-correction. Instead of solving the task in one step, i.e. by training a single translation model[1], we propose a two-step approach that avails itself of two separate models. We chose a Long Short Term Memory (LSTM) architecture (Hochreiter and Schmidhuber, 1997) for the actual sequence-to-sequence translation step (henceforth called *translator*). In addition, we inserted a model before the translator that functions as a filter by separating erroneous sequences[2] from correct ones (henceforth called *detector*). The benefits of this approach are two-fold: 1) By decreasing the proportion of correct OCRed data, the CER of the data fed into the translator is artificially increased. This is assumed to improve the translation results.[3] 2) By excluding the majority of correct sequences from the translation step, we can avoid to mistakenly insert errors in them.

This paper is structured as follows: Section 2 briefly summarizes the relevant previous work. Section 3 presents an initial OCR post-correction approach using a standard sequence-to-sequence model. It also describes the used data and the applied preprocessing steps. Afterwards, Section 4 presents our alternative two-step approach to OCR post-correction. Both components are analysed independently using different test sets. In Section 5 we show first results of the full pipeline applied to one data set. Finally, in Section 6 we discuss our results and give a short outlook.

[1]Although, technically speaking, a sequence-to-sequence model consists of two model components, we are calling this approach *one-step* as both components are trained simultaneously.

[2]We use the terms *sequence* and *line* interchangeably.

[3]It is crucial to note that the CER gets increased *only* by removing correct sequences with the detector model. At no point do we manually insert artificial OCR errors.

2 Related Work

Traditionally, the problem of OCR post-correction has been approached by the application of statistical language modelling techniques, frequently in combination with lexical resources or n-grams (Tong and Evans, 1996). String distance metrics are frequently used to generate correction candidates for erroneous tokens, aided by a probabilistic scoring mechanism or noisy channel model (Kolak and Resnik, 2002). However, these methods are less successful for historical documents that exhibit large amounts of OCR errors, historical spelling variation and where fitting language resources are scarce (Piotrowski, 2012).

Accordingly, Reffle and Ringlstetter (2013) compute a two-channel profile for historical language variation and OCR errors to improve the correction rate while minimizing the injection of new errors. This method was further improved by Fink et al. (2017) with the addition of an adaptive feedback mechanism and the treatment of non-interpretable tokens as conjectured errors. An alternative technique is Anagram Hashing (Reynaert, 2008), but more recent evaluations indicate that this approach requires fine-tuning (Reynaert, 2016). Since different OCR engines (or models) also produce different OCR error patterns, another strategy is merging multiple OCR results, supported by heuristics and lexical resources (Volk et al., 2011). OCR post-correction has also been studied as a problem of statistical (Afli et al., 2016) or neural machine translation (Mokhtar et al., 2018; Hämäläinen and Hengchen, 2019).

An overview of the recent state-of-the-art in OCR post-correction is provided by the ICDAR competition on post-OCR text correction. The competition comprises of two tasks, error detection and error correction, with evaluation sets for English and French language. The 2017 edition saw Weighted Finite-State Transducers (WFST) obtain best results in the error detection task, with an SMT/NMT system achieving the highest reduction of errors (Chiron et al., 2017). In the 2019 edition, a system based on BERT (Devlin et al., 2018) came out best for both tasks (Rigaud et al., 2019). For comparison, we refer to Amrhein and Clematide (2018) as our baseline (see Section 6).

3 The Standard Sequence-to-Sequence Approach

We first approached the OCR post-correction task by implementing a standard LSTM-based sequence-to-sequence translation model using the PyTorch framework (Paszke et al., 2019). The architecture consists of an encoder model responsible for calculating a matrix representation of the OCRed input sequence. This is used by the subsequent decoder model for creating a correct output sequence.

Data. Using the OCR-D pipeline (Neudecker et al., 2019) [4], we processed 35 German works that were published from the 17th to the 19th century. All works are part of the corpus created by the *Deutsches Textarchiv* ("German Text Archive", DTA)[5], which contains double-keyed transcriptions, i.e. ground truth (GT). We aligned lines of GT and the OCRed documents using *dinglehopper*[6], before we calculated the CER for each individual OCR sequence. Furthermore, we removed sequence pairs with an error rate of more than 10%, which resulted in a data set of 167,848 line pairs. Given that experiments with the full data set have shown negative results, we consider this removal step necessary for training the translation model. It is important to note that the removed sequence pairs contain incorrect alignments, which we assume to be detrimental for the training process.

Following Amrhein and Clematide (2018) we applied a sliding window approach in order to 1) increase the data set size and 2) increase the importance of a token's direct context. To this end, we defined a context window of two preceding and one succeeding words around a given token (i.e. context-context-token-context). We incrementally moved this window one token at a time. Context windows never crossed page boundaries. This procedure increased our data set to 707,427 line pairs.

After this reformatting step, we removed non-German sequences using *langid* (Lui and Baldwin, 2012). The remaining line pairs were split into training (365,000 lines), validation (56,800 lines) and test sets (56,800 lines). All sets were encoded using a greedy approach inspired by the encoding mechanism used in the BERT framework (Devlin et al., 2018). Each 3-, 2- and 1-character combination occurring

[4]Specifically, we use the implementation from `https://github.com/qurator-spk/ocrd-galley`.

[5]`http://www.deutschestextarchiv.de/`

[6]`https://github.com/qurator-spk/dinglehopper`

in the GT received an encoding value. Referring to this character combination-encoding mapping, we encoded the OCR sequences. Whenever a 3-character encoding encountered in the OCR data did not exist in the mapping, a 2- or, if needed, 1-character encoding was chosen. Finally, the encoded sequence vectors were 0-padded to the length of 40.

Model Architecture and Results. Both encoder and decoder models had one layer and a hidden node size of 256. The decoder further included an attention mechanism (Bahdanau et al., 2014). We trained the sequence-to-sequence model monodirectionally for 970 epochs using a learning rate of 0.0001 and a batch size of 200. In order to facilitate the task of the decoder, we made use of the teacher forcing technique (Williams and Zipser, 1989) using a ratio of 0.5. Applying the trained translator model on the test set, we achieved a CER of 1.6%. As the original CER in the test set amounts to 1.2%, this model actually increases the error rate. Referring to Amrhein and Clematide (2018), we argue that this model is not capable of efficiently correcting the low CER of this data set.

4 The Two-Step Approach

In order to improve the translator model's results, we decided on building a two-step pipeline. This pipeline consists of an initial OCR error detection model and a subsequent error translation (i.e. correction) model.[7] In this architecture, only sequences that the detector considers erroneous would be forwarded to the translator model. This procedure has the added benefit of artificially increasing the proportion of OCR errors which are fed into the translator. It also has the potential of decreasing the amount of erroneously changed correct characters.

4.1 The Detector Model

A successful detector model is characterised by a low false positive rate, i.e. by a small number of lines incorrectly classified as erroneous. Hence, we are particularly interested in increasing the detector's precision. As building a model that barely inflicts harm upon the correct data is of utmost importance, we do not prioritize a high recall, which indicates the proportion of identified incorrect sequences.

We developed a model based on a bidirectional LSTM architecture that sequentially processes an OCR sequence and outputs for every character encoding probabilities of it being erroneous or correct with respect to the GT encoding. Experiments showed that using a 3-layer structure with a hidden size of 512 yielded best results. A linear layer was set on top of the LSTM architecture, which reduced the high dimensional tensors to the output size of 3 (correct character/s, incorrect character/s, 0-padding). Logits were inputted into a softmax layer to obtain probabilities. We further used a dropout probability of 20% between LSTM layers. We trained the model for 138 epochs using a batch size of 200. Sequence-wise labels were calculated as follows: 1) Only character encodings with an error probability of $>99\%$ were treated as erroneous. 2) All sequences with at least one incorrect encoding were labelled as incorrect.

Data. We used the data[8] described in Section 3 and applied the same preprocessing steps. However, instead of directly encoding the GT data, we used it to create class targets for the LSTM model. The OCR data was again encoded using the greedy encoding mechanism and padded with 0s.

	Predicted negative	**Predicted positive**
Target negative	41386	1123
Target positive	3730	10561

Table 1: Confusion Matrix (positive = incorrect sequence; negative = correct sequence)

Results. Applying the trained detector model on the test set yielded promising results. We obtained an F1 score of 81%, a precision of 90% and a recall of 74%. In line with the objective described in this section, we obtained a high precision score and low false positive rate. This is confirmed by Table 1

[7]The code (two-step approach): `https://github.com/qurator-spk/sbb_ocr_postcorrection`.

[8]The data (two-step approach): `https://zenodo.org/communities/stabi/`.

(see column *Predicted positive*). Notably, we also achieved an acceptable recall, which indicates that a substantial proportion of incorrect sequences is also classified as such. Hence, the majority of incorrect sequences gets actually passed on to the translator model.

4.2 The Translator Model

The second component of our OCR post-correction pipeline is a sequence-to-sequence translation model as described in Section 3. For the sake of comparison of both approaches we refrained from making changes to the model architecture or training parameters. The model was trained for 876 epochs.

Data. We processed another 28 works using the OCR-D pipeline. After applying the same preprocessing steps as described in Section 3, the data set included 467,044 line pairs. As the translator model is to be applied on sequences classified as erroneous by the detector model, we needed to perform an additional step for the creation of training, validation and test sets. Keeping in mind that the detector model's precision is 90%, this means that we can expect 10% of the lines classified as erroneous to be actually correct. To cope with these misjudgements, we randomly added correct sequences to the erroneous lines such that each set included approximately 10% correct sequences.

However, this step severely reduced the overall size of the data set. In order to create a sufficiently large set for training, we used the vast majority of sequences to this end (196,000). As this decision resulted in small validation and test sets, we enhanced them using additional data from the respective sets used for validating and testing the detector model (final size of both sets: 22,000).

Results. Applying the translator model to the test set resulted in a reduced CER compared to the original error rate of the OCRed data. We achieved a CER of 3.6%, which is a reduction of 0.7% to the former 4.3% in the uncorrected data.

5 The Full OCR Post-Correction Pipeline

As both components have been analysed independently in Section 4, we now present results yielded by applying the full two-step pipeline on a combination of both test sets (83,600). As the detector's task is the reduction of correct sequences in the data that is to be fed into the translator, we did not remove any correct sequences manually.

Results (Detector). Applying the detector model on the test set yielded an F1 score of 79%, a precision of 87% and a recall of 72%. Although these scores are somewhat reduced compared to the results obtained by the detector if used on the first test set alone, we argue that the scores are appropriately similar to proceed with OCR post-correction using the translator model. 19,800 of the 83,600 sequences were classified as erroneous and, hence, were forwarded to the translator.

Results (Translator). Calculating the mean CER over the full test set resulted in a score of 1.1%. In order to measure the full test set's CER after the translation step has been conducted, we first applied the translator component on the sequences classified as erroneous and then combined the corrected sequences with the lines previously classified as correct. Calculating the CER on the corrected test set yielded a score of 0.9%, which is a relative improvement of 18.2%. For reasons of comparison, we fed the same test set into our one-step correction model, which achieved a CER of 2.1%. In addition, for both approaches we calculated the rates of correct characters being erroneously changed. While our two-step architecture achieved a score of 0.3%, 6% of characters were falsely changed by our one-step model.

6 Discussion and Outlook

In this paper, we presented an alternative approach to automatic OCR post-correction using a two-step architecture consisting of a detector and a translator model. While the detector model identifies erroneous sequences in an OCR data set, the translator model is used to correct these sequences. This approach has two major benefits. First, removing correct sequences from the to-be-corrected data set increases the proportion of OCR errors fed into the translator model. This is supposed to improve the translator's results. Second, a detector model with a high precision limits the amount of correct sequences processed by the translator, thereby reducing the likelihood of correct characters being erroneously changed. As a successful OCR post-correction model is supposed to leave the correct data unaltered, the importance of the second aspect should not be underestimated.

Comparisons of both approaches' results show that by inserting the detector model the translator's results can be substantially improved. Whereas the simple one-step translator model actually increases the CER from 1.1% to 2.1%, we were able to reduce the error rate to 0.9% by applying the two-step architecture. This is a relative improvement of 18.2%. While this CER improvement is smaller than the score achieved for the English monographs (pre-correction CER: 1.8%, rel. improvement: 31.3%) in Amrhein and Clematide (2018), their improvement achieved for the French monographs (pre-correction CER: 1.6%, rel. improvement: 18.3%) is similar to our result.[9] We are aware that it is difficult to directly compare our results with Amrhein and Clematide (2018) given the differences between the two studies (German vs English/French data, differences in the model hyperparameters, notable differences in pre-correction CER). However, we consider the partially similar results as a promising indicator that our two-step model can actually compete with previously proposed methods.

While the CER reduction is noteworthy in itself, a special focus should be laid on the rates of erroneously changed correct characters. The two-step approach yielded a promising rate of changed correct characters of 0.3%. However, correcting the data set with the one-step model resulted in a high rate of 6%, which indicates that the correct sub-parts of the data have been severely modified. Given that we did not use different model parameters for training both sequence-to-sequence translators, we can conclude that this reduction is associated with the insertion of the detector. In other words, our results can be interpreted as evidence for the assumption that a low CER poses a notable problem for a one-step translation model.[10]

As the detector component already strongly improves the OCR post-correction results, our future work will primarily focus on the further development of the translator component. On the one hand, we plan to improve the attention mechanism implemented in the translator's decoder model. As OCR errors are characterized by their local context, using a local attention approach (Luong et al., 2015) may have benefiting effects on OCR post-correction. Furthermore, we will investigate alternatives to the LSTM architecture. In particular, we consider the application of Generative Adversarial Networks (Goodfellow et al., 2014) as a fruitful path of future research.

Acknowledgements

This work was partially supported by the German Federal Ministry of Education and Research (Bundesministerium für Bildung und Forschung, BMBF), project grant QURATOR - Curation Technologies (Unternehmen Region, Wachstumskern, grant no. 03WKDA1A)[11]. We thank the anonymous reviewers for their helpful comments.

References

Haithem Afli, Loïc Barrault, and Holger Schwenk. 2016. Ocr error correction using statistical machine translation. *Int. J. Comput. Linguistics Appl.*, 7(1):175–191.

Chantal Amrhein and Simon Clematide. 2018. Supervised ocr error detection and correction using statistical and neural machine translation methods. *Journal for Language Technology and Computational Linguistics (JLCL)*, 33(1):49–76.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Guillaume Chiron, Antoine Doucet, Mickaël Coustaty, and Jean-Philippe Moreux. 2017. Icdar2017 competition on post-ocr text correction. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 1423–1428. IEEE.

[9]Presented results of Amrhein and Clematide (2018) are referred to as *NMT context* in their paper.

[10]Importantly, our results can also be interpreted as counter-evidence against the hypothesis that the one-step translation model merely failed due to an inappropriate setting of hyperparameters.

[11]`https://qurator.ai`

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Florian Fink, Klaus U Schulz, and Uwe Springmann. 2017. Profiling of ocr'ed historical texts revisited. In *Proceedings of the 2Nd International Conference on Digital Access to Textual Cultural Heritage*, pages 61–66.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 2672–2680, Cambridge, MA, USA. MIT Press.

Mika Hämäläinen and Simon Hengchen. 2019. From the paft to the fiiture: a fully automatic nmt and word embeddings method for ocr post-correction. *arXiv preprint arXiv:1910.05535*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November.

Okan Kolak and Philip Resnik. 2002. Ocr error correction using a noisy channel model. In *Proceedings of the second international conference on Human Language Technology Research*, pages 257–262. Morgan Kaufmann Publishers Inc.

Marco Lui and Timothy Baldwin. 2012. langid.py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations*, pages 25–30, Jeju Island, Korea, July. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September. Association for Computational Linguistics.

Kareem Mokhtar, Syed Saqib Bukhari, and Andreas Dengel. 2018. Ocr error correction: State-of-the-art vs an nmt-based approach. In *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, pages 429–434. IEEE.

Clemens Neudecker, Konstantin Baierer, Maria Federbusch, Matthias Boenig, Kay-Michael Würzner, Volker Hartmann, and Elisa Herrmann. 2019. Ocr-d: An end-to-end open source ocr framework for historical printed documents. In *Proceedings of the 3rd International Conference on Digital Access to Textual Cultural Heritage*, DATeCH2019, pages 53—–58, New York, NY, USA. Association for Computing Machinery.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037.

Michael Piotrowski. 2012. Natural language processing for historical texts. *Synthesis lectures on human language technologies*, 5(2):1–157.

Ulrich Reffle and Christoph Ringlstetter. 2013. Unsupervised profiling of ocred historical documents. *Pattern Recognition*, 46(5):1346–1357.

Christian Reul, Uwe Springmann, Christoph Wick, and Frank Puppe. 2018. State of the art optical character recognition of 19th century fraktur scripts using open source engines. *arXiv preprint arXiv:1810.03436*.

Martin Reynaert. 2008. Non-interactive ocr post-correction for giga-scale digitization projects. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 617–630. Springer.

Martin Reynaert. 2016. Ocr post-correction evaluation of early dutch books online-revisited. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 967–974.

Christophe Rigaud, Antoine Doucet, Mickaël Coustaty, and Jean-Philippe Moreux. 2019. Icdar 2019 competition on post-ocr text correction. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1588–1593. IEEE.

Xiang Tong and David A Evans. 1996. A statistical approach to automatic ocr error correction in context. In *Fourth Workshop on Very Large Corpora*.

Martin Volk, Lenz Furrer, and Rico Sennrich. 2011. Strategies for reducing and correcting ocr errors. In *Language technology for cultural heritage*, pages 3–22. Springer.

Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Comput.*, 1(2):270–280, June.