OCR Processing of Swedish Historical Newspapers Using Deep Hybrid CNN–LSTM Networks

Molly Brandt Skelbye
Dept. of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
412 96 Gothenburg, Sweden
molly.brandt09@gmail.com

Dana Dannélls
Språkbanken Text, Dept. of Swedish
University of Gothenburg
405 30 Gothenburg, Sweden
dana.dannells@svenska.gu.se

Abstract

Deep CNN–LSTM hybrid neural networks have proven to improve the accuracy of Optical Character Recognition (OCR) models for different languages. In this paper we examine to what extent these networks improve the OCR accuracy rates on Swedish historical newspapers. By experimenting with the open source OCR engine Calamari, we are able to show that mixed deep CNN–LSTM hybrid models outperform previous models on the task of character recognition of Swedish historical newspapers spanning 1818–1848. We achieved an average character accuracy rate (CAR) of 97.43% which is a new state–of–the–art result on 19th century Swedish newspaper text. Our data, code and models are released under CC BY licence.

1 Introduction

Making previously inaccessible historical data available by converting them into digital resources through Optical Character Recognition (OCR) techniques is important for preserving our cultural heritage and thereto provide greater insight into the past. Today there are several leading off-the-shelf OCR engines such as the commercial ABBYY FineReader,[1] open source OCRopus (Breuel, 2008), Tesseract (Smith, 2007) and Calamari (Wick et al., 2020), offering a comprehensive set of pre–trained models that could be applied during recognition for different languages. These models, in particular models for historical data, vary in accuracy for some languages mainly because they were trained on very limited amount of data. To overcome this limitation researchers have been training language specific character recognition models for different time periods (Furrer and Volk, 2011; Breuel et al., 2013; Krishna et al., 2018; Drobac et al., 2019). There have also been attempts to improve the accuracy of the models after recognition by applying post-correction methods (Drobac and Lindén, 2020; Dannélls and Persson, 2020). In this paper we examine to what extent deep CNN–LSTM hybrid neural networks can improve the character accuracy rate (CAR) on 19th century Swedish newspaper text during recognition. Following Drobac and Lindén (2020) approach we trained a character model for Swedish in Calamari and achieved an average CAR of 97.43% which is a new state–of–the–art result for historical Swedish newspaper text.[2]

2 Related Work

Mainly due to the introduction of recurrent neural networks (RNNs) in particular with the Long Short–Term Memory (LSTM) architecture (Breuel, 2017) great progress has been made in the field of OCR in recent years. Today most of modern OCR systems leverage deep learning algorithms for training models and to improve performance. For instance some state-of-the-art OCR systems use shallow LSTM neural networks, consisting of one or two hidden layers (Breuel, 2008). LSTM models became a standard in 2013 (Breuel et al., 2013) and LSTM-based networks have proven to be one of the most effective approaches for complex natural language processing (NLP) tasks (Hochreiter and Schmidhuber, 1997; Alom et al., 2019). Likewise Convolutional Neural Networks (CNNs) have shown outstanding results for image data processing tasks (Krizhevsky et al., 2017) including

[1] https://pdf.abbyy.com

[2] Data, code and models are released under CC BY licence: https://github.com/mskelb/EXJOBB

Proceedings of Recent Advances in Natural Language Processing, pages 190–198
Sep 1–3, 2021.
https://doi.org/10.26615/978-954-452-072-4_023

feature extraction (Lecun and Bengio, 1995). Hybrids of these two network structures are used in a high diversity of fields and have achieved state-of-the-art results in many cases (Wick et al., 2018).

Breuel (2017) presented a groundbreaking deep hybrid CNN–LSTM implementation for text recognition, that outperformed previously state of the art methods based on shallow LSTMs. Reul et al. (2017) show that transfer learning drastically improves character accuracy rates on early printed books, compared to when training models from scratch. A year later this have become the default approach for training models. In their continued research Reul et al. (2018) utilize a combination of cross-fold training and confidence voting and succeed to significantly reduce character error rates on early printed books, compared to training a single model on a single fold. Along the same lines, in our work we also train an ensemble of models on a cross-fold of the same GT data and then combine the models through voting.

Drobac and Lindén (2020) and Wick et al. (2018) approach the complex task of OCR for historical prints by utilizing these types of deep CNN–LSTM hybrid networks. To improve OCR results for historical Finnish and Swedish newspapers and journals Drobac and Lindén (2020) train mixed–language models and after post–processing of the OCR output achieve 1.7% CER for the Finnish, and 2.7% CER for the Swedish test set from 1771 to 1874. Unlike Drobac and Lindén (2020) we do not apply any post-processing method but rather focusing on increasing the accuracy of the Swedish character model.

In Wick et al. (2018) error–rates are successfully reduced to a factor of up to 55% for digitised historical texts from the ICDAR 2017 dataset, achieving an average CER of 1.5%. To further improve these results, confidence voting was applied, resulting in CER below 0.5%. Moreover, Wick et al. (2018) show these types of deep neural networks significantly outperform shallow networks in terms of both recognition capabilities and speed.

3 Calamari Deep CNN–LSTM Hybrid Networks

Calamari is a high–performance Tensorflow–based package for line based recognition using state of the art Deep Neural Networks (DNNs) (Wick et al., 2020).[3] The advantage of Calamari is that the software supports customized deep network architectures composed by Convolutional Neural Networks (CNNs) and Long Short–Term Memory (LSTM) layers, trained by the Connectionist Temporal Classification (CTC) algorithm described in Graves et al. (2006). It uses the Tensorflow 3 framework for deep neural network computations, and consequently supports training and recognition on the graphics processing unit (GPU), which is proven to significantly reduce the overall computation time (Wick et al., 2020).

Calamari also provides additional features that might improve accuracy rates, such as pre–training, early stopping, cross–fold training, data argumentation, and confidence voting of different predictions. However, while data augmentation has been shown to improve accuracy for small data sets, it impairs the accuracy for larger data sets according to Wick et al. (2020). Moreover, Calamari with its deep network architecture has proven to significantly outperforms the shallow one–dimensional LSTM based neural network approach, such as the one used by OCRopus (Breuel, 2008) in terms of both recognition capabilities and speed (Drobac and Lindén, 2020; Wick et al., 2018).

The default neural network structure consists of two consecutive CNN blocks containing a convolution and a max–pooling layer, both connected with a ReLU–activation function. Each of the two convolution layers has a kernel size of 3 x 3, the first one consists of 40 filters, and the second one of 60 filters. The pooling layer has a kernel size and stride of 2 x 2. This is followed by a bidirectional LSTM layer, and finally an output layer with a drop–out rate of 0.5 in order to prevent overfitting. Given the predictions of the output–layer and the GT labels, the CTC–loss is computed.

4 Data and Preprocessing

The reference data we experimented with forms part of the KubHist data (Adesam et al., 2019). It is a large collection of approximately 300 thousand Swedish historical newspaper editions, spread over roughly 200 years. The collection has been digitised and OCRed by the National Library of Sweden.

Part of the KubHist, more specifically a reference data of 400 pages spanning between 1818 and 2018, was processed through an enhanced OCR–process (Dannélls et al., 2019) by combin-

[3] https://www.tensorflow.org/

ing two OCR engines: Abbyy FineReader,[4] and the open source system Tesseract.[5] This two–OCR engine system was originally developed in cooperation with the Norwegian software company Zissor in 2017 and is based on the principle of evaluating and comparing the OCR results from multiple engines. The approach has been proven to improve the character recognition accuracy for some newspapers (Dannélls et al., 2020). Many errors however remain. The reference data comes with a ground truth (GT) data available in plain text files.[6] It was first segmented down to paragraph level,[7] and further manually transcribed through double–keying.[8]

This reference material has been previously used to evaluate the two–OCR engine system, focusing on the time period 1818–2018. The evaluation results show that accuracy varies from 56,65% to 98,41%, on character level, depending on the newspaper edition. These results further constitute the very baseline of this project.

4.1 Challenges

The challenge of the KubHist corpus lies in the amount of OCR errors that are currently very high in some parts of the collection. Especially texts from the early 19th century where it is estimated that most of the text (roughly 75%) is printed in Blackletter.[9] The Blackletter typeface is generally challenging for OCR systems to recognize due to the many font variations, low distinctiveness of characters, and in many cases, lack of training data of acceptable quality (Holley, 2009; Furrer and Volk, 2011). Large part of the OCR errors is a result of the low accuracy of the pre–trained language models that were trained on a limited amount of Swedish data and were used in the OCR process.

4.2 Data Preprocessing

To test the performance of our mixed deep CNN–LSTM hybrid models, part of the available reference material has been used for training, validation and testing respectively. This dataset we experimented with contains two pages from each newspaper edition during the time period 1818–1848. In all 30 newspaper editions, one from each year, and 60 newspaper pages that have been further pre–processed through image binarization and de–skewing using the ocropus-nlbin script provided by OCRopus.[10][11] For the binarization procedure adaptive thresholding has been applied. All newspaper pages have been further re–segmented into text line images using the ocropus-gpageseg script,[12] also provided by OCRopus, comprising a total of 8 413 lines, 67 441 words and 423 414 characters. To maintain the data as clean as possible incorrectly segmented text lines as well as non textual content such as vertical and horizontal lines etc. have been manually removed.

5 Experiments and Results

Our strategy for dividing the dataset into training and test sets has been to randomly selecting lines from each newspaper page. 80% has been allocated for training and 20% for testing. This way of randomizing generates a good diverse training data and hence a good representation of the data that is important for constructing a model able to generalize well (Drobac and Lindén, 2020). In total, 6 742 lines containing 53 963 words and 338 701 characters have served as training set, 1 671 lines containing 13 478 words and 84 713 characters as test set. For training of a single model (single voter) the final training set has been further split into training and validation subsets, again with a split ratio of 80:20. In total the generated training subset comprises 5 400 lines and the validation set 1 349 lines. More details can be found in Table 1.

Dataset	GT lines	Words	Characters	B:A
Training	6 742	53 963	338 701	75% : 25%
training set	5 400	N/A	N/A	
validation set	1 349	N/A	N/A	
Testing	1 671	13 478	84 713	75% : 25%
Total	8 413	67 441	423 414	

Table 1: Details of the training, validation and test sets. B:A is the distribution of Blackletter and Antiqua.

[4] Abbyy FineReader version 11.1.16.

[5] Tesseract version 4 that is based on a CNN–LSTM hybrid structure developed by Google.

[6] Another way to save the data is in XML files, while keeping tack of the layout and segmentation in the document, unfortunately this kind of information is lacking.

[7] However not line level.

[8] The double–keying process requires two independent transcriptions of the same material, that are later compared to each other in order to detect transcription errors. This is a method that ensures very high accuracy rates (Susanne Haaf, 2013).

[9] After mid 20th century, Blackletters (including Fraktur) was gradually replaced by standard Antiqua–based typefaces.

[10] https://github.com/ocropus/ocropy/blob/master/ocropus-nlbin

[11] OCRopus is a collection of document analysis programs, https://github.com/ocropus/ocropy.

[12] https://github.com/ocropus/ocropy/blob/master/ocropus-gpageseg

Once the line images have been prepared we use them together with their ground truth to train models with Calamari. During training early stopping has been applied in order to avoid overfitting and improve generalization. If performance on the held out validation dataset after each epoch has not improved after 5 epochs or if validation loss has begun increasing training was stopped. One approach for reducing overfitting is to add more examples to the training data. However, since segmenting newspaper pages into image lines and linking them to their corresponding GT lines has turned out to be relatively time consuming. Adding more data in order to investigate whether results can be improved has not been done.

To improve performance and find the optimal neural network structure for our data a series of experiments have been performed using different network configurations. These include testing out different combinations and dimensions of the CNN and LSTM layers and in this way expanding the network. These experiments are described in the following sections. In all our experiments both single models are trained as well as using a combination of cross-fold training and subsequent confidence voting. Training was performed using a 5–fold over the same training data resulting in 5 models with different characteristics. These 5 models are then used to recognize the held out test lines. For each test line a total of 5 output sequences are generated, 1 from each model. These 5 output sequences later serve as the input for the voting process in order to determine the final output by using confidence scores.

After experimenting with different network configurations both by training single models and by combining cross-fold training and confidence voting we found that the best results were achieved using the following neural network, seen in Equation 1.

$$\begin{aligned} &\textit{cnn = 80 : 3 x 3, pool = 2 x 2,}\\ &\textit{cnn = 100: 3 x 3, pool = 2 x 2,}\\ &\textit{lstm = 200, dropout = 0.5,}\\ &\textit{lstm = 200, dropout = 0.5.} \end{aligned} \tag{1}$$

The optimized neural network consists of two consecutive CNN blocks containing a convolution layer followed by a max–pooling layer. Both connected with an ReLU–activation function. Each of the two convolution layers have a kernel size of 3 x 3. The first one consists of 80 filters and the second one of 100 filters. The pooling layer has a kernel size and stride of 2 x 2. The two CNN blocks are followed by two LSTM layers containing a total of 400 nodes divided over two layers with 200 nodes each.

5.1 Experimental Setup

We ran four experiments using the following model combinations:

M1: single model trained using Drobac and Lindén (2020) neural network in Equation 3.
M2: single model trained using Calamari's default neural network in Equation 2.
M3: single model trained using the optimized neural network in Equation 1.
M4: 5 best models trained using cross-fold training with the same optimized neural network in combination with subsequent confidence voting.

Experiment I In this initial experiment we train a single model using Calamari's default neural network as specified in Equation 2

$$\begin{aligned} &\textit{cnn = 40 : 3 x 3, pool = 2 x 2,}\\ &\textit{cnn = 60: 3 x 3, pool = 2 x 2,}\\ &\textit{lstm = 200, dropout = 0.5.} \end{aligned} \tag{2}$$

Experiment II We further train a single model following the same network architecture as in Drobac and Lindén (2020) and in Equation 3. Similar to the default Calamari network this network contains two consecutive CNN blocks containing a convolution and a max–pooling layer. Both are connected with a ReLU–activation function as can be seen in Equation 4.2. Each of the two convolution layers has a kernel size of 3 x 3. The first one consists of 128 filters and the second one of 128 filters. The pooling layer has a kernel size and stride of 2 x 2. In addition, the LSTM layers contain a total of 1 200 nodes divided over two layers with 600 nodes each. Here we want to investigate how performance changes using a deeper network.

$$\begin{aligned} &\textit{cnn = 128 : 3 x 3, pool = 2 x 2,}\\ &\textit{cnn = 128: 3 x 3, pool = 2 x 2,}\\ &\textit{lstm = 600, dropout = 0.5.}\\ &\textit{lstm = 600, dropout = 0.5.} \end{aligned} \tag{3}$$

Experiment III In this experiment we want to find the optimal neural network for our data. In

Table 2: CAR results for the four different models. Respective model combinations: M1–Drobac, M2–Calamari, M3–Single model trained with the optimized network, M4–5 best models trained with the optimized network.

	Model(s)	Dataset	CAR (%)	CER (%)	Voted
		Test set			
(1)	M1		95.55	4.45	No
(2)	M2		96.06	3.94	No
(3)	M3		96.44	3.56	No
(4)	M4		**97.43**	**2.57**	Yes
(5)	M4	50-fraktur	96.84	3.16	Yes
(6)	M4	50-antiqua	96.48	3.52	Yes

Reul et al. (2018) it was shown that a combination of cross-fold training and confidence voting led to significantly lower character error rates compared to training a single model. Therefore to test the models' ability of predicting never seen before data we have followed the same approach in our continued experiments. However, in order to investigate how much this combination can improve performance models have been trained both by (1) single training resulting in a single voter and (2) by cross-fold training resulting in 5 models.

Experiment IV This experiment investigates the transferability of models and their ability to generalize on previously unseen data. In this case unseen data is a collection consisting of journals and newspaper published in Finland between 1771 and 1874. Two relatively small datasets of text line images and their corresponding GT have been randomly sampled from the Swedish test dataset in Drobac and Lindén (2020). The first set contains 50 lines printed in Fraktur and the second 50 lines printed in Antiqua. Models have been tested on each separate test set using confidence voting. What has been interesting to see here is how mixed models behave on a specific font type.

5.2 Evaluation Metrics

Calamari uses the Character Error Rate (CER) metric defined as the "edit distance (ed) of two sequences s_1 and s_2 normalized by the maximum length" (Wick et al., 2020) as stated in Equation 4. Edit distance corresponds to the Levenshtein distance and the sequences to the text lines and the corresponding GT lines. There are two common ways of measuring OCR errors or accuracy. One of which is at the character level (CER or CAR) and the other one at the word level (WER or WAR) (Holley, 2009).

$$\text{CER} = \frac{ed(s_1, s_2)}{max(\mid s_1 \mid, \mid s_2 \mid)} \tag{4}$$

While CER is used for measuring performance of our models CAR has previously been used in the evaluation process of the two–OCR engines in Dannélls et al. (2020). In order to give a fair comparison of results a conversion has been made between CER and CAR. Moreover, when testing an ensemble of 5 models on the held out test lines we use the confidence based voting to combine them.

5.3 Results

Table 2 summarises the performance in average CAR (%) and CER (%) of the different combinations of the trained models tested on different datasets. Models in (1)–(4) have been evaluated using our test data. Models in (5) have been evaluated using 50 lines of Fraktur and in (6) 50 lines of Antiqua has been used originating from the Swedish test set in Drobac and Lindén (2020).

As Table 2 shows the test results trained with a single model (M2) reveals a significant improvement in CAR. An average 96.06% CAR was achieved corresponding to 3.94% CER. This can be compared to our baseline with an average CAR of 82.37% in Dannélls et al. (2020) for the specific time period.

The test results of M1 shows slightly lower CAR of 95.55% corresponding to 4.45% CER compared to using Calamari's default network.

The best CAR was achieved when voting with 5 models (M4 in Table 2) originating from 5-fold training using the neural network in Equation 1. We achieved CAR of 97.43% corresponding to CER of 2.57% after evaluation on the test set.

Training a single model (M3 in Table 2) with the same neural network and then evaluating on the test set resulted in a slightly lower CAR of 96.44%

Table 3: CAR for each newspaper edition spanning between 1818–1848. The second column show previously obtained results from the evaluation of the two OCR–engine system in Dannélls et al. (2020), which constitutes the baseline for this project. Third column show results obtained after cross-fold training and voting using the neural network architecture in Equation 1. The forth column show the different in percentage units between the baseline and our results.

	BASELINE	BEST	
NEWSPAPER EDITION	**CAR (%)**	**CAR (%)**	**+/- (%)**
STOCKHOLMSPOSTEN 1818-09-23	61.08	96.87	+35.79
GÖTHEBORGS ALLEHANDA 1819-05-26	56.65	94.79	+38.14
CARLSCRONAS WEKOBLAD 1820-05-10	78.59	94.89	+16.30
GÖTHEBORGSKA NYHETER 1821-10-27	80.84	94.95	+14.11
GÖTHEBORGS TIDNINGAR 1822-12-10	70.28	96.46	+26.18
DAGLIGT ALLEHANDA 1823-02-20	83.56	98.35	+14.79
WEXJÖBLADET 1824-08-21	80.55	98.85	+18.30
POST- OCH INRIKES TID. 1825-06-16	81.03	96.18	+15.15
STOCKHOLMS DAGBLAD 1826-11-27	78.76	96.03	+17.27
AFTONBLADET 1827-07-02	68.09	93.20	+25.11
HELSINGBORGSPOSTEN 1828-01-29	84.80	98.65	+13.85
DAGLIGT ALLEHANDA 1829-04-28	68.69	98.44	+29.75
NORRKÖPINGS TIDNINGAR 1830-03-30	81.97	98.44	+16.47
POST- OCH INRIKES TID. 1831-12-16	77.21	97.05	+19.84
GÖTEBORGS HAND. OCH SJÖ. 1832-10-15	92.15	97.92	+5.77
GÖTHEBORGS ALLEHANDA 1833-08-30	84.21	98.97	+14.76
MALMÖ ALLEHANDA 1834-03-12	87.41	98.34	+10.93
WEXJÖBLADET 1835-09-18	87.88	98.34	+10.46
POST- OCH INRIKES TID. 1836-12-08	96.40	98.46	+2.06
GEFLEBORGS LÄNS TIDNING 1837-02-01	82.15	98.49	+16.34
FREJA 1838-05-18	95.48	98.08	+3.32
CARLSCRONAS WEKOBLAD 1839-09-25	74.54	97.74	+23.2
SVENSKA BIET 1840-12-16	97.08	97.24	+0.16
NAJADEN 1841-09-03	95.36	95.78	+0.58
NORRLANDSPOSTEN 1842-01-21	85.97	96.95	+10.96
GÖTEBORGS HAND. OCH SJÖ. 1843-06-03	91.10	98.02	+6.93
WERMLANDSTIDNINGEN 1844-06-05	85.08	98.04	+12.96
JÖNKÖPINGSBLADET 1845-05-03	86.0	97.74	+11.74
NERIKES ALLEHANDA 1846-03-18	85.29	96.76	+11.44
GÖTHEBORGSKA NYHETER 1847-07-24	86.00	98.65	+12.65
SNÄLLPOSTEN 1848-05-22	89.41	99.73	+10.32
Average	82.37%	97.43%	+15.06

Table 4: Confusion matrices over the 10 most common character errors made by the single model (left) and after voting using an ensemble of 5 models (right). GT means the true character, PRED means the predicted character, COUNT means number of occurrence, PERCENT is the percentage of the total number of errors. Note that GT: "␣␣" and PRED: "␣" means a space has been deleted.

Single model				**Conf. voted models**			
GT	PRED	COUNT	PERCENT	GT	PRED	COUNT	PERCENT
ä	a	124	4.06%	ä	a	118	5.43%
ö	o	95	3.11%	å	ä	58	2.67%
å	ä	67	2.19%	ö	o	52	2.39%
␣␣	␣	53	1.73%	”	”	46	2.12%
u	n	45	1.47%	å	a	34	1.56%
ä	å	39	1.28%	n	u	31	1.43%
”	”	38	1.27%	ſ	f	29	1.33%
f	ſ	37	1.21%	ä	å	22	1.01%
i	l	33	1.08%	r	t	22	1.01%
				f	ſ	22	1.01%

corresponding to a CER of 3.56%. Compared to using voting this is nearly a 1% drop in CAR corresponding to approximately 827 more errors in the output. In total the best CAR of 97.43% is an improvement of 15.06% in comparison to the baseline. The results for each individual newspaper edition can be found in Table 3. In total CAR has improved for each individual newspaper edition.

All results were achieved through voting by the 5 models trained using the optimized neural network in Equation 1. Evaluation of the results on the test dataset containing 50 lines of Fraktur showed 96.84% CAR corresponding to 3.16% CER, displayed at row (5) in Table 2. On the test set containing Antiqua a slightly lower CAR of 96.44% was achieved corresponding to 3.56% CER.

5.4 Error Analysis

Table 4 show confusion matrices over the 10 most common character errors made by the single model (left) and after voting using an ensemble of 5 models (right). All models were trained using the optimized neural network in Equation 1. The most common confusions are in both cases as expected similar characters, such as "ä" and "a", "å" and "ä", "ö" and "o" and the confusion between long–s "ſ" and "f". Most notable when using voting the number of confusions of "ö" and "o" dropped from 95 to 52 occurrences and the deletion of a space dropped from 53 to below 10 occurrences. In summary, all confusions were reduced using voting. However, the confusions between "ä" and "a", and "ö" and "o" still constitute the most frequent confusions.

6 Conclusion and Future Work

In this paper we have investigated how deep CNN–LSTM hybrid neural networks can be utilized in order to improve current OCR results for 19th century Swedish newspaper text. Mixed deep CNN–LSTM hybrid models have been successfully trained in Calamari for the task of character recognition of Swedish historical newspaper texts spanning 1818–1848. Initial testing with the Calamari default network revealed a significant improvement in accuracy compared to our baseline (avg. 82.37% CAR) resulting in a CAR of 96.06%. This test alone showed the advantage of training individual mixed models over pre–trained models from commercial systems such as ABBYY FineReader. Highest CAR of 97.43% was achieved through voting with 5 best models using the optimized network. Thus, the combination of cross-fold training and confidence based voting significantly improve accuracy rates compared to training a single model using the same neural network. Furthermore, our best results show an significant improvement over the baseline results for the specific time period.

A promising future direction is to incorporate active learning (Reul et al., 2018). Active learning is based on the principle of maximal disagreement. An ensemble of voters (or models) are first trained on a set of training lines with their corresponding GT. Then, the voters are given unseen text lines and make predictions. Those lines where voters disagree the most on is then added to the training data for subsequent training enabling a maximal learning effect.

Acknowledgments

This work has been funded by the Swedish Research Council as part of the project *Evaluation and refinement of an enhanced OCR-process for mass digitisation* (2019–2020; dnr IN18-0940:1). It is also supported by Språkbanken Text and Swe-Clarin, a Swedish consortium in Common Language Resources and Technology Infrastructure (CLARIN) Swedish CLARIN (dnr 821-2013-2003). The authors would like to thank the RANLP anonymous reviewers for their valuable comments.

References

Yvonne Adesam, Dana Dannélls, and Nina Tahmasebi. 2019. Exploring the quality of the digital historical newspaper archive kubhist. In *Proceedings of the 4th Conference of The Association Digital Humanities in the Nordic Countries)*, pages 9–17, Copenhagen. University of Copenhagen, Faculty of Humanities.

Md Zahangir Alom, Tarek M. Taha, Chris Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Mahmudul Hasan, Brian C. Van Essen, Abdul A. S. Awwal, and Vijayan K. Asari. 2019. A state-of-the-art survey on deep learning theory and architectures. *Electronics*, 8:292.

Thomas M Breuel. 2008. The OCRopus open source OCR system. In *Electronic Imaging*. International Society for Optics and Photonics.

Thomas M. Breuel. 2017. High Performance Text Recognition Using a Hybrid Convolutional-LSTM Implementation. *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, 01:11–16.

Thomas M Breuel, Adnan Ul-Hasan, Mayce Ali Al-Azawi, and Faisal Shafait. 2013. High-Performance OCR for Printed English and Fraktur Using LSTM Networks. *2013 12th International Conference on Document Analysis and Recognition*, pages 683–687.

Dana Dannélls, Torsten Johansson, and Lars Björk. 2019. Evaluation and refinement of an enhanced OCR process for mass digitisation. In *Proceedings of the Digital Humanities in the Nordic Countries 4th Conference*, pages 112–123, Copenhagen. University of Copenhagen, Faculty of Humanities.

Dana Dannélls and Simon Persson. 2020. Supervised OCR post-correction of historical Swedish texts: What role does the OCR system play? In *Proceedings of the Digital Humanities in the Nordic Countries 5th Conference, Riga, Latvia, October 21-23, 2020*, volume 2612 of *CEUR Workshop Proceedings*, pages 24–37. CEUR-WS.org.

Dana Dannélls, Lars Björk, Ove Dirdal, and Torsten Johansson. 2020. Evaluation of a Two-OCR engine Method: First Results on Digitized Swedish Newspapers Spanning over nearly 200 Years. In *CLARIN Annual Conference 2020, (Virtual Event), 5-7 October, 2020. Book of Abstracts*.

Senka Drobac, Pekka Kauppinen, and Krister Linden. 2019. Improving OCR of historical newspapers and journals published in Finland. In *Proceedings of the 3rd International Conference on Digital Access to Textual Cultural Heritage*, pages 97–102. Association for Computing Machinery.

Senka Drobac and Krister Lindén. 2020. Optical character recognition with neural networks and post-correction with finite state methods. *International Journal on Document Analysis and Recognition (IJDAR)*, pages 1 – 17.

Lenz Furrer and Martin Volk. 2011. Reducing OCR Errors in Gothic-Script Documents. In *Proceedings of Language Technologies for Digital Humanities and Cultural Heritage Workshop*, pages 97–103.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 369–376, New York, NY, USA. Association for Computing Machinery.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Rose Holley. 2009. How Good Can It Get? Analysing and Improving OCR Accuracy in Large Scale Historic Newspaper Digitisation Programs. *D-Lib Magazine*, 15(3/4).

Amrith Krishna, Bodhisattwa P. Majumder, Rajesh Bhat, and Pawan Goyal. 2018. Upcycle your OCR: Reusing OCRs for post-OCR text correction in Romanised Sanskrit. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 345–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90.

Yann Lecun and Yoshua Bengio. 1995. *Convolutional networks for images, speech, and time-series*. MIT Press.

Christian Reul, Uwe Springmann, C. Wick, and F. Puppe. 2018. Improving OCR Accuracy on Early Printed Books by Utilizing Cross Fold Training and Voting. *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, pages 423–428.

Christian Reul, C. Wick, Uwe Springmann, and F. Puppe. 2017. Transfer Learning for OCRopus Model Training on Early Printed Books. *ArXiv*, abs/1712.05586.

Ray Smith. 2007. An overview of the Tesseract OCR engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, pages 629–633. IEEE.

Alexander Geyken Susanne Haaf, Frank Wiegand. 2013. Measuring the correctness of double-keying: Error classification and quality control in a large corpus of TEI-annotated historical text. *Journal of the Text Encoding Initiative*, (4).

Christoph Wick, Christian Reul, and Frank Puppe. 2018. Comparison of OCR Accuracy on Early Printed Books using the Open Source Engines Calamari and OCRopus. *Journal of Language Technology and Computational Linguistics*, 33(1):79–96.

Christoph Wick, Christian Reul, and Frank Puppe. 2020. Calamari – A High-Performance Tensorflow-based Deep Learning Package for Optical Character Recognition. *Digital Humanities Quarterly*, 14(2).