Improving OCR of historical newspapers and journals published in Finland

Senka Drobac
University of Helsinki
senka.drobac@helsinki.fi

Pekka Kauppinen
University of Helsinki
pekka.kauppinen@helsinki.fi

Krister Lindén
University of Helsinki
krister.linden@helsinki.fi

ABSTRACT

This paper presents experiments on Optical character recognition (OCR) of historical newspapers and journals published in Finland. The corpus has two main languages: Finnish and Swedish and is written in both Blackletter and Antiqua fonts. Here we experiment with how much training data is enough to train high accuracy models, and try to train a joint model for both languages and all fonts. So far we have not been successful in getting one best model for all, but it is promising that with the mixed model we get the best results on the Finnish test set with 95 % CAR, which clearly surpasses previous results on this data set.

CCS CONCEPTS

• **Applied computing → Optical character recognition**.

1 INTRODUCTION

Optical character recognition (OCR) of Finnish historical newspapers and journals published in Finland between 1771 and 1920 still yields unsatisfactory results. The online collection digitized and published by the National Library of Finland digi.kansalliskirjasto.fi contains over 11 million pages in mostly Finnish and Swedish, of which approximately 5.11 million are freely available [7]. Good quality OCR is essential to make this collection useful for harvesting and research.

While optical character recognition of printed text has reached high accuracy rates for modern fonts, historical documents still pose a challenge for character recognition. Some of the reasons for this are fonts differing in different materials, missing orthographic standards (same words are spelled differently), and sometimes poor image quality.

In previous work, Drobac et al. [4] and Kettunen et al. [6] create OCR models only for Finnish data, although half of the collection is in Swedish (until 1890 the main publication language) [6]. Furthermore, both works focus on the recognition of the Blackletter fonts. However, our sampling tests show that about 50 % of Swedish and 25 % of Finnish texts are published in the Antiqua typeset. Therefore, we need an approach that recognizes both languages and both typesets.

In this work, we create a small set of Swedish data with approximately 6,000 randomly picked line images accompanied with manually transcribed text and add this data to the Finnish 9,300 randomly picked line image dataset by Drobac et al. [4] to train OCR models using Ocropus software. We trained 5 models: Finnish only, two Swedish only models (one with 3,000 lines and the other with 6,000 lines) and two mixed models which contain Finnish and Swedish data combined. Both mixed models have 9,300 Finnish lines, one mixed model has 3,000 Swedish lines and the other 6,000.

The results show that with already a small amount of Swedish data we get quite good character accuracy rate (CAR) on the Swedish test set (92.9 %). Interestingly, additional Swedish data also increases accuracy on the Finnish test set (95.0 % CAR), in comparison with previously reported 93.5 %, which can be explained mainly by better recognition of the Antiqua typeface. Another interesting finding is that there is no significant difference between Swedish models trained on 3,000 and 6,000 lines when tested on a Swedish test set. However, the mixed model including 3,000 Swedish lines performs slightly better than the one with 6,000 lines (0.6 % on average on all test sets). This shows the importance of picking a representative set of training data.

1.1 Related work

In [11], they apply different OCR methods to historical printings of Latin text and get the highest accuracies when using Ocropy. Some work on Blackletter fonts has been reported in [3] where models were trained on artificial training data and got high accuracies when tested on scanned books with Blackletter text.

In [9], alongside with the overview of different OCR methods, they present the architecture of Ocropy and explain different steps of a typical OCR process.

In [4], they use Ocropy to recognize historical newspapers and journals published in Finland and they report character accuracy rates between 93 % and 95.21 %. In [6], they create ground truth data from the same corpus, perform recognition with Tesseract[1] and report between 85.4 % and 87 % word accuracy rates.

In [10], they use Ocropy to recognize scanned images of books printed between 1487 and 1870 and report character accuracy rates above 90 %.

DATeCH2019, May 8–10, 2019, Brussels, Belgium

ACM ISBN 978-1-4503-7194-0/19/05…$15.00
https://doi.org/10.1145/3322905.3322914

[1] https://github.com/tesseract-ocr/

2 DATA AND RESOURCES

2.1 Data

In our experiments, we use two data sets, Finnish and Swedish. Both of them are extracted from a corpus of historical newspapers and magazines that has been digitized by the National Library of Finland. The data sets consist of image files of individual lines of printed text as well the contents of said lines as plain text.

The Finnish data set was taken from [4], originally referred to as the DIGI set. It consists of approximately 12,000 pairs of manually transcribed image lines and corresponding images that were randomly picked from the time period 1820 - 1939. Our analysis showed that about 75 % of the data set is written in Blackletter and 25 % in Antiqua typeface. The set contains only Finnish text.

We created the Swedish data set as follows: First we randomly picked 1,000 sub-directories from the publicly available newspaper and journal corpus with data from 1771 until 1874. One subdirectory corresponds to one publication (for example, one issue of a newspaper) having several pages. Earlier, the entire corpus had been segmented and recognized with ABBYY FineReader 11[2], so we used this information to harvest the data. We extracted all lines of previously recognized text from each sub-directory and used the HeLI [5] tool for language identification on each line. Since we know that the pages are dominantly written in either Finnish or Swedish, we counted for each page how many lines were recognized as written in either Finnish or Swedish. Based on that count we decided to keep only pages that had more Swedish than Finnish lines. Finally, we randomly picked line images from the dominantly Swedish pages, and manually transcribed and saved them as plain text files. We collected the data in 3 stages, therefore we repeated this process 3 times (two times we got a bit more than 2,000 lines and the third time around 3,000), each time making sure that we did not get duplicates.

While this harvesting method might not seem ideal because things could go wrong with language identification on lines of OCRed pages that consist of both Finnish and Swedish text, in practice it proved to work quite well. While manually transcribing the data, we removed lines whose images were of poor quality, as well as lines not written in Swedish, and together they amounted to 5 % of the total line count. In the end, we were left with a total of 6,995 line pairs of image lines with accompanying ground truth text.

2.2 Ocropy

Ocropy[3] (previously known as OCRopus [1], [2], [3]) is a leading open source software toolkit for training OCR models using one dimensional long short term memory neural networks. In addition to character recognition, Ocropy offers tools for pre-processing documents (line segmentation, binzarization/normalization), tools for creation and correction of ground truth and evaluation tools.

We also found the Ocrocis "A Tutorial"[4] useful as it contains Ocropy instructions. Ocrocis is a set of wrapper tools for Ocropy.

3 METHOD

In this section we describe preparation of the data, OCR setup and evaluation practices.

3.1 Preparing the data

The Swedish data set is divided into two training sets: *swe-3k* which has 3,351 training line pairs and *swe-6k* with 6,159 line pairs. For testing we use our development and test sets, each with 418 line pairs. The reason for this configuration lies in the fact that the Swedish data was harvested in 3 phases. In the first phase we got around 2,100 line pairs but training on such a small data set gave us poor results so we decided to get more data. In the second phase we got an additional 2,100 lines, and combined them with the first set to get a total of 4187 line pairs. Then we randomly divided training, development and test sets in the ratio 80 % : 10 % : 10 %. This left us with the *swe-3k* training set and two sets of 418 line pairs for the dev and the test sets. In the third phase, we just added more training data to the existing training set giving us *swe-6k*.

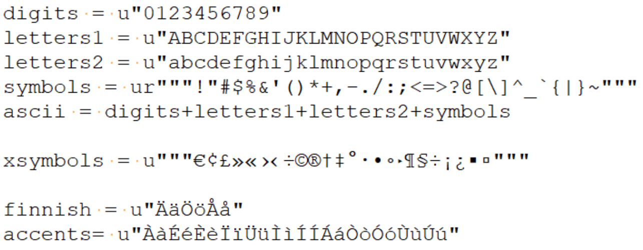
For the Finnish model, we used the original DIGI training set from [4], hereafter called *fin*. However, picking randomly, we reduced the development and test sets to 418 line pairs each, to get the same size sets as for Swedish and to have a faster testing process. To verify that there is no big difference between the original and reduced test sets, we tested several models on both the original and reduced test sets and got an average difference of a 0.3 % character accuracy rate in favour of the original test set.

We also wanted to see how the models behave on different font families, so we manually divided both test sets into Blackletter and Antiqua subsets. The Swedish test set contains 48.1 % of Blackletter and 51.9 % Antiqua lines, while the Finnish test set has 77.3 % of Blackletter and 22.7 % Antiqua lines. Since the data was randomly picked, we assume that this is a good representation of the corpus itself.

To find an optimal size of the Finnish training set, we transformed the *fin* training set into training sets with different sizes. The smallest one consists of 3,000 line pairs randomly picked from the entire training set. Each following data set had 1,000 training pairs more than the previous one. This gave us training sets with 3,000, 4,000, 5,000 . . . 9,000 training line pairs.

3.2 OCR

The training and prediction were done in a standardized way for Ocropy. First the line images were binarized with the `ocropus-nlib`



```
digits = u"0123456789"
letters1 = u"ABCDEFGHIJKLMNOPQRSTUVWXYZ"
letters2 = u"abcdefghijklmnopqrstuvwxyz"
symbols = ur"""!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~"""
ascii = digits+letters1+letters2+symbols

xsymbols = u"""€¢£»«›‹÷©®†‡°·•◦‣¶§÷¡¿▪▫"""

finnish = u"ÄäÖöÅå"
accents= u"ÀàÉéÈèÏïÜüÌìÍÍÁáÒòÓóÙùÚú"
```

Figure 1: Definitions of character sets, used while training Ocropy models

[2] https://www.abbyy.com/en-us/support/finereader-11
[3] https://github.com/tmbdev/ocropy
[4] http://cistern.cis.lmu.de/ocrocis/tutorial.pdf

Table 1: OCR character accuracy rates and word accuracy rates for models trained on different training sets and tested on Swedish and Finnish test sets. Columns show different OCR models. The model *fin* is trained on the entire Finnish training set, while *fin-7k* is trained on a 7,000 line subset. Models *swe-3k* and *swe-6k* are trained on the Swedish training set, the first one on approximately 3,000 Swedish training lines, and the second one on the entire training set. Models *fin + swe-3k* and *fin + swe-6k* are mixed models, trained on Finnish and Swedish training data together. Rows show CAR/WAR values for different test sets: swe-test is a full Swedish test set and *swe-blackletter* and *swe-antiqua* are subsets, divided by font family. Similarly, *fin-test* is the full Finnish test set and *fin-blackletter* and *fin-antiqua* are subsets.

model:	fin-7k	fin	swe-3k	swe-6k	fin + swe-3k	fin + swe-6k
swe-test	**85.7 / 55**	83.6 / 48	**92.9 / 75**	92.8 / 75	**90.6 / 69**	90.1 / 67
swe-blackletter	**86.6 / 56**	84.0 / 47	92.8 / 74	**92.9 / 74**	**90.8 / 69**	90.3 / 66
swe-antiqua	**84.8 / 53**	83.2 / 48	**92.9 / 77**	92.8 / 76	**90.4 / 69**	90.0 / 67
fin-test	**94.5 / 78**	93.9 / 75	91.5 / 64	**92.3 / 67**	**95.0 / 79**	94.4 / 77
fin-blackletter	**96.2 / 82**	95.7 / 80	92.4 / 65	**93.4 / 69**	**96.3 / 82**	95.8 / 80
fin-antiqua	**89.0 / 62**	87.9 / 60	88.3 / 60	**88.4 / 61**	**90.7 / 67**	89.7 / 65

tool, then we trained models with `ocropus-rtrain`, which saves a model after every 1,000 iterations. We did in total 1 million iterations for each model. We tested all saved models on the development set and choose the model with the best dev result. In case of mixed models, we did tests on both Finnish and Swedish development sets independently and then chose the model with the best average result for the two dev sets. Finally, prediction was done with the `ocropus-rpred` function. We evaluated the best models on all test sets.

While `ocropus-nlib` and `ocropus-rpred` were used with default settings, in `ocropus-rtrain` we used custom character configurations. For us it proved better to use a predefined character set than the -C option, where the character set is automatically learned from training data. For Finnish models we used the combination

```
default = ascii+xsymbols+finnish
```

while for Swedish and mixed models we also added accented vowels

```
default = ascii+xsymbols+finnish+accents
```

where character set definitions are described in Figure 1.

We also tried adding accents while training Finnish models, but we got significantly lower results than the *fin* model shown in Table 1 (-4.93 % on the swe-test and -2.18 % on the fin-test). Similarly, we got worse results with Swedish models trained without the accents. Anyhow, more experimenting with the character setup is needed to better understand how it affects the trained model.

3.3 Evaluation

To evaluate results on both development and test sets, we measured the performance of the system by using character accuracy rate (CAR), which is essentially the percentage of correct characters in the system output and is a common metric in OCR-related tasks. It is the number of correct characters divided by the sum of correct characters and errors in the system output. We also calculated word accuracy rate (WAR), which similarly to CAR is the percentage of correct words in the system output. We calculate it as the number of correct words divided by the sum of correct characters and errors in the system output.

$$CAR, WAR = \frac{correct}{correct + errors} \quad (1)$$

To get the number of errors, we first aligned ground truth and OCR lines on the character level (for both CAR and WAR). Then we calculated the overall Levenshtein distance [8] between the system output and the ground truth including deletions and insertions.

Since both Finnish and Swedish test sets have around 16,000 characters, we used only one decimal place for CAR. Going further than that would not express any significant difference in our test sets as a 0.01 % CAR improvement means 1.6 characters. For similar reasons, we left out all decimal points in the WAR measure.

For languages with relatively long words such as Finnish, character accuracy rates and character error rates are arguably better indicators when comparing the overall quality of the text between languages than, for instance, the word error rate, since longer words are more likely to contain an error. The legibility of Finnish text may actually improve considerably with increasing CAR without any notable change in the WAR. Furthermore, CAR makes it easier to compare different OCR systems, because different alignment and evaluation calculations can make a big difference on the word level.

4 RESULTS

In this section we present the test results. We trained individual Finnish and Swedish models, as well as mixed models (Finnish and Swedish training data combined). All models were tested on our Swedish and Finnish test sets (*swe-test* and *fin-test*), as well as their Blackletter and Antiqua subsets. The results (CAR/WAR) are shown in Table 1.

The Finnish model *fin* was trained on the entire Finnish training set *fin* with 9,300 line pairs. We also tested a Finnish model trained on a 7,000 line pair subset *fin-7k* of the full training set. There are two Swedish models, one trained on the entire Swedish training set (*swe-6k*) and another on an approximately 3,000 line pair subset (*swe-3k*). Mixed models *fin + swe-3k* and *fin + swe-6k* are trained on both Finnish and Swedish data combined. For the first one, we combined *fin* and *swe-3k* training sets, while for the second one

Table 2: Character accuracy rates and word accuracy rates of Finnish models trained on various sized training sets and tested on the Finnish test set. Rows show CAR/WAR results for models trained on different sized training sets, starting from a model trained on 3,000 line pairs (fin-3k) until the model that contains all 9,300 training line pairs (fin). The first column represents results on the full Finnish test set, the second one results on the Blackletter subset and the third one on the Antiqua subset.

model	fin-test	fin-blackletter	fin-antiqua
fin-3k	94.0 / 76	95.8 / 80	87.9 / 60
fin-4k	94.1 / 77	95.9 / 81	87.8 / 62
fin-5k	93.2 / 72	95.1 / 77	86.7 / 55
fin-6k	**94.4 / 77**	**96.2 / 82**	**88.2 / 62**
fin-7k	**94.5 / 78**	**96.2 / 82**	**89.0 / 62**
fin-8k	**94.3 / 77**	**96.2 / 82**	**88.1 / 60**
fin-9k	94.0 / 76	95.8 / 80	88.1 / 61
fin	93.9 / 75	95.7 / 80	87.88 / 60

we combined all the training data from both languages (*fin* and *swe-6k*).

Tables 3 and 4 show the ten most common recognition mistakes on the Swedish test set by models *swe-3k* in Table 3 and model *fin+swe-6k* in Table 4.

Similarly, Tables 5 and 6 show the ten most common recognition mistakes on the Finnish test set by the model *fin* in Table 5 and the model *fin-7k* in Table 6.

The first 3 columns in the tables give results on the entire test set, the next 3 columns on the Blackletter subset and the final 3 columns on the Antiqua subset. On each test set, the first column is the frequency of the mistakes, the second column the recognition result and the third one the ground truth. Deletions are marked with "_" in the OCR column and insertions with "_" in the ground truth column.

In order to see how much training data is needed for the Finnish model to saturate, we trained additional Finnish models on various sizes of Finnish training sets. Table 2 shows CAR/WAR results of those models, together with the complete *fin* model, on the Finnish test set. The first model *fin-3k* was trained on 3,000 line pairs randomly picked from the entire training set. Then the next one (*fin-4k*) is trained on the fin-3k with an addition of 1,000 lines randomly picked from the rest of the training set. The same principle continues with an additional 1,000 training pairs added to the previous training set. Each time models are trained from scratch. In the last row are the results for the model with all training lines (*fin*). It turned out that *fin-7k* yielded the best result. For comparison, we included it in the test results in Table 1.

5 DISCUSSION

In this section we analyze and discuss the potential reasons behind the results in Table 1 and the previous literature.

Swedish models have similar results for both Blackletter and Antiqua on the Swedish test set while the Finnish test set indicates

Table 3: A confusion matrix for the Swedish test set and Blackletter and Antiqua subsets after recognition with the *swe-3k* model

All			Blackletter			Antiqua		
#	ocr	gt	#	ocr	gt	#	ocr	gt
28	å	ä	25	å	ä	13	_	
21	_		13	ä	å	9	_	l
19	ä	å	8	_	i	8	_	,
14	_	l	8	_		7	l	f
13	_	,	7	f	s	6	.	,
10	_	-	7	_	-	6	ä	å
10	,	.	7	i	t	5	,	.
10	_		6	t	k	4	a	u
9	_	i	6	_		4	s	e
8	i	t	5	s	f	4	_	

Table 4: A confusion matrix for the Swedish test set and Blackletter and Antiqua subsets after recognition with the *fin + swe6k* model

All			Blackletter			Antiqua		
#	ocr	gt	#	ocr	gt	#	ocr	gt
60	ä	å	45	ä	å	20	_	
33	_		16	s	f	15	ä	å
21	e	c	13	_		14	e	c
17	å	ä	11	å	ä	10	_	,
17	s	f	10	a	g	8	l	_
13	_	,	8	_		8	.	,
13	_	l	7	_	t	7	.	_
12	_	t	7	_	l	6	,	.
10	.	_	7	e	c	6	K	R
10	,	.	7	_	-	6	å	ä

Table 5: A confusion matrix for the Finnish test set and Blackletter and Antiqua subsets after recognition with the *fin* model

All			Blackletter			Antiqua		
#	ocr	gt	#	ocr	gt	#	ocr	gt
21	_		17	_		10	a	s
18	a	_	10	_	i	9	a	_
12	_	i	9	a	_	5	i	l
12	_	-	8	_	-	5	t	i
11	a	s	7	k	t	4	.	,
10	t	i	7	ä	a	4	_	
9	_		7	M	N	4	_	-
8	k	t	6	n	u	4	e	_
8	_	l	6	k	l	3	a	o
7	ä	a	6	_	l	3	i	_

Table 6: A confusion matrix for the Finnish test set and Blackletter and Antiqua subsets after recognition with the *fin-7k* model

All			Blackletter			Antiqua		
#	ocr	gt	#	ocr	gt	#	ocr	gt
24	_		19	_		9	.	_
13	_	i	10	_	i	6	a	ä
11	t	i	9	_		5	_	
11	_		9	t	i	4	.	,
10	.	_	6	_	t	4	_	ä
8	l	i	5	J	I	4	i	_
8	_	t	5	_	-	4	l	_
8	a	ä	5	_	Y	4	l	i
7	.	,	4	t	l	4	,	.
7	_	Y	4	ä	a	3	_	,

a need for improvement of the Finnish models on Finnish Antiqua to match its Blackletter results.

There seems to be no really significant difference in performance results between models *swe-3k* and *swe-6k* on Swedish test sets. This indicates that the Swedish model saturates already on 3,000 line pairs and that adding more randomly picked data does not improve the model's performance. This matches the test that the Finnish model saturates on the *fin-7k* training data.

It is also surprising to see how well Swedish models recognize Finnish data. Both Swedish models recognize Finnish Antiqua approximately as well as the Finnish models. The Swedish models also achieve 92.4 % and 93.47 % CAR on Finnish Blackletter. The reasons for this could be that Swedish models have more variation covering the Finnish variation quite well despite language differences. Conversely, some lack of variation could also be the reason why the Finnish models do poorly on the Swedish test set.

One type of concrete difference between languages is the size of the frequently used character set. The Swedish test set has 103 unique characters compared with 88 for Finnish. A larger frequently used character set in Swedish than in Finnish could be the reason why we have poorer overall results on Swedish and why Swedish training data enhances results on Finnish test data, i.e. by adding variety and training data for rare characters. It would be interesting to see if adding more of the rare characters (instead of just random data) for training would bring further improvements to the Swedish models.

Combining the *fin* and the *swe-3k* training sets gives the best results on the Finnish test set. It could be that a small addition of different data adds variety to the model as well as covers more of the rare cases. Results in the Table 2, indicate that already 6-8 thousand lines are enough for Finnish. Maybe if we identified similar data and removed some of it, an even smaller training data set would be enough for Finnish. It would also be interesting to train the combined model on the *fin-7k* data set with the *swe-3k* to see if this yields further improvements.

Confusion matrices show that there are some more frequent mistakes, but the majority of the mistakes comes from a large number of low-frequent confusions. Note that fixing the top ten mistakes together would add only 0.8 % to the CAR, i.e. there is no specific mistake that causes most of the errors. In the Swedish test set ä is often confused with å and the other way around, but it is only 47 (28 + 19) mistakes in 16,879 characters, i.e. 0.3 %. As the top 10 mistakes only add 0.8 %, and the best CAR is 92.9 % on the *swe-test*, 6.3 % of the mistakes are caused by rather infrequent confusions. To improve models, we need to focus our training on less frequent characters and make sure we have enough representation for every character in the training data. Adding to the challenge are two different font families and the font variation in them.

We calculate our results as in [4] although we use a reduced version of the data set. To compare the [4] results stating CAR 93.5 % and the [6] result stating WAR 85.4 %, we calculated the CAR/WAR with our tools getting 93.5/74 for [4] and 93.0/85 for [6], i.e. [6] report lower CAR and higher WAR than [4]. However, it is important to realize that the difference is that in cases where the OCR mistakenly recognizes 2 or more words instead of one word, [6] write only the first word in their table, discarding the rest, so it is not possible to compare the results exactly, but they seem to be in the same range. In addition, our results clearly surpass the previous results overall and on Finnish Blackletter in particular.

6 CONCLUSIONS

While our results clearly surpass the previous results overall on the data set with historical newspapers and journals published in Finland achieving 95 % CAR on the Finnish test data, we found that representativity and variation are important in the training data allowing us to get quite good results with small amounts of training data of only a few thousand lines, i.e. adding general training data does not necessarily guarantee improvement in test results despite the data being randomly selected from the corpus. In particular, we found that adding representation for minority fonts to the training data improves overall performance on the test data.

REFERENCES

[1] Thomas Breuel. 2009. Recent progress on the OCRopus OCR system. In *Proceedings of the International Workshop on Multilingual OCR*. ACM, 2.

[2] Thomas M Breuel. 2008. The OCRopus open source OCR system. In *Electronic Imaging 2008*. International Society for Optics and Photonics, 68150F–68150F.

[3] 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. In *2013 12th International Conference on Document Analysis and Recognition*. IEEE, 683–687.

[4] Senka Drobac, Pekka Kauppinen, and Krister Lindén. 2017. OCR and post-correction of historical Finnish texts. In *Proceedings of the 21st Nordic Conference on Computational Linguistics*. 70–76.

[5] Tommi Sakari Jauhiainen, Bo Krister Johan Linden, Heidi Annika Jauhiainen, et al. 2016. Heli, a word-based backoff method for language identification. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects VarDial3, Osaka, Japan, December 12 2016*.

[6] Kimmo Kettunen, Jukka Kervinen, and Mika Koistinen. 2018. Creating and using ground truth OCR sample data for Finnish historical newspapers and journals. (2018).

[7] Kimmo Kettunen and Mika Koistinen. 2018. Re-OCR in Action–Using Tesseract to Re-OCR Finnish Fraktur from 19 th and Early 20 th Century Newspapers and Journals. *VIENNA* (2018), 11.

[8] Vladimir I. Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady* 10 (1966), 707.

[9] Faisal Shafait. 2009. Document image analysis with OCRopus. In *Multitopic Conference, 2009. INMIC 2009. IEEE 13th International*. IEEE, 1–6.

[10] Uwe Springmann and Anke Lüdeling. 2016. OCR of historical printings with an application to building diachronic corpora: A case study using the RIDGES herbal corpus. *arXiv preprint arXiv:1608.02153* (2016).

[11] Uwe Springmann, Dietmar Najock, Hermann Morgenroth, Helmut Schmid, Annette Gotscharek, and Florian Fink. 2014. OCR of historical printings of Latin texts: problems, prospects, progress. In *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*. ACM, 71–75.