Proceedings of 1st International Workshop in Computational Linguistics for Uralic Languages (IWCLUL 2015); ‹http://dx.doi.org/10.7557/scs.2015.2›

Can Morphological Analyzers Improve the Quality of Optical Character Recognition?

Miikka Silfverberg
University of Helsinki
Dept. of Modern Languages

mpsilfve@iki.fi

Jack Rueter
University of Helsinki
Dept. of Modern Languages

jack.rueter@helsinki.fi

December 16, 2014

Abstract

Optical Character Recognition (OCR) can substantially improve the usability of digitized documents. Language modeling using word lists is known to improve OCR quality for English. For morphologically rich languages, however, even large word lists do not reach high coverage on unseen text. Morphological analyzers offer a more sophisticated approach, which is useful in many language processing applications. This paper investigates language modeling in the open-source OCR engine Tesseract using morphological analyzers. We present experiments on two Uralic languages Finnish and Erzya. According to our experiments, word lists may still be superior to morphological analyzers in OCR even for languages with rich morphology. Our error analysis indicates that morphological analyzers can cause a large amount of real word OCR errors.

1 Introduction

Digital media is an integral part of modern society. Thus digitization of printed matter is crucial for the viability of minority languages. It also serves the linguistic community by making printed media widely available. Simply scanning documents, however, is not enough because few applications can deal with images directly. *Optical Character Recognition* (OCR) can substantially improve the usability of digitized documents for example by allowing search engines to index them. In this paper, we investigate improving the quality of OCR for languages with rich morphology, that is languages with extensive inflection, derivation and compounding.

OCR engines can benefit from language modeling, which is a field encompassing a variety of techniques that aim at improving the function of language processing applications by capturing key properties of the target language. For example, translation software and speech recognizers benefit greatly from sophisticated statistical language models. In OCR, however, simple language models such as word lists are commonly used.

Word lists are adequate in applications designed for languages with limited morphology such as English. Nevertheless, morphologically rich languages, including the Uralic languages, require more elaborate approaches. For these languages, even extensive word lists are unlikely to reach high coverage on previously unseen text [1].

In contrast to word lists, *morphological analyzers* [2], which encode the derivational and inflectional morphology of a language, can achieve substantially higher coverage. Thus it is conceivable that language models utilizing morphological analyzers could improve the quality of OCR for morphologically rich languages.

In this paper, we present experiments on OCR for two Uralic languages with rich morphology, Finnish and Erzya. We performed the experiments using the open-source OCR engine Tesseract [3] and open-source morphological analyzers for both languages. As baselines, we use both OCR systems without language modeling and systems using word lists.

In light of our experiments, it seems that morphological analyzers do help in OCR of morphologically rich languages compared to a baseline without language modeling. We were, however, unable to get improvements over using word lists harvested from the Wikipedia databases for Erzya and Finnish. This result is somewhat surprising, as the morphological analyzers have higher coverage on the test material than the word lists do. Error analysis revealed that the high coverage of the morphological analyzers may in fact present a problem for the OCR process, as it leads to a substantial number of real word errors.

Although we did not get improvements over word lists, it is worth pointing out that for some under-resourced minority languages morphological analyzers created by linguists represent the best readily available lexical resources in machine readable format. The reason for this is that digital content on the Internet can be scarce and the orthography of the material may be non-standard. Therefore, using a morphological analyzer as part of an OCR engine can still be motivated.

The paper is structured as follows. We describe related work in Section 2. In Section 3, we describe the Tesseract OCR engine, morphological analyzers and their integration. Section 4 details the experimental setup. In Section 5, we present the results of the experiments and a brief error analysis for the experiment on Finnish.

We conclude the paper in Section 6.

2 Related Work

Although, morphological analyzers have been used in OCR *post-processing*, this is, to our knowledge, the first investigation of utilizing morphological analyzers as language models *during* the OCR process. There are, however, other approaches to language modeling for OCR of morphologically rich languages, which have been investigated.

Smith et. al [4] add a module, which expands the vocabulary by generating additional word forms from stem suffix pairs. In contrast to our approach, their method requires no additional linguistic resources, since the sets of stems and affixes are harvested from word lists. We believe, however, that this approach is unlikely to work well with languages that have extensive compounding such as Finnish. Data sparseness will be a grave problem.

There is a large body of literature on spelling correction for morphologically rich languages, for example [5] and [6], and similar approaches have been successfully applied to OCR post-processing, for example [7, 8, 9]. In our work, we wanted to improve the language model instead of using post-processing to correct errors, because post-processing cannot in principle give as good results as improved language modeling. The reason is that knowledge about the reliability of predictions of the individual characters has already been lost before the post-processing stage.[1]

Finally, character based statistical language models have been investigated, but the results of this approach are mixed [10]. It seems that statistical language models do improve performance when the baseline is low, but they may in fact degrade the performance of high accuracy OCR systems. Statistical language modeling, however, has given good results in the related field of handwritten text recognition [11], where the overall performance is much lower.

3 Methods

In this section, we describe the Tesseract OCR engine, the HFST finite-state library, HFST morphological analyzers, and the process of combining these utilities.

[1] If this knowledge were available at the post-processing stage, post-processing could probably be used to the same effect as language modeling.

3.1 Tesseract

The Tesseract[2] OCR engine [3] was originally developed at HP Labs between 1984 and 1995 for high quality OCR of English. In 2005 it was released as an open-source project and has since been applied to several languages and alphabets, for example Finnish. Tesseract was therefore a natural starting point for exploring improvements for OCR of the Uralic languages.

The recognition process of Tesseract can be seen as a pipeline consisting of four stages [3]: (1) identification of character boundaries, (2) grouping of characters into words and lines, (3) word level recognition, and (4) resolution of ambiguous word spacing.

Our work focuses on the third stage of the pipeline, namely word level recognition. Word level recognition encompasses two sub-tasks: character recognition using a character classifier and word recognition using a combination of an additional word level classifier and various language models. During word level recognition, the word level classifier and language models give competing suggestions based on the output of the character recognizer. The highest scoring suggestion becomes the OCR output.

The existing language models in Tesseract are word lists, which are compiled into *directed acyclic graphs* (DAG) for fast processing. Tesseract incorporates a number of different language models[3], for example: A short list of *frequent word forms*, a more extensive *dictionary*, *punctuation patterns* and a list of *word forms containing digits*.

Each language model and the adaptive classifier have associated weights which determine their relative importance. For example, the frequent word model has a greater weight than the dictionary model reflecting the higher prior for seeing frequent words.

When the character model returns a scored set of possible word forms, each of the language models and the word level classifier return the highest scoring word form known to the model. These suggestions are further re-scored using the model specific weights. Finally, the highest scoring suggestion is selected.

We modify this system by replacing the word lists with a morphological analyzer. The associated weight for the analyzer is the same as for the dictionary model in the original system.

3.2 Helsinki Finite-State Technology

Helsinki Finite-State Technology (HFST) [12] is an open-source C++ library and collection of tools for constructing finite-state transducers and morphological analyzers

[2] https://code.google.com/p/tesseract-ocr/

[3] https://code.google.com/p/tesseract-ocr/wiki/TrainingTesseract3

based on finite-state technology. Morphological analyzers compatible with the HFST library exist for several languages. We know of at least fifteen Uralic languages[4] with HFST morphological analyzers, for example Erzya and Finnish.

3.3 Morphological Analyzers as OCR Language Models

As mentioned above, Tesseract internally represents language models as directed acyclic graphs or DAGs. HFST morphological analyzers are finite-state transducers (FST), which are closely related to DAGs. The main difference is that finite-state transducers transform strings instead of simply accepting or discarding them. Additionally, finite-state transducers can be cyclic unlike DAGs. By modifying both Tesseract and the analyzers, we were able integrate morphological analyzers into Tesseract.

There exists a straight-forward conversion (*projection*) from FSTs to finite-state automata, which are identical to DAGs in other respects, but may be cyclic like FSTs.

It turned out, that Tesseract is not in principle incompatible with cyclic graphs. The existing implementation simply did not offer a way to produce cyclic graphs. Fortunately, it was not difficult to implement a sub-class for the Tesseract language model class, `Dawg`, which does support cyclic graphs. Additionally, we implemented a driver for HFST automata in optimized lookup format, which supports lookup speeds of up to 100 000 words/s [13].

HFST morphological analyzers can contain so called flag diacritics [14], which are used to compress the finite-state machine by introducing non-determinism in a controlled way. Tesseract employs a search algorithm for finding word suggestions that requires that the language model be deterministic. Hence, it cannot handle flag diacritics. Fortunately, HFST includes utilities which can be used to eliminate flag diacritics from a finite-state machine without changing its behavior.

All the necessary steps to transform an HFST morphological analyzer into a Tesseract language model have been incorporated into the HFST interface as the tool `hfst-fst2tesseract`.

4 Experiments

In this section we describe the Finnish and Erzya data sets used in the experiments, the evaluation procedure and the experiments.

[4] http://giellatekno.uit.no/all-lang.eng.html

4.1 Data

We evaluate the impact of morphological analyzers in OCR for two Uralic languages, Erzya and Finnish. The Erzya language has a relatively rich morphological system of regular inflection, most extensive in the verbs and nouns. The verbs attest to object and subject conjugation in 7 moods, whereas there are 9 declensions for 9-15 regular case forms in nouns, with additional conjugation possibilities in two tenses for 3-6 of those. Erzya is written using the Cyrillic alphabet. Finnish is similar to Erzya in that it has a extensive noun and verb inflection. Additionally, Finnish has a productive compounding mechanism, which gives rise to an extensive vocabulary. Unlike Erzya, Finnish is written using the Latin alphabet.

We perform experiments on excerpts from novels. For Finnish, we use pages 5 - 21 of the novel *Elokuu* (August) by F.E. Sillanpää [15] (3219 tokens, 24096 characters) and for Erzya, we use pages 3 - 21 from the translation of the, originally Russian, novel *Ава* (Mother) by Maksim Gorky [16] (4539 tokens, 58548 characters). In order to estimate the effect of different language models on scanned material of varying quality, the data were scanned in different resolutions: 100, 200 and 300 dpi.

Even without language modeling, Tesseract performs quite well on scanned images of quality 300 dpi. The result requires relatively little manual correction. In contrast, 100 dpi images usually result in quite poor performance. In fact, manual correction may take longer than simply writing the text from scratch.

4.2 Resources

For constructing Tesseract systems with word lists as language models, we used the XML dumps of the Erzya[5] and Finnish[6] Wikipedias. We used the utility `wp2text` [7] for extracting the text contents from the XML files.

We formed lists containing the N most frequent word forms for various N in the range 1000 up to 1 million for Finnish and 1000 up to 68 000 for Erzya (there were no more unique word forms in the Erzya Wikipedia).

In addition to Wikipedia text, we used freely available morphological analyzers for Finnish and Erzya. OMorFi [17] is a broad coverage Finnish morphological analyzer available online[8]. For Erzya, we used the Erzya analyzer distributed by the Giellatekno project [18].

[5] `ftp://wikipedia.c3sl.ufpr.br/wikipedia/myvwiki/20140927/myvwiki-20140927-pages-meta-current.xml.bz2`

[6] `ftp://wikipedia.c3sl.ufpr.br/wikipedia/fiwiki/20141018/fiwiki-20141018-pages-meta-current.xml.bz2`

[7] `https://github.com/yohasebe/wp2txt`

[8] `https://code.google.com/p/omorfi/`

The coverages of different linguistic resources on test data are shown in Table 1. For both languages, the coverage is best using the morphological analyzer. However, for Finnish, the coverage of the one million word list comes very close.

Erzya	Coverage
1K word list	28.0%
10K word list	49.5%
68K word list	58.6%
Morphological analyser	80.6%

Finnish	Coverage
1K word list	32.2%
10K word list	52.8%
100K word list	71.4%
1000K word list	84.5%
Morphological analyser	86.7%

Table 1: Coverages of linguistic resources on the text tokens of the Erzya and Finnish test material.

4.3 Experiments

We trained five different OCR systems for Finnish and four systems for Erzya:

- A system without a language model (the baseline).
- Systems using 1000 and 10 000 word vocabularies both for Finnish and Erzya, a 68 000 word system for Erzya and 100 000 and 1 million word systems for Finnish.
- A system using a morphological analyzer as language model.

For Finnish, we constructed the baseline system simply by deleting the vocabularies (`freq-dawg` and `word-dawg`) from the existing Tesseract OCR system for Finnish [9]. For Erzya, we trained our own baseline system.

In order to compile systems with vocabularies ranging from 1000 words to 1 million words, we extracted the most common N words from the Wikipedia, compiled them into a directed acyclic graph using the Tesseract utility `wordlist2dawg` and used the graphs as word models (`word-dawg`) in a system which otherwise was identical to the baseline system.

The morphological analyzers, were first processed using the HFST utility `hfst-fst2tesseract`. We then combined them with a system identical to the baseline systems.

[9] see: `https://code.google.com/p/tesseract-ocr/downloads/list`

4.4 Evaluation

It is tempting to view OCR as a special case of sequence labeling, since an OCR engine essentially labels the characters in a digitized text using alphabetical symbols. This suggest evaluation based on character error rate in relation to a gold standard text.

Unfortunately, simple metrics such as character error rate cannot be used, since OCR frequently changes the length of the underlying text, because spurious characters may be inserted and characters in the text may be deleted. Therefore, we evaluated by measuring the *edit distance* [19] of the OCR result and the gold standard.

In practice, we first aligned the texts on character level using the Unix utility `diff`. We then computed the number of edits required to transform the OCR result into the gold standard text. We call this figure the *edit count* (EC). For each experiment, we report both raw edit counts and the reduction in edit count (ER) compared to the baseline OCR system without language modeling.

The edit reduction, ER, from a baseline B to an improved edit count C is

$$\mathrm{ER} = \frac{B - C}{B}$$

If, the baseline B is in fact better than the count C, ER will be negative.

We divided the test material into pages, and performed paired one sided Wilcoxon tests to asses the statistical significance of our results with confidence level 95%. We compared all systems to the baseline model. We additionally compared the best word-list system to the system using a morphological analyzer.

5 Results

In this section we show the results for Finnish in Table 2 and for Erzya in Table 3.

For the Finnish novel, all systems utilizing some kind of language modeling fared better than the baseline system without any kind of vocabulary information. The morphological analyzer performed better than the other systems on the lowest image quality 100 dpi. Otherwise, it in fact performed worse than the other systems utilizing language modeling.

For resolutions 300 and 200 dpi, all language models gave statistically significant improvements over the baseline in the 95% confidence interval. The best word list system was better than the morphological analyzer. For 100 dpi, only the morphological analyzer performed significantly better than the baseline, but not significantly better than the best word list model.

The results for Erzya paralleled those of Finnish. The morphological analyzer improves over the word lists only for the lowest resolution 100 dpi. For resolution 200

	300 dpi	200 dpi	100 dpi
No language model	0.0% (794)	0.0% (1265)	0.0% (15504)
1000 words	32.1% (539)	36.8% (799)	2.1% (15172)
10 000 words	**35.3%** (514)	44.7% (699)	4.0% (14891)
100 000 words	31.5% (544)	44.0% (708)	3.2% (15014)
1 million words	33.5% (528)	**45.4%** (691)	2.4% (15131)
Morph. analyzer	25.3% (593)	30.0% (885)	**5.7%** (14621)

Table 2: ER (and edit counts) for the Finnish novel Elokuu using different systems and resolutions.

dpi, the morphological analyzer does not seem to have any effect. For 200 and 300 dpi, the morphological analyzer was significantly worse than the best word list model. For 100 dpi, the model with 1000 word vocabulary was significantly worse than the baseline, but the other results were not statistically significant.

	300 dpi	200 dpi	100 dpi
No language model	0.0% (3257)	0.0% (3224)	0.0% (15788)
1000 words	20.9% (2576)	11.7% (2846)	-10.7% (17473)
10 000 words	29.5% (2295)	**22.7%** (2492)	1.8% (15498)
68K words	**30.9%** (2249)	21.9% (2517)	0.5% (15702)
Morph. analyzer	8.0% (2996)	-0.1% (3230)	**2.8%** (15353)

Table 3: ER (and edit counts) for the Erzya novel Ава using different systems and resolutions.

5.1 Error Analysis

We examined the errors of the Finnish OCR system using a morphological analyzer and the best performing word list system for the highest image quality 300 dpi.

We classified errors into two types: real word errors and others. Real word errors are errors, where the resulting incorrect word is known by the language model, for example a genitive of 'his/her' “Hänen” was recognized erroneously as the genitive of 'pike' “Hauen”. Other errors simply encompass all other error types, common examples include insertion and deletion of punctuation and casing errors such as lower case “v” being recognized as an upper case “V” and vice versa.

A total of 18% of the errors produced by the morphological analyzer were real word errors. In contrast the word list only gave 2% real word errors.

6 Discussion and Conclusions

In light of our experiments, it seems that morphological analyzers may do more harm than good in OCR. For higher resolutions, 200 and 300 dpi, the morphological analyzers fared worse than even the smallest vocabulary of 1000 words. This happens both for Finnish and Erzya. We believe, the large amount of real word errors is to blame. However, for the lowest image quality 100 dpi, the morphological analyzers do improve performance. It is interesting to compare these results to statistical language modeling for OCR, which also improves results when performance is low, but can degrade it otherwise [10].

Interestingly, vocabulary size does not seem to be a very good predictor of performance. For Finnish, the 10 000 word vocabulary performs best on the 300 dpi material with. Similarly, the system with 10 000 word vocabulary performs best for Erzya material in 200 dpi resolution. Overall, the results for all but the smallest vocabularies lie very close to each other. All OCR results are better for Finnish, which probably reflects the quality of the baseline models.

It would seem that the effect of language modeling is already exhausted at 10 000 words.[10] Therefore, it is not horribly surprising that the morphological analyzer does not achieve better results than the systems using word lists. The fact that its performance is so low, however, was mildly surprising.

In order to limit the number of real word errors, we tried excluding all compounds that had not been attested in real text from the morphological analyzers. Unfortunately, this did not improve the results.

It might be worth while trying to include word frequency information into the language model. However, this remains future work, as it would require extensive changes to Tesseract.

Acknowledgments

We, the authors, want to thank Jussi-Pekka Hakkarainen, Timo Honkela and Krister Lindén for helpful ideas and Tommi Pirinen for creating OMorFi. Additionally, we want to thank the Kone Foundation for funding. Finally, all three anonymous reviewers made useful suggestions. We are in your debt!

[10] This may be a consequence of Tesseract’s approach to language modeling described in Section 3.1.

References

[1] M. Creutz and K. Lagus. Unsupervised Models for Morpheme Segmentation and Morphology Learning. *ACM Trans. Speech Lang. Process.*, 4(1):3:1–3:34, February 2007.

[2] K. Koskenniemi. *Two-level Morphology: A General Computational Model for Word-Form Recognition and Production.* PhD thesis, University of Helsinki, 1983.

[3] R. Smith. An Overview of the Tesseract OCR Engine. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR)*, pages 629–633, Curitiba, Brazil, 2007. The IEEE Computer Society Conference Publishing Services.

[4] R. Smith, D. Antonova, and D.-S. Lee. Adapting the Tesseract Open Source OCR Engine for Multilingual OCR. In *Proceedings of the International Workshop on Multilingual OCR*, (MOCR), pages 1:1–1:8, New York, NY, USA, 2009. ACM.

[5] K. Oflazer and C. Güzey. Spelling correction in agglutinative languages. In *Proceedings of the Fourth Conference on Applied Natural Language Processing*, (ANLC), pages 194–195, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics.

[6] T. A. Pirinen and S. Hardwick. Effect of Language and Error Models on Efficiency of Finite-State Spell-Checking and Correction. In *Workshop On Finite State Methods In Natural Language Processing (FSMNLP)*, Donostia, Spain, 2012. Association for Computational Linguistics.

[7] K. Takeuchi and Y. Matsumoto. Japanese OCR Error Correction Using Stochastic Morphological Analyzer and Probabilistic Word N-gram Model. *International Journal of Computer Processing of Oriental Languages*, (1), 2000.

[8] G. Prószéky, M. Naszódi, and B. Kis. Recognition Assistance – Treating Errors in Texts Acquired from Various Recognition Processes. In *Proceedings of The 17th International Conference on Computational Linguistics (COLING): Project Notes*, Taipei, Taiwan, 2002. Association for Computational Linguistics.

[9] W. Magdy and K. Darwish. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 408–414, Sydney, Australia, 2006. Association for Computational Linguistics.

[10] R. Smith. Limits on the Application of Frequency-based Language Models to OCR. In *The Eleventh International Conference on Document Analysis and Recognition (ICDAR)*, pages 538–542, Beijing, China, 2011. The IEEE Computer Society Conference Publishing Services.

[11] U.-V. Marti and H. Bunke. Using a Statistical Language Model to Improve the Performance of an HMM-based Cursive Handwriting Recognition Systems. In *Hidden Markov Models*, pages 65–90. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2002.

[12] K. Lindén, E. Axelson, S. Drobac, S. Hardwick, M. Silfverberg, and T. A. Pirinen. Using HFST for Creating Computational Linguistic Applications. In A. Przepiórkowski, M. Piasecki, K. Jassem, and P. Fuglewicz, editors, *Computational Linguistics*, volume 458 of *Studies in Computational Intelligence*, pages 3–25. Springer Berlin Heidelberg, 2013.

[13] M. Silfverberg and K. Lindén. HFST Runtime Format—A Compacted Transducer Format Allowing for Fast Lookup. In *Workshop On Finite State Methods In Natural Language Processing (FSMNLP)*, Pretoria, South Africa, 2009.

[14] K. R. Beesley and L. Karttunen. *Finite State Morphology*. CSLI publications, 2003.

[15] F. E. Sillanpää. *Elokuu*. Otavan kirjapaino Oy, 2008 (originally 1941).

[16] Горький М. *Ава. Роман / Перевод А. Мартынов, М. Лукьянова.* Саранск: Мордовской государственной издательствась, 1952.

[17] T. A. Pirinen. Modularisation of Finnish Finite-State Language Description – Towards Wide Collaboration in Open Source Development of Morphological. In *Proceedings of the Eighteenth Nordic Conference of Computational Linguistics (NODALIDA)*, pages 299–302, Riga, Latvia, 2011.

[18] S. Moshagen, J. Rueter, T. Pirinen, T. Trosterud, and F. M. Tyers. Open-Source Infrastructures for Collaborative Work on Under-Resourced Languages. In *Proceedings of The Ninth International Conference on Language Resources and Evaluation, LREC*, pages 71–77, Reykjavik, Iceland, 2014.

[19] V. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics-Doklady*, 10(8):707–710, 1966.