# OCR Error Detection on Historical Text Using Uni-Feature and Multi-Feature Based Machine Learning Models

**Dana Dannélls**
Språkbanken Text
University of Gothenburg
dana.dannells@svenska.gu.se

**Shafqat Mumtaz Virk**
Språkbanken Text
University of Gothenburg
shafqat.virk@svenska.gu.se

## Abstract

Detecting errors that are caused by Optical Character Recognition (OCR) systems is a challenging task that has received much attention over the years. Recent work has explored machine learning methods using hand-crafted feature engineering, which, in addition to the difficulty in identifying the best feature combinations, is often very time and resources expensive. This raises the question: Do we always need many features to achieve better results? This is an open-ended question and its answer might depend on the task at hand. For OCR error detection, we experimented and found that interestingly a uni-feature based system conquered multi-feature based systems on a Swedish data set achieving state-of-the art results, and performed equally well on an English dataset. We also experimented to find which machine learning algorithm is more suitable for the task at hand by comparing the performance of five well-known machine learning algorithms, namely Logistic regression, Decision Trees, Bernoulli Naive Bayes, Naive Bays, and Support Vector Machines.

## 1 Introduction

Post processing is a conventional approach for correcting errors that are caused by Optical Character Recognition (OCR) systems. Traditionally, the task is divided into two subtasks: (1) Error detection, classify words as either erroneous or valid, and (2) Error correction, find suitable candidates to correct the erroneous words (Kolak and Resnik, 2005; Kissos and Dershowitz, 2016; Mei et al., 2016). Previous research has shown the worth of machine learning based approaches for both subtasks (Schulz and Kuhn, 2017; Nguyen et al., 2018, 2019; Dannélls and Persson, 2020). In the current work we aim to improve on the first task by using machine learning techniques.

Training an accurate machine learning model requires hand-crafted feature engineering, which involves finding best feature combinations and parameter settings. This also applies to OCR error correction, in particular because of the diversity of OCR errors, finding a suitable set of features is challenging (Amrhein and Clematide, 2018). At the same time, feature computation is often time and labour expensive. This raises the question: Do we always need a rich feature set for achieving better results or, depending on the task at hand, fewer features could lead to better or equally good results? Another related question is which machine learning algorithm is better for post OCR error detection?

In this paper we report on our experiments and present the results on two datasets, one for English and one for Swedish historical texts, to answer the above raised questions.

## 2 Related work

There are two approaches to OCR detection and correction. One approach is developing fine-tune methods for improving the OCR system, for example Tesseract has built-in post-correction functions for exploiting language specific data to help OCR. Another approach, that is taken here and has been adapted by the majority of previous authors, is to develop a method on top of the output of a specific system. The obvious advantage of the latter approach is that the developed method is not tailored to a particular system and could be applied on any OCR output regardless of the OCR system.

The majority of methods of OCR post correction apply supervised (Evershed and Fitch, 2014; Drobac et al., 2017; Khirbat, 2017) or unsupervised (Hammarström et al., 2017) machine learning techniques, depending on whether the ground truth is available.

In a more recent work, Mei et al. (2016) experimented with 6 features containing n-gram and context information, and reported a recall for bounded (true punctuation) detection of 73.5% using a support vector machine (SVM). Khirbat (2017) reported 69.6% precision, 44.2% recall and 54.1% F-score after training an SVM with 3 features: presence of non alpha-numeric characters, bi-gram frequency of the word and context information, that is if the word appears with its context in other places.

Nguyen et al. (2019) experimented with 13 character and word features on two datasets of English historical handwritten documents (monograph and periodical). The features they experimented with include character and word n-gram frequencies, part-of-speech, and the frequency of the OCR token in its candidate generation sets which they generated using edit-distance and regression model. They trained a Gradient Tree Boosting classifier and achieved a recall of 61% and 76% and an F-score of 70% and 79% on each dataset respectively. Their results are the highest reported on English historical material.

Dannélls and Persson (2020) trained an SVM model and experimented with 6 statistical and word based features, including the number of non-alphanumeric characters, number of vowels, word length, tri-gram character frequencies, number of uppercase characters and the amount of numbers occurring in the word. They reported 67% recall, and 63% F-score, which is the best results achieved on Swedish historical material.

In addition to the already tested features, some authors discussed how to improve the OCR accuracy with word2vec based features (Egense, 2017; Hämäläinen and Hengchen, 2019).

## 3 Error Detection Using Machine Learning Classification

Machine learning classifiers are known to have their pros and cons depending on the task. To our knowledge, there are no previous studies to examine the performance of different machine learning techniques for detecting OCR errors. We compared between 5 popular state-of-the-art machine learning classifiers to learn which of them is most suitable for this task. More specifically, we explored Logistic Regression, Decision Tree, Bernoulli Naive Bayes, Naive Bayes and SVM. All classifiers we experimented with are part of the Sci-kit Python library (Pedregosa et al., 2011).

## 4 Data

We have experimented with two datasets, Swedish and English.

The Swedish dataset comprises a selection of digitized versions of older Fraktur prints from 1626-1816,[1] and all pages from Olof v. Dalin's Swänska Argus from 1732-1734,[2] consisting of a total of 261,323 tokens. Ground truth data was produced by a transcription company who specializes manual transcriptions of historical material. In addition, the material was processed with three OCR systems: Abbyy Finereader 12, Tesseract 4.0 and Ocropus 1.3.3. Each one of these systems are using their own build-in dictionary and the quality of the OCR results differs significantly between the systems. When we compiled the training and testing sets in our experiments, described in Section 5, we included instances from all three systems to avoid the risk of developing a method that is biased towards a particular OCR system (Dannélls and Persson, 2020).[3]

The English dataset comprises newspaper text from the Sydney Morning Herald 1842-1954, consisting of 10,498,979 tokens and a ground truth data of randomly sampled paragraphs (Evershed and Fitch, 2014). The material was processed with Abbyy Finereader 14. The training and testing sets complied from this material contain instances from this particular OCR system only.

## 5 Experiments and results

We devised two experimental settings, Experiment I and Experiment II, to answer the two previously raised questions concerning what is the best feature (or feature combination) and machine learning method for OCR error detection.

### 5.1 Experiment Setup

**Experiment I** We experimented in three settings. First, we form our baseline by training an SVM model on the 6 features reported by Dannélls and Persson (2020). This set of features includes: (1) whether the word contains an alphanumeric character, (2) tri-gram word frequency, (3) whether the word contains a vowel, (4) whether the word

[1] https://spraakbanken.gu.se/en/resources/svensk-fraktur-1626-1816

[2] https://spraakbanken.gu.se/en/resources/dalin-then-swaanska-argus-1732-1734

[3] Datasets are available under CC-BY license and can be accessed from https://spraakbanken.gu.se/en/resources#refdata.

| | Swedish dataset | | | English dataset | | |
|---|---|---|---|---|---|---|
| | Precision | Recall | F-score | Precision | Recall | F-score |
| Baseline | 0.82 | 0.68 | 0.70 | 0.73 | 0.59 | 0.60 |
| Multi-feature | 0.80 | 0.71 | 0.73 | 0.81 | 0.62 | 0.63 |
| Uni-feature | 0.78 | 0.83 | 0.79 | 0.80 | 0.62 | 0.63 |

Table 1: Evaluation results of error detection for English and Swedish datasets with SVM.

length is over 13 characters, (5) whether the first letter appears in upper case, (6) whether the word contains a number. Second, analogous to previous approaches (Mei et al., 2016; Khirbat, 2017; Nguyen et al., 2019), we enhanced the feature set with 4 additional features: (1) the actual word (2) the actual word length, (3) the word proceeding and following the actual word, (4) whether the word exists in the pre-trained word2vec model provided by Hengchen et al. (2019), here we simply apply a look-up method against the word2vec space. In a more advanced setting this feature could be computed by accessing the actual vector in the model and extracting relevant information from it (Egense, 2017)

Third, we removed all features and trained the SVM model only on one feature, the actual word.

In this experiment, we chose randomly selected subsets of 50K tokens from the Swedish and the English datasets. 10K tokens were used as a development set, and the remaining 40K tokens were divided into training (80%) and testing (20%) datasets. Unfortunately, due to the time constraints and memory issues we were not able to apply 5-cross validation which is the reason why the remaining 40K tokens were divided into two subsets.

**Experiment II** In this experimental setting, we trained 5 machine leaning models on one feature that is the actual word. Here we use the same datasets as in experiment I. Unlike experiment I where the remaining 40K tokens were divided into training and testing, 5-cross validation was applied.

### 5.2 Results

The results from experiment I are presented in Table 1. Even though we experimented with the same feature combination as reported in Dannélls and Persson (2020), our baseline is 70% F-score compared to their reported 63% F-score probably owing to parameter settings and the chosen sub datasets. The results on the Swedish dataset show that the model trained on one feature outperforms the model trained on the 6 and 10 feature sets respectively. With an F-score of 79% it is the best performing model on Swedish historical text reported so far.

Our baseline results on the English dataset are not as high compared to the F-score reported by Mei et al. (2016) and Nguyen et al. (2019). The reason for this is because we are experimenting with completely different datasets with respect to both size and content. The results do show an improvement over the baseline with the multi-feature set which also confirms previous results. Interestingly, the results on the English dataset show no difference in performance between the multi-feature and the uni-feature sets. We believe the difference in the results between Swedish and English can be characterized to the nature of the data. A manual inspection of the dataset reveals that the Swedish dataset is representative with regards to its vocabulary. Hence, more words in the Swedish test set were seen in the training set as compared to the English counterpart.

The results from experiment II, where only one feature was used to train different machine learning models are presented in Table 2. We can observe that both Decision Tree and SVM outperform the other models on the Swedish dataset, achieving 80% F-score. Bernoulli Naive Bayes is almost as good with an F-score of 79%. Decision Tree is the best performing model on the English dataset with the highest F-score of 71%.

## 6 Discussion and Conclusion

As mentioned previously, in the first experiment we trained an SVM model on a baseline feature set, a multi-feature set, and a uni-feature containing the word itself. By training the model on the word itself, we are necessarily turning the machine learning model into a dictionary look-up kind of system. In an ideal case, one could develop an actual dictionary based system and compare its performance to the proposed uni-feature based system to further investigate the machine learning impact. We leave it as a future work. The results show that the one

| | Swedish dataset | | | English dataset | | |
|---|---|---|---|---|---|---|
| | Precision | Recall | F-score | Precision | Recall | F-score |
| Logistic Regression | 0.82 | 0.74 | 0.76 | 0.74 | 0.60 | 0.65 |
| Decision Tree | 0.84 | 0.79 | **0.80** | 0.71 | 0.73 | **0.71** |
| Bernoulli Naive Bayes | 0.84 | 0.78 | 0.79 | 0.79 | 0.58 | 0.63 |
| Naive Bayes | 0.67 | 0.54 | 0.37 | 0.70 | 0.60 | 0.59 |
| SVM | 0.84 | 0.79 | **0.80** | 0.74 | 0.60 | 0.66 |

Table 2: Evaluation results of error detection for English and Swedish datasets trained with different models on one feature. The best performing models are highlighted in bold.

feature model is sufficient, not only for improving over the baseline, but also for reaching better results than previously reported on Swedish text. The results on the English data show that a uni-feature model is as good as a multi-features model. This means that with the dictionary of words over the training data alone we can better predict whether a word contains an OCR error or not.

Training supervised machine learning models on large amount of features is a computationally expensive task. This has been demonstrated in previous work where hand-crafted features were considered at the expense of high computational costs. What makes the proposed approach interesting is that it eliminates the need to compute many features for detecting OCR errors. On the other hand, we are aware that it relies on the availability of large amount of training data which is also costly, and training will take more time because the model is computing categorical features with many possible combinations. Many other different feature sets and various combinations could be explored in this experiment to examine the effect of the method in more depth. For example, the context feature was computed by only looking at the proceeding and following token. Another possible approach is to look at token appearing in a larger context window. The word2vec feature was computed against the word2vec model, treating it as a bag-of-words model. A better approach could be to perform some computation against the vectors. Unfortunately, due to the time constraints we were unable to continue and experiment with those. Instead, these are left for future work.

The aim of the second experiment was to answer the question: Which machine learning algorithms is the most appropriate for detecting OCR errors. The results show SVM is as good as Decision Tree on the Swedish dataset, on the English dataset Decision Tree outperforms the other models. The latter finding supports earlier work on English (Abuhaiba, 2006) and sheds light on the strengths of machine learning algorithms, other than SVM, for the task at hand.

Notwithstanding, in this work we kept the datasets rather small mostly because of the time constraints and memory issues. Unfortunately, we were unable to experiment with the complete datasets, which leaves several open questions regarding the representativeness of the chosen data. In the future, we plan to experiment with bigger datasets, and our hope is that we will improve on the results reported in this study. We also think, context features combined with the candidate word (the uni-feature used in the current study) will make a better feature combination, and might improve the results, but we leave it to be further explored.

## Acknowledgments

The work presented here was funded by (1) the *Dictionary/Grammar Reading Machine: Computational Tools for Accessing the World's Linguistic Heritage* (DReaM) Project awarded 2018–2020 by the Joint Programming Initiative in Cultural Heritage and Global Change, Digital Heritage and Riksantikvarieämbetet, Sweden; (2) the Swedish Research Council as a part of the project *South Asia as a linguistic area? Exploring big-data methods in areal and genetic linguistics* (2015–2019, contract no. 421-2014-969); (3) *From Dust to Dawn: Multilingual Grammar Extraction from Grammars* project funded by Stiftelsen Marcus och Amalia Wallenbergs Minnesfond 2007.0105, Uppsala University; (4) the Swedish Research Council as part of the project *Evaluation and refinement of an enhanced OCR-process for mass digitisation* (2019–2020, grant agreements IN18-0940:1 and 421-2014-969). It is also supported by Språkbanken Text and Swe-Clarin, a Swedish consortium in Common Language Resources and Technology In-

frastructure (CLARIN) Swedish CLARIN (grant agreement 821-2013-2003).

## References

Ibrahim S I Abuhaiba. 2006. Efficient OCR using Simple Features and Decision Trees with Backtracking. *Journal for science and engineering*, 31(2):223–244.

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.

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.

Senka Drobac, Pekka Kauppinen, and Krister Lindén. 2017. OCR and post-correction of historical Finnish texts. In *In Proceedings of the 21st Nordic Conference on Computational Linguistics NoDaLiDa*. Linköping University Electronic Press.

Thomas Egense. 2017. Automated Improvement of Search in Low Quality OCR Using Word2Vec. In *Proceedings of DHN 2017*, Digital Humanities in the Nordic Countries.

John Evershed and Kent Fitch. 2014. Correcting Noisy OCR: Context Beats Confusion. In *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*, DATeCH '14, pages 45–51, New York, NY, USA. Association for Computing Machinery.

Mika Hämäläinen and Simon Hengchen. 2019. From the paft to the fiiture: a fully automatic NMT and word embeddings method for OCR post-correction. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 431–436, Varna, Bulgaria. INCOMA Ltd.

Harald Hammarström, Shafqat Mumtaz Virk, and Markus Forsberg. 2017. Poor Man's OCR Post-Correction: Unsupervised Recognition of Variant Spelling Applied to a Multilingual Document Collection. In *Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage*, DATeCH 2017, pages 71–75, NY, USA.

Simon Hengchen, Ruben Ros, and Jani Marjanen. 2019. A data-driven approach to the changing vocabulary of the 'nation' in English, Dutch, Swedish and Finnish newspapers, 1750-1950. In *Proceedings of the Digital Humanities (DH) conference 2019, Utrecht, The Netherlands*.

Gitansh Khirbat. 2017. OCR post-processing text correction using simulated annealing (OPTeCA). In *Proceedings of the Australasian Language Technology Association Workshop 2017*, pages 119–123.

Ido Kissos and Nachum Dershowitz. 2016. OCR error correction using character correction and feature-based word classification. In *Document Analysis Systems(DAS) 12th IAPR Workshop*, pages 198–203. IEEE.

Okan Kolak and Philip Resnik. 2005. OCR post-processing for low density languages. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, Vancouver, B.C., Canada.

Jie Mei, Aminul Islam, Yajing Wu, Abidalrahman Mohd, and Evangelos E Milios. 2016. Statistical learning for OCR text correction. *arXiv preprint*, abs/1611.06950.

T. Nguyen, A. Jatowt, M. Coustaty, N. Nguyen, and A. Doucet. 2019. Post-ocr error detection by generating plausible candidates. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 876–881.

Thi-Tuyet-Hai Nguyen, Mickaël Coustaty, Doucet Antoine, and Nhu-Van Nguyen. 2018. Adaptive Edit-Distance and Regression Approach for Post-OCR Text Correction. In *20th International Conference on Asia-Pacific Digital Libraries, ICADL*.

Thi-Tuyet-Hai Nguyen, Adam Jatowt, Mickael Coustaty, Nhu-Van Nguyen, and Antoine Doucet. 2019. Deep Statistical Analysis of OCR Errors for Effective Post-OCR Processing. In *Proceedings of the 18th Joint Conference on Digital Libraries*, JCDL '19, page 29–38. IEEE Press.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research*, 12:2825–2830.

Sarah Schulz and Jonas Kuhn. 2017. Multi-modular domain-tailored OCR post-correction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2716–2726, Copenhagen, Denmark. Association for Computational Linguistics.