A System for Identifying and Exploring Text Repetition in Large Historical Document Corpora

Aleksi Vesanto,[1] **Asko Nivala,**[2,3] **Tapio Salakoski,**[1] **Hannu Salmi,**[2] **and Filip Ginter**[1]
[1]Turku NLP Group, Department of FT
[2]Cultural History
[3]Turku Institute for Advanced Studies
University of Turku, Finland
first.last@utu.fi

Abstract

We present a software for retrieving and exploring duplicated text passages in low quality OCR historical text corpora. The system combines NCBI BLAST, a software created for comparing and aligning biological sequences, with the Solr search and indexing engine, providing a web interface to easily query and browse the clusters of duplicated texts. We demonstrate the system on a corpus of scanned and OCR-recognized Finnish newspapers and journals from years 1771 to 1910.

1 Introduction

The task of finding repeated passages from old newspapers and magazines is relevant to the historians who study the spread of news in time and space. The underlying corpora – in our case scanned and OCR-transcribed newspapers and journals, some over 200 years old – pose a number of technical challenges. Firstly, the size of the corpora is large, in the millions of pages range. And, more importantly, the text produced using OCR is often of poor quality – sometimes nearly unreadable as shown in Figures 1 and 2. This makes the corpora inaccessible to commonly used fast methods such as Passim (Smith et al., 2014) which rely on identifying seed overlaps that are several full words in length, a rare occurrence in our data whose error rate has been estimated to 25-30% in terms of words, depending on period of print (Kettunen et al., 2016).

In this demo, we present a system for identifying text repetitions and forming their clusters using BLAST (Altschul et al., 1990), a software developed to compare and align biological sequences. To browse and search these clusters, we index them using Solr, an open-source search engine, and provide a web interface that is capable of searching and visualizing these repeated text clusters and their associated metadata.

We demonstrate the software and its web interface on a corpus of OCR scanned old Finnish newspapers and journals from years 1771 to 1910, around 3 million pages in total.

2 Software Architecture

2.1 Data Preprocessing and Indexing

NCBI BLAST is built for fuzzy-aligning protein and nucleotide sequences and querying massive sequence databases. As such, it seems an ideal tool for the task, but the assumption of working with biological data is ubiquitous throughout the BLAST codebase, and it cannot be easily used for matching arbitrary alphabets. Therefore, to apply BLAST in our setting, we need to first encode our whole corpus into protein sequences composed of an alphabet of 23 amino acids. As we are limited by the number of distinct amino acids, we can only map the 23 most common lowercase letters in our corpus to distinct amino acids. We then lowercase our corpus and replace all characters using this mapping. Characters that do not have an equivalent in our mapping are discarded – and naturally restored later. This encoding also simultaneously works as a preprocessing method, as the documents have a lot of noise in them in the form of arbitrary characters and spaces. These characters are not among the 23 most common letters, so they are discarded in the encoding process. Interestingly, although space is the most used character in the corpus, we found that discarding spaces nevertheless makes the BLAST query process more than twice as fast and the hits we find are also slightly longer. Once encoded into protein sequences, the documents are indexed using BLAST for a subsequent fast retrieval.

Proceedings of the 21st Nordic Conference of Computational Linguistics, pages 330–333,
Gothenburg, Sweden, 23-24 May 2017.

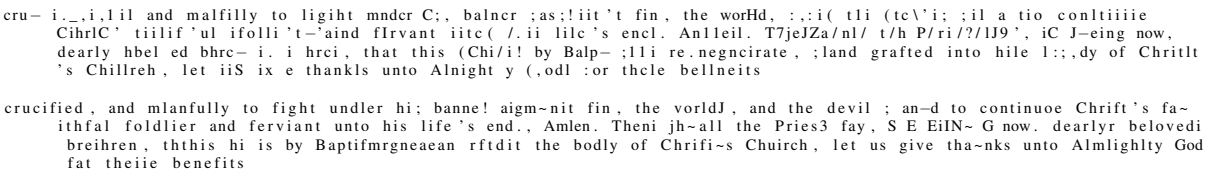

Figure 1: A hit pair from a run with ECCO dataset. (OCR-scanned books from 18th century)

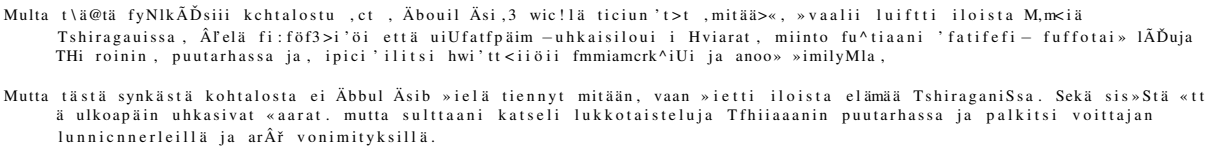

Figure 2: A hit pair from Finnish newspapers.

2.2 Clustering

Every document in the corpus, 3 million pages in our case, is subsequently matched by BLAST against the entire indexed collection – i.e. all pairwise comparisons are calculated. The matching document pairs contain the starting and ending offsets from each document, which we use to connect and combine pairs that share a text passage with a sufficient overlap. Because the matching is fuzzy and the texts are very noisy, if the same text passage is reused in a number of documents, each of the identified document pairs will mark a slightly different beginning and end of the passage. For instance, if a passage from a document A is reused in documents B and C, the offsets in A will be slightly different between the A-B and A-C pairs. To deal with the problem, we calculate consensus indexes that combine all passages in one document from individual document pairs that are close to each other – in our example the two passages in A from the A-B and A-C pairs. We do this by averaging the starting and ending indexes of passages that overlap by at least 80%, obtaining consensus passages.

After identifying all the distinct consensus passages for each document, we create a graph with the consensus passages in individual documents as nodes, and edges between corresponding passage pairs. Subsequently, we extract all connected components from the graph, providing us with an initial estimate of clusters of documents that share a passage. The identification of passage clusters through connected components in the graph can be seen as a high recall method. A stray edge – not uncommon in the noisy data – may connect two otherwise disjoint clusters together. To deal with this, we separate these clusters using community detection. To this end, we apply the Louvain method (Blondel et al., 2008) which identifies communities within the connected components of the graph and we subdivide those connected components that have several distinct, highly-connected communities (subcomponents). This removes the likely stray edges that were connecting them. After this subdivision, we obtain the final clusters and the nodes within them are the repeated text passages we seek.

2.3 Finnish newspapers

We applied our system to old OCR-scanned Finnish newspapers and journals from years 1771 to 1910, around 3 million pages in total. We found nearly 8 million passage clusters containing around 49 million repeating passages. We only considered hits that are 300 characters or longer, as the shorter hits would either be too fractioned to be useful or they are just boilerplate text. The most computationally intensive part of the process is running the BLAST queries, which took 150,000 CPU-core hours. Clearly, a dataset of this size requires an access to a cluster computer, which is not surprising given the complexity involved in fuzzy-matching 3 million pages of text against each other. This computationally intensive step however only needs to be performed once and its results can be reused at a later point.

3 Web User Interface

For the user interface, we index our data with Solr. More specifically, we index the data as nested doc-

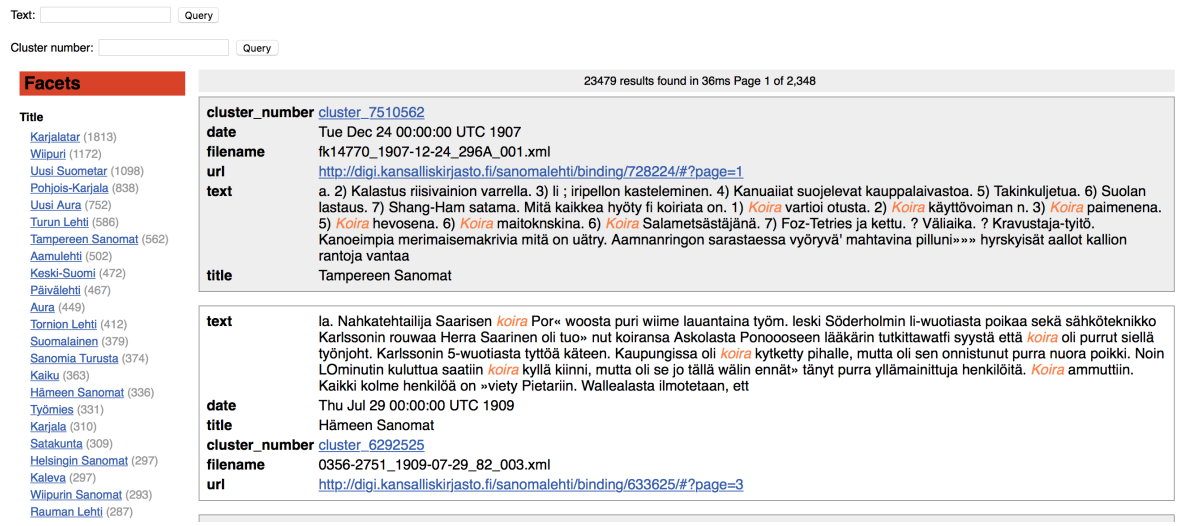

Figure 3: A screenshot showing the user interface.

uments, where the parent document is the cluster and child documents are the hits within that cluster. Solr is capable of querying the data very efficiently, easily allowing for a swift, real-time search. Solr has built-in support for Apache Velocity Template Engine and out of the box it provides a simple browse page where one can browse the query results. Using this template engine, we implement an easy-to-use interface suitable to the nature of the data.

A screenshot of a result page is shown in Figure 3. At the top, a search field allows a free text search. Below is a field for direct search by cluster number. This will result in all hits that belong to that cluster as well as other information about the cluster, such as average length of hits, number of hits and the year of its first occurrence. On the right, we see a small snippet of the results. For every matching hit we can see the name of the original file, date when that issue was published, the name of the newspaper or journal, URL for viewing the original scanned document online, cluster number and the text itself with the query hits highlighted. Clicking the cluster number link shows all hits, i.e. occurrences of the same repeated text passage, within the cluster. Finally on the left we have a facet view, currently giving an overview of hits from a specific magazine. The rich query language employed by Solr gives us the capability of performing fuzzy and proximity search, which is especially useful in our case of low-quality OCR-recognized documents.

As one would expect from a mature search engine like Solr, querying this large collection of repeated text clusters is effortless and real-time. For instance, querying for *kissa*, the Finnish word for *a cat*, found over 23,000 results, returning the first page of 10 results in 38ms.

4 Conclusions

The ability to identify text repetition in large historical corpora is of great importance to historians, and the problem of fuzzy match in large text collections is of a broader interest in corpus research and NLP in general. We have presented a fully implemented and currently deployed software and web interface to identify repeated text passages in large corpora of poor OCR quality, and present them through a simple web interface. We have shown that the BLAST algorithm works efficiently in identifying regions of similarity in historical text corpora, even in cases where the quality of OCR is extremely low, for instance where the original text has been printed with Gothic typeset (*Fraktur*), or with poor paper and ink quality. The development of new tools for text reuse detection is essential for further enhancement of the use of scanned historical documents, and work with noisy corpora in general.

Acknowledgments

The work was supported by the research consortium *Computational History and the Transformation of Public Discourse in Finland, 1640-1910*, funded by the Academy of Finland. Computational resources were provided by CSC — IT Centre for Science, Espoo, Finland.

References

Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman. 1990. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, Oct.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.

Kimmo Kettunen, Tuula Pääkkönen, and Mika Koistinen. 2016. Between diachrony and synchrony: Evaluation of lexical quality of a digitized historical finnish newspaper and journal collection with morphological analyzers. In *Baltic HLT*.

David A. Smith, Ryan Cordell, Elizabeth Maddock Dillon, Nick Stramp, and John Wilkerson. 2014. Detecting and modeling local text reuse. In *Proceedings of the 14th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '14, pages 183–192, Piscataway, NJ, USA. IEEE Press.