TU WIEN Informatics

Cross-lingual Search in Pre-processed Archival Facsimile Documents

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering und Internet Computing

eingereicht von

David Banyasz, BSc
Matrikelnummer 01426657

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof. Dr. Allan Hanbury
Mitwirkung: Dipl.-Ing. Dr.techn. Sebastian Hofstätter

Wien, 9. Juli 2023

David Banyasz	Allan Hanbury

TU WIEN Informatics

Cross-lingual Search in Pre-processed Archival Facsimile Documents

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering and Internet Computing

by

David Banyasz, BSc
Registration Number 01426657

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof. Dr. Allan Hanbury
Assistance: Dipl.-Ing. Dr.techn. Sebastian Hofstätter

Vienna, 9th July, 2023

David Banyasz | Allan Hanbury

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

David Banyasz, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 9. Juli 2023

David Banyasz

Acknowledgements

I would like to express my gratitude to my advisors Sebastian Hofstätter and Prof. Allan Hanbury for their guidance, invaluable feedback and seemingly endless patience with me.

I would also like to thank my close friends and family for their support, and lastly, my girlfriend Jennifer, for encouraging me from start to finish and keeping me motivated on countless occasions along the way.

This thesis contributes to the research project "Visual History of the Holocaust: Rethinking Curation in the Digital Age" (`www.vhh-project.eu`). This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 822670.

Kurzfassung

Jüngste Fortschritte in der Textdigitalisierung und -verarbeitung haben eine Vielzahl von Möglichkeiten eröffnet, historische Archive effizient und automatisiert zu bearbeiten und zu digitalisieren. Verarbeitungsschritte, die auch Spracherkennung, optische Zeichenerkennung (optical character recognition - OCR), Named Entity Recognition (NER), Markierung von Erkennungsfehlern und automatische oder manuelle Korrekturen umfassen, können zu digitalisierten Archiven führen, die sowohl qualitativ hochwertige Faksimile-Darstellungen von gescannten Originaldokumenten als auch extrahierte Text-Metadaten nahe am Originaltext in einem maschinenfreundlichen Format liefern.

Im Rahmen des Forschungsprojekts "Visual History of the Holocaust" (VHH) ist die Erforschung digital aufbereiteter Archive ein wichtiger Schritt für den zukünftigen Arbeitsablauf von Archivaren und Historikern gleichermaßen. Nach einer Analyse und Kategorisierung der Anforderungen der Mitarbeiter des VHH-Projekts schlagen wir eine neuartige, semantisch erweiterte Suchabfrage-Methode und ein Konzept zur dynamischen Generierung von suchrelevanten Faksimile-Bildausschnitten vor. Diese Arbeit demonstriert einen auf diesen Methoden basierenden Human-in-the-Loop Such- und Recherchearbeitsablauf, indem sie einen Prototyp einer Suchbenutzeroberfläche bereitstellt, die auf die intuitive Erkundung von Themen in einem mehrsprachigen historischen Faksimile-Archivkorpus ausgerichtet ist.

Abstract

Recent advances in text digitization and processing have opened up plenty of possibilities for historical archives to be processed and digitized in an efficient and automated manner. Processing steps, also involving language detection, optical character recognition (OCR), named entity recognition (NER), recognition error detection, and automated or manual correction can result in digitized archives providing both high-quality facsimile representations of original document scans and extracted text metadata close to the original text in a machine-friendly format.

In the context of the research project "Visual History of the Holocaust" (VHH), exploration of digitally enhanced archives is an important step forward in the future workflow of archivists and historians alike. After analysing and categorizing the requirements of collaborators in the VHH project, we propose a novel semantically extended retrieval method and a concept for dynamically generating retrieval-relevant facsimile image snippets. This work demonstrates a Human-in-the-Loop retrieval and research workflow based on these methods by providing a search user interface prototype geared towards intuitively exploring topics across a multilingual historical facsimile archive corpus.

Contents

CHAPTER 1

Introduction

Historical archives contain a wealth of information that is only starting to be leveraged in the digital realm. The physical process of accessing all historical documents pertaining to a specific topic in itself can be quite time-consuming, tedious, and costly. Relevant documents can be scattered across the globe in various archive locations, and documents are potentially accessed throughout several research projects by multiple researchers – each time creating this overhead of time for each interested individual. To reduce this overhead, and also to make historic documents more accessible to a broader audience, the process of digitizing and enriching documents is an established practice in the community of historical research. This has led to a high demand for more efficient solutions in digitizing, preserving, processing, and curating historical documents.

Digitization typically starts by taking high-resolution scans or photographs of a document. This represents a digital facsimile – a digitized high-quality visual reproduction that is as faithful to the original document as possible. For documents portraying mainly visual information in the form of pictures, film or illustrations, the next processing steps might involve automated classification of the contents. In the case of mainly text-based documents, their digital facsimiles initially come with no metadata and therefore need to be further processed to make the contained information digitally accessible. Thankfully, recent advances in text digitization and processing make it a very streamlined process to automate text recognition and annotation in such documents. These advances include language detection, optical character recognition (OCR), automated correction of recognition errors resulting from OCR, and named entity recognition (NER) to pinpoint names, locations, and events which might not be recognized first-hand by relying solely on grammar and sentence structure of analysed document text. This puts researchers in the favourable position of having access to digitized and annotated facsimile documents from archive locations worldwide and being able to curate diverse digital archives. While having access to a curation of digitized archive documents might be overall beneficial, it may also come with its downsides. For instance, searching such a digitized document

corpus without the proper interface can become a daunting task and might not yield satisfying results, since the corpus might contain highly domain-specific keywords and could feature documents written in a multitude of source languages. For instance, Citavi[1], a tool for literature and knowledge management, enables researchers to upload documents, and collaboratively extract relevant segments as well as annotate them. This and similar tools help alleviate the challenge of working with an extensive digital document corpus, but still requires researchers to actively find and subsequently link, and highlight parallels between documents.

As part of the Visual History of the Holocaust Project[2] (VHH project), researchers work on preserving, curating, and showcasing historical records of the Holocaust in the form of audiovisual and written documents by exploring digital technologies that increase the mass appeal and access of broader audiences to the historic source material and also help expert researchers interact with the material in a unique way. As has been previously stated, digital preservation tools have been rapidly improving and have also led to an influx of post-processed and annotated digitized facsimile text documents within the VHH project.

In an effort to prevent the pitfalls of being overwhelmed by a huge amount of digitized archive data and also to improve the work process of researchers, we identify the core workflows and unmet requirements of the researchers involved in the VHH project when it comes to working with digitized documents. We conduct an initial analysis and categorization of the gathered requirements and based on that, we devise a dynamic search user interface prototype geared towards intuitive navigation and deep exploration of topics across a digital corpus of multilingual text-based facsimile. By involving the researchers in our iterative design and development process, we ensure that all the high-level requirements are in line with our proposed technical solutions and are adjusted and improved throughout incremental feedback loops. The proposed prototype aims to enable semantic search by extending user search queries with domain-specific synonyms, while also extending the search area to all languages within the digitized document corpus. Based on query-relevant terms in a document, our tool presents dynamically generated image snippets of the facsimile document, which are fully interactive via the positional OCR metadata extracted from all documents when being imported into our tool. Therefore, text passages within a facsimile can be manually marked and can then be used to kick off new search queries or refine previous queries.

1.1 Contributions of the Work

The main contributions presented in this work are as follows:

[1] https://www.citavi.com/de
[2] https://www.vhh-project.eu/

- **Evaluating the Requirements of Researchers for Search Workflows on Digitized Document Corpora -** Even though our contributions as computer scientists are of a mainly technical nature, the requirements that inform our decision-making stem from the historians involved in the VHH project. More specifically, the requirements are first formulated from the typical workflow of the researchers and of course any desired improvements that are made possible by then applying these high-level specifications to technical requirements. The requirements should result in contributions that try to replicate the positive aspects of established search workflows, while leveraging the digitization of the document corpora and therefore aiding the researchers in their search tasks. The formed requirements are also adjusted in a close direct feedback loop with the involved researchers.

- **Processing Annotated PDF Facsimile Documents -** Document processing is a vital aspect of building an extensible digitized document corpus for later search and exploration. Our processing interface accepts PDF documents in general, but is tailored towards digitized facsimile documents that were annotated by OCR tooling before being fed to our interface. It is a common step in a typical researcher's digitization workflow to run OCR on freshly digitized facsimiles and receive a versatile annotated PDF file.

  Input documents contain two major categories of data, which are each extracted and processed separately. Firstly, the visual information of the facsimiles is extracted in the form of image files, which are later used for generating dynamic snippets. Additionally, we extract all the OCR annotation data to determine the text contained within a document and also where all the words and symbols are placed within a document's page. The gathered words are processed and grouped based on their according word families and are mapped to their layout coordinates within a page. All the extracted information is additionally stored in a search index to be accessible for search queries. Our index is organized in single document pages, which allows us to retrieve relevant pages instead of full documents to improve overall performance.

- **Extending Search Queries Through Translation and Synonyms -** Historical documents pertaining to World War II and the Holocaust are written in a multitude of diverse languages. While historians might be fluent in different languages, it can be quite cumbersome to have to translate and repeatedly launch the same search task if the goal is to retrieve all documents linked to a specific topic, regardless of source language.

  To ensure a seamless multilingual workflow for researchers, we devise two query extensions that are applied to search tasks launched against our index. All terms within an incoming search query are extended by being separately translated into an extensible set of (currently 10) languages. Additionally, we match the queried terms to a map of researcher-curated domain-specific phrases that represent synonyms to common search phrases in a multitude of languages. With both these measures

combined, we extend the search area across languages while refining the specificity of the query within the given domain.

- **Generating Image Snippets of Query-Relevant Document Sections Dynamically -** As a brief reminder, we extract both the visual representation of the facsimile and also the OCR metadata of all recognized text, including the layout position within a document. We can use this information as part of our search workflow concept to display documents that are relevant to a search task in the form of their digitized facsimile representation. Additionally, thanks to the positional information, relevant documents can interactively be annotated and searched and show highlighted phrases that are relevant to the search upfront. Another advantage granted by the OCR metadata is the ability to determine which sections inside a document page are relevant. We leverage this knowledge by providing tooling to dynamically crop relevant image snippets that can highlight the most important sections of retrieved document pages.

- **Intuitive Document Search User Interface Prototype -** Our proposed search user interface prototype combines a lot of the previously introduced contributions. Based on the information gathered during requirements collection and refinement, we devise a search workflow that incorporates facsimile representations as the primary way of interacting with the underlying digitized document corpus. We use the proposed query extension to enable multilingual exploration focused on the domain of World War II and the Holocaust. Resulting search hits show relevant image snippets from select document pages. The user experience is aimed at enabling exploratory search. Our navigation shortcuts and the interactive facsimile allow users to quickly jump between searches and documents, launch a new search task or refine a search task to dive deeper into a specific topic. All of these measures are intended to replicate the intuitive workflow of piling through a stack of documents and leafing through individual pages, while fully leveraging the extracted and processed data to enhance the process of researching a topic and finding links across documents.

1.2 Structure of the Thesis

Chapter 2 categorizes and summarizes the requirements reached in conclusion with the involved archivists and historians. We cover existing approaches in the form of tools and applications as well as prior research in Chapter 3.

Our proposed import workflow for extracting positional and text metadata from annotated PDF documents is detailed in Chapter 4. The different components of the search query extension, ranging from a language-independent search index and term-based search query translation to synonym extension, are discussed in Chapter 5. Chapter 6 highlights how we combine query result data and extracted positional metadata to determine, pool and generate relevant facsimile image snippets.

We conclude this work by proposing topics for future work in addition to defining the limitations of the thesis and summarize our contributions in Chapter 7.

Appendix A elaborates on implementation and software architecture details, whereas Appendix B introduces our user interface concept prototype which is intended for showcasing an intuitive search workflow in digital archive data.

CHAPTER 2

Analysis of the Requirements of Historians

To be able to provide approaches for digitized historical archives in tune with the actual requirements of the domain, we had to familiarize ourselves with the working routine of active historians. Our proposed work needs to match the latest historian workflow and future potential requirements as closely as possible. Oberbichler et al. [1] for example, explore interdisciplinary collaboration in digital historical newspaper archives and argue that reciprocal understanding of the involved workflows in the different disciplines ranging from computer science, to digital cultural heritage curation, and also to digital humanities studies, only serve to improve research sprouting from these collaborative efforts.

Therefore, we set up an initial digital work group meeting with historians involved in the VHH project and collaboratively worked out the first requirements for our digital archive exploration concept. The requirements of the concept were refined in incremental steps with successive meetings, at times aided by an early visual prototype and in later stages by functional prototypes in varying grades of completion throughout our feedback-based iterative development process.

There are various tools for annotating and searching through digitized domain-specific document corpora and, more specifically, digitized historical archives. In contrast to existing solutions, the general goal reached in cooperation with our historian stakeholders was to digitally replicate the intuitive workflow of leafing through documents and discovering connections between different documents and even throughout archives. With this in mind, the conventional workflow is extended through extracted textual and positional data to build an intuitive search and exploration tool, that allows direct interaction with enhanced digitized facsimile across multiple archive sources.

The list directly below briefly summarizes the categorized requirements, while the following subsections elaborate on each individual requirement:

- Input Processing
  - **Archival Facsimile Documents** Leverage and preserve input facsimile quality
  - **Source-agnostic** Unify input process for annotated PDFs regardless of pre-processing source
  - **Language Support** Support growing number of document input languages from various archive sources
- Search Experience
  - **Replicate Archival Exploration Digitally** Create a digital search experience close to physically exploring topics in an archive
  - **Extending search requests** Increase search recall by translating search queries and finding synonyms for searched terms
  - **Enhanced Facsimile** Use the digital replicas as an interactive tool for navigating through a reconstructed historic archive
  - **Human-in-the-Loop Workflow** Facilitate an intuitive participation-based exploration workflow
- Design & Architecture
  - **Explainable Results** Highlight the underlying search query modifications
  - **Modular & Expandable Tooling** Create a stable and easily adaptable set of tools surrounding the proposed prototype
  - **Performance** Observe and improve various performance metrics throughout the development and evaluation process

2.1 Input Processing

One implicit requirement arising during the conceptualization of an archive search tool is the general question of processing input data. The proposed tool should provide an interface for receiving and processing documents for later retrieval purposes. Beyond the question of designing an input interface, in the following subsections, we want to highlight the assessed challenges and goals regarding expected traits of the input documents and which traits are beneficial for further processing. For instance, challenges such as preferable formats and post-processing states in which data should ideally be delivered in and language support in a research project with ever expanding historic archive sources. Digital archives often are 'quietly incomplete', meaning that not all documents and

resources in a physical archive location will be digitally processed and therefore only a subsection of the actual archive can be interacted with digitally. Such archives, where researchers can't be certain, which documents are missing from the digital archive representation, are a major reason why some historians tend to prefer physical archive visits to digital archive exploration tools despite the convenience of digital archives and the travel expenses for cross-globe archive trips [2]. Therefore, it is of utmost importance to make the barrier of entry to include an archive document into a digital archive as low as possible.

2.1.1 Archival Facsimile Documents

Digital facsimile documents, which should be as true to the original document as possible, play a vital role in the preservation, curation, and exploration of historical archives. Therefore, such documents represent the basis of any visual interaction concepts presented in this work. By letting users directly interact with these documents and digitally augmenting the possibilities to interact with them, we strive to replicate the authentic experience of looking through documents in a physical archive.

For any facsimile input we receive in our application prototype, we intend to preserve the quality where possible and only scale down resolution or other aspects temporarily if required due to performance and usability considerations.

2.1.2 Source-agnostic

The laborious process of digitizing historic documents can start anywhere across the globe in one of numerous archives. Documents can be – based on their individual condition and archive restrictions – photographed or scanned. There are a multitude of tools for curating, annotating, extracting text information and processing archive documents. Historians and archivists can incorporate several of these tools in their digitization and curation workflow, i.e., for manual transcription or automated text recognition. While many documents are digitized facsimile representations, some additions to future archives could possibly be born-digital and therefore lack any additional visual indicators apart from the rendered text.

Accordingly, it can be a gargantuan task to support several input formats and even harder to maintain them. Our goal is to cover the largest area of potential processing outputs by supporting post-processed or born-digital PDF input files. Whether an incoming document has been merely manually annotated, or was sent through a sophisticated OCR pipeline, the necessary metadata can be extracted and processed from an input PDF in a fairly streamlined manner. Even though inputs and outputs throughout the workflow can be scattered across various file formats, we make the educated assumption, that born-digital files or digitized documents post-processing can be delivered to our tool in the form of a PDF. Further processing steps and challenges involving input PDFs are discussed in Chapter 4.

2.1.3 Language Support

Archives, where new digitization candidates are sourced, can span the whole globe. Historic documents, in case of the overarching VHH research project generally pertaining to the Holocaust and World War II, are preserved in various countries on different continents and were written in a multitude of diverse languages and scripts. This also does not account for future excursions into different topics and subtopics or the exploitation of new archive collections potentially containing a whole new set of languages and scripts.

As a result of this wealth of diverse inputs, an open challenge is to support the parsing and processing of most, if not all languages, that the tool is confronted with. Only with this goal in mind, the application and underlying tooling can fully represent and leverage the provided archive data. As a consequence, incoming archive documents are to be assigned their most likely source language based on a language detection tool, which in turn allows extracted text segments to be handled with the appropriate language pre-processing mechanisms and categorized before being integrated into our language-agnostic search index.

Aside from language support as a vital sub-task of input processing, there are also some considerations to make during the search process. Subsection 2.2.2 among other topics highlights the tasks surrounding language support during the search process.

2.2 Search Experience

An intuitive yet in-depth search experience should be an essential aspect of an archive exploration tool. Even though the requirements were worked out together with and mainly for historians, people with varying levels of expertise from students interested in the topic to experts should be able to capably use future tools derived from our proposed concepts and prototypes.

The following subsections highlight some important aspects we incorporated into our development of a search user interface (UI) prototype. The properties of the UI prototype are expanded on in Appendix B.

2.2.1 Replicate Archival Exploration Digitally

The physical experience of exploring an archive is something we strive to replicate digitally as closely as possible. After interviewing a sample group of historians from a research group, Force & Wiles [2] find that the queried historians seem to prefer physical documents to digital archives to – among other reasons – be able to turn the pages themselves. This notion was in turn echoed by the historians involved in the VHH project. Therefore, this work seeks to incorporate the extracted data in its entirety and replicate the feeling of physically exploring an archive as closely as possible. To achieve a similar experience, the proposed search prototype will visualize documents only through their

facsimile representation. By only interacting with the digitized archive through facsimile documents, we aim to create an involvement and immersion with the source data, which could not be achieved through interfaces based on text alone. This way, users can leaf through document pages and look at the written documents in a manner close to the physical experience.

While the use of facsimile to interact with archive documents is essential for our concept, it is not a completely novel approach. Existing digital archive user interfaces, which we review in Chapter 3 are for the most part designed for digitized newspaper corpora and display facsimile article clippings in search result pages. The novelty of our concept comes not from merely visually displaying facsimile, but rather by extending the classic digital facsimile capabilities of passively inspecting documents and allow users to investigate different topics across the whole document corpus and also across various source languages. Topics can be explored through explicit search commands or implicitly by interacting with the facsimile. Facsimile documents containing similar topics are laid out next to each other, and a previous search can easily be brought back to focus. In conclusion, all measures combined strive to replicate the experience of delving into an archive, chasing down documents pertaining to a specific area of interest, but additionally having digital amenities like automated multilingual archive-wide search, search backtracking and highlighting of relevant sections.

2.2.2 Extending search requests

One major assessed requirement for the resulting exploration concept prototype is the intuitive browsing of topics across a multilingual historical archive. Historians, archivists, and other interested parties alike should be able to launch a search request and cast a wide net for further investigation. With exploration at the forefront, the recall of relevant documents from the archive index becomes crucial. Incoming search requests should therefore be extended to increase recall, through various means.

Since we are dealing with a multilingual document corpus, we must account for language discrepancies between the query language and document languages. Consequently, language support is not only an important factor when processing input documents, but is also essential for the search experience. Search queries, similar to input documents, need to be run through language detection. Additional steps should involve query term translation to the most likely translation candidates across different languages present in the corpus. We can therefore use this extended multilingual query to allow for a more comprehensive ranking of document pages in other languages and an increased cross-language retrieval.

Aside from language-specific query extensions, we aim to increase recall by leveraging a project internal dataset of domain-specific synonyms. As one might expect, this synonym dataset contains terms and sayings phrased differently, but also occasionally provides translations for concepts that are hard to directly translate in other languages. These

terms can be incorporated into the query process by searching for base concepts and extending them with their synonyms from the dataset.

Both proposed measures of increasing the overall recall are discussed in-depth in Chapter 5.

2.2.3 Enhanced Facsimile

As previously mentioned, historical archive digitization is an enormous interdisciplinary undertaking involving multiple stakeholders across several gathering and processing steps, each with their challenges and pitfalls. At the end of this arduous process typically lies the question of how to further exploit the gathered data. The gathered data, in our specific case, written documents, can yield a high-quality facsimile representation of the original and extracted or manually annotated or corrected text. In one additional step, these documents can potentially yield semantic metadata in the form of dates and names of specific events, places, and people. Hawkins [3] even argues that the wealth of digitized archives is no use without properly incorporating and dealing with linked open data across archives, and we also see this as an important aspect to be covered in future work.

Digital facsimile documents on their own are great tools for introducing interested parties to the original source material and allowing domain experts to interact with archives by proxy. Still, we want to allow more ways of interacting with the source material, by extending the basic form of facsimile documents. During the archive document import, any embedded text fragments – either through manual annotation or automated recognition – are extracted with their specific position inside the respective document page's boundaries. Together with the historians and archivists, we assessed possible use cases in the search interface concept for enhancing facsimile documents with bounded text metadata.

One major benefit of having both a detailed digital facsimile representation and exact text positions for each document page can be gained by combining them. Overlaying the facsimile with properly placed invisible text-boxes opens up several interaction possibilities and yields much needed context that can save time compared to splitting OCR and facsimile representations [4]. Firstly, the interface can automatically visually highlight text sections that are relevant to the user's search query, but most notably, users can mark and highlight sections themselves. Manually marked sections can in turn be used as a steppingstone to search for a new topic or refine existing searches.

This enhancement concept can not only be applied to full document views, where users can explore facsimile documents page by page in their original scale. Search result views could also benefit from this, even though showing full document pages for each individual entry on the result page would potentially take up too much screen space and not give a proper overview. Accordingly, one additional challenge pertaining to facsimile enhancement is the adjusted presentation in search results. More specifically, we want to retain the interactive functions of auto-highlighting and manual marking, but only show specific relevant excerpts from single document pages. This way, users still get to interact

with facsimiles, while getting a good overview of several relevant document pages. We present our considerations and findings regarding facsimile enhancement in Chapter 6.

2.2.4 Human-in-the-Loop Workflow

Adhering to the previous and subsequent requirements, we strive to provide an intuitive and helpful archive search and exploration concept with this work that hopefully inspires further research in this area. Still, it is vital to properly convey – through this work and by extension in the prototype and accompanying tools – the reliance on the user as the main component in what basically is a Human-in-the-loop workflow concept. All tools necessary for digitally exploring the historical archive data are provided, but it is the user's responsibility to start with a research topic and work out dynamic links inside the corpus along the way.

Users are encouraged and required to work out their search goals themselves and are guided by a search interface promoting this workflow through the following steps:

- **Cast a wide net** Start search with an initial topic showing many tangentially relevant results aided by query extensions
- **Side-by-Side** Look at several documents or search topics simultaneously to detect new links between them
- **In-Document search** Manually highlight text sections in documents and search result snippets
  - **Deep dive** Find new topics inside another document
  - **Refine** Filter the current search topic by adding new parameters and change the found results
- **Search history** Revisit prior searches not to lose fleeting thoughts or branch into another search direction from previous search paths

2.3 Design & Architecture

In the following subsections, we elaborate on any additional agreed-upon design or architecture considerations involving our presented concept. These considerations, while mentioned in early meetings, were refined and iterated over throughout the development process. Therefore, the subsequent sections also mention more implementation-specific aspects than the preceding requirement sections.

2.3.1 Explainable Results

Any outputs of our work should be reproducible, easily interpretable, and, in case of the prototype, aided by visual representations of the relations between different query terms.

It was also an explicit requirement of the historians to provide means to visualize the internal query modifications, which we will further elaborate on in Chapter 5. Based on this requirement, we also provide an optional view for each query, that demonstrates the natural language processing steps (NLP) conducted on the query terms, as well as query extensions based on language translations and synonym dataset matches. The manual modification of the native underlying query, which is internally created after a user launches a search query, was brought up during requirements elaborations to support expert users. As of this writing, this remains an open challenge, but we argue that our concept strikes a good balance between query transparency and usability and urge future work to provide more concepts also geared towards expert users.

2.3.2 Modular & Expandable Tooling

As a direct result of our requirements analysis, we aim to produce a prototype encompassing all gathered and discussed goals and providing a visual representation for the discussed concepts. The resulting search prototype draws upon an orchestrated multi-container architecture. As a consequence of the orchestration, existing services can be easily scaled up from a prototype trial setting to production requirements.

Each microservice contained in the architecture deals with different areas of concern and can be used independently. The modular nature also allows to decouple different steps in the input processing and visualization process, which opens up a multitude of expansion options. For example, while our current input method of choice relies on annotated PDFs, future additional modules could support new input types, or use the underlying data for an entirely different search interface. This reinforces both feature robustness, because of the possibility to focus on core capabilities, but also allows for flexibility to adapt to future requirements.

2.3.3 Performance

As previously described, the prototype development process is intended to be of incremental nature, interlaced with stakeholder feedback for various milestones. This process based on various iterations is vital to keep the actual implementation aligned with the defined requirements. Equally important is that the fulfilment of requirements may be grasped differently by some parties, or the perception of the defined requisites may change in light of actually implemented features.

One aspect of the essential takeaways from feedback in incremental steps should be performance considerations. Since one vital component of our work is to develop and evaluate a search interface prototype including the necessary backbone architecture, performance needs to be monitored along the way and improved, where possible.

During input document handling and dynamic query-time snippet generation, computing- and memory-intensive image editing and PDF parsing processes are at work, which need to be kept at minimum loads to maintain server stability and support request scalability and overall response times.

While response times might be a universal requirement, the user interface needs to provide a fluid search experience across all different modes of interaction. Given the decision to develop our prototype user interface as a web application, miniscule changes in visualization can lead to incompatibilities in any of the popular browser stacks.

2.4 Summary

To conclude, we gathered the requirements of the historians involved in the VHH project for a novel user interface concept focused on exploring digitized historical archives containing written documents. Over the span of several meetings and discussions, we established the importance of digital facsimile documents as the foundation of our search experience. On top of working with digitized representations of the source material to provide an authentic experience, we devise an interactive user-driven workflow for investigating topics and navigating diverse and multilingual document corpora. This workflow is supported by enhancing the facsimile documents with positional and textual metadata to allow users to highlight and search sections directly inside the visual representation of the documents. By enhancing facsimile documents in our user interface concept and translating search requests to find relevant documents across languages, we hope to mimic the process of physically exploring an archive, while providing helpful digitally aided improvements.

CHAPTER 3

Background and Related Work

Within the sections in this chapter, we aim to present prior research and approaches that either serve as a basis for our methodology in this work and its findings or provide interesting parallels to our prototype implementation.

Since our approach significantly benefits from pre-processed archive documents, Section 3.1 introduces some methods for processing text in facsimile documents. We explore various endeavours of experimenting with different ways of visualizing information in user interfaces packed with dense information in Section 3.2. Digitized historical archives can contain various kinds of source data ranging from film material, and metadata of past events, to digitized newspapers and written documents or manuscripts. Some ways to go about presenting extensive archive corpora are highlighted in Section 3.3. As a consequence of the multilingual nature of our archive corpus, we briefly inspect findings surrounding multi- and cross-lingual user interfaces in Section 3.4.

3.1 Text Document Processing & Correction

Accurate and efficient text processing represents a vital step in the process of bringing digital archives to life. As previously mentioned, Oberbichler et al. [1] examine workflows and communication inside digital archive project spaces across disciplines. They argue that poor initial OCR results should be viewed as a beginning stage to build upon. Fixing poor OCR results can take different forms, from manual correction workflows, to re-OCRing digitized documents and employing OCR post-correction pipelines.

Looking back almost a decade, Tranouez et al. [5] present DocExplore, a unified tool for manuscript management. DocExplore is meant to assist users as early as uploading facsimile manuscripts, manually annotating or transcribing them combined with OCR-guided suggestions. While the tool allows search based on OCR, word spotting and additional indexing metadata and features a functional facsimile viewer, its focus lies

on its curation and authoring capabilities, which enable users to create multimedia presentations straight from the archive.

3.1.1 Re-OCRing

As early as 2011, in an attempt to improve existing solutions in the field, Marx et al. [6] look at an established web portal containing over 250,000 OCRed facsimile archive documents. They notice slow loading speeds due to full documents being downloaded to just view them and no meaningful query-relevant snippets being shown in the results list. With this in mind, they build a tool which reconstructs the PDFs together with the OCR data into more manageable file sizes and also succeeds to increase readability for blurred text passages. They also propose to add an intermediate solution between generic document summaries and the full document, by showing a quick view containing the OCRed text without layout information.

With Tranksribus [1], Colutto et al. [7] provide a comprehensive tool for automated manuscript processing, text recognition with manual transcription or correction, and subsequent document search. The automated text recognition pipeline based on layout analysis and handwritten text recognition (HTR) provides pre-trained neural models out of the box, but the models can also be individualized by training on custom datasets. While initially intended for HTR, the OCR process behind Tranksribus proved to also work considerably well for printed text recognition and re-OCRing of already digitized texts and was therefore extended and incorporated into the NewsEye project[2], which aims to improve digital cultural heritage mainly in the form of newspaper archives.

Staying in the topic of neural models for manuscripts, Wilkinson et al. [8] propose a deep neural network approach for word spotting, i.e., detecting word segments and putting them into a word embedding space, where similar words can be searched and highlighted based on their similarity to the original query. The tool provides some transparency to the embedding proximities between matches based on the intensity of the highlighting colour.

3.1.2 OCR Post-Correction

For large archive corpora, re-OCRing can be a very daunting task. In these cases, measures to improve OCR output by employing post-correction workflows can be essential. OCR-D proposed by Neudecker et al. [9] is an extensive open source OCR framework. While containing conventional OCR capabilities, it also features an unsupervised neural OCR post-correction model trained on historical German ground truth data. Their model is based on the noisy channel concept, which aims at finding correct versions of scrambled words in the context of surrounding text. Even though such post-correction measures can greatly reduce spelling errors and wrongly recognized words or characters in a target

[1] https://readcoop.eu/transkribus/?sc=Transkribus
[2] https://www.newseye.eu/

corpus, Neudecker et al. also supply the option of manually reviewing corrections to mitigate false positives in automated corrections.

Hämäläinen and Hengchen [10] devise a similar unsupervised neural post-correction method leveraging neural machine translation. To train their model, they create a method of extracting parallel training data from OCRed corpora by grouping and comparing semantically similar words with their respectively found OCR errors. This model is not language-specific and can therefore be trained to support other languages than English, as in their published model.

3.2 Search Result Visualization

How to present information from a diverse and complex archive corpus is an essential part of enabling interaction with the underlying source material.

Théron et al. [11] survey the development of different visualization methodologies in historical lexicography, based on tools employed by the Royal Spanish Academy (REA). Reviewed approaches range from exploring the use of a word across history shown in snippets in a simplistic text-based interface, to word-stem trees and geographically pinpointed uses of words over time optionally aided by natural user interfaces based on gesture navigation. In their findings, they emphasize a core concept of information visualization, namely 'Information Overload'. Given the wealth of possibilities to visualize digitized data, designers, and engineers alike might be prone to over-stimulating users with too many data points and a multitude of complex ways to interact with the information. Yet, it is highly advisable to constrain information visualization to the range of human capabilities when it comes to ingesting information and the focus and the attention required to properly processing relationships between data points.
In the same vein, Chen et al. [12] conduct an empirical study on multi-view visualizations across numerous publications to figure out positive patterns and negative trends among data visualization interfaces. Amid many other discoveries, they find that too many sub-views in an interface go hand in hand with decreased usability, and most surveyed interfaces resort to two- or three-panel views at most in their interfaces to keep things simple.
On the opposite end of the spectrum of information visualization, Windhager et al. [13] review and categorize information visualization tools and research projects, that don't go the conventional retrieval route of employing grid-based search result pages and allow more innovative interactions with the underlying source material. As a result of their findings, they produce design principles for cultural heritage visualization and, aside from other suggestions, advise tailoring tools specifically to the structure of the underlying source data and the tasks and requirements in the local space of each project.

Hoeber [14] further punctuates the importance of an expressive visualization language for interactive information retrieval and conducts a brief survey into past interface approaches he himself was involved in. Take for example the search UI concept HotMap [15], which

presents a search user interface that shows an occurrence frequency heatmap for each query term and lets users sort based on individual query term frequencies.

Another approach comes in the form of Bow Tie, where Khazaei et al. [16] suggest a library search application, which visualizes backward and forward citations next to search results by using citation metadata mappings between documents. This citation visualization, presented – as the name suggests – as bow ties, also grants the ability of lateral navigation between these documents.

Another feature is the possibility of query refinement. By looking at a keyword histogram of articles of interest, users can toggle relevant keywords for future searches. While citation-based visualization and refinement may not be directly applicable to OCRed facsimile archive documents, the process of moving laterally through a document corpus by intuitively refining query parameters is highly reminiscent of workflows we propose in our prototype in Appendix B. Providing more refinement through metadata is of course in high contention to be a useful feature update in future iterations.

With KLink, Shukla et al. [17] propose a search user interface that explores a novel way of helping users navigate digital academic libraries. Next to each search result, users can see keywords provided by the authors, which can be easily used as facets to visually enhance the search. By toggling these facets, users can visually highlight other documents tagged with this keyword and filter existing results. In addition to that, documents can be placed in individual modifiable workspaces to be able to revisit and extend previously explored topics. Even though KLink uses workspaces and search facets to allow backtracking and exploration, we argue that this approach follows a similar philosophy to our combination of search refining and convenient search history navigation.
In terms of faceted search, di Sciascio et al. [18] go a step further with their proposed tool uRank. Search result pages can be fully customized by users by adjusting the relative relevance of each single term from the initial query through intuitive weighing sliders. Adjustments don't alter the result page instantly, but rather highlight the affected result items and how they are going to move in the ranking if the change is propagated. An additional sidebar with summarized keywords extracted from the results can be used to add keywords to the ranking on the go. With these features, uRank offers result transparency and fine-tuning to users.

Discourse surrounding digitized material can get very complex and requires tools supporting the analysis and recording of narratives and arguments arising from prior text analysis. Viscourse, a tool proposed by Martin-Rodilla and Sánchez [19], provides a novel take on extracting and linking text segments from digital documents and visually linking them to support an argument or capture a discussion.

3.3 Digital Archive Visualization

Historical archives can cover a large timespan, cross the boundaries of language, and often consist of several distinct types of input formats that each benefit from different

ways of presenting them.

3.3.1 Newspaper Archives

One highly researched subsection of digitized archives involves historical newspaper collections. Through their findings, Ehrmann et al. [20] bring out an interesting parallel between historical newspaper archives and our endeavours to visualize historical archives with a more diverse corpus. They conduct a survey of interfaces for digitized historical newspapers and gather the various features available in these interfaces. One significant aspect in the context of our work is how many interfaces deal with facsimile display and OCR text. Most surveyed interfaces provide some form of snippet previews and search-relevant highlighting in facsimiles (83% and 79% respectively) and over half of them offer the option of displaying OCR text in some shape or form. Concerning post-OCR features, automatic post-OCR-correction as well as user-suggested corrections seem to be an emerging trend by the time of this article (2017), but have not yet fully caught on. To conclude, Ehrmann et al. show several well-established and emerging trends in visualizing and interacting with facsimile in digitized historical newspaper archives.

Late & Kumpulainen [4] supplement these previous findings by carrying out a qualitative study with historians that mainly use digital newspapers during their research process. They interview the involved historians on their workflow when dealing with digital surrogates. One remarkable finding is how the historic scholars deal with digitized OCRed manuscripts or generally hard to read documents. In exemplary tools, that offer both a facsimile view and OCR text, scholars tend to switch back and forth continuously, since the facsimile view provides great insight for article boundaries, while the OCR text can at times supply better context for unreadable passages, while at other times only the combination of both views gives enough context to comprehend the articles.

We find that interactive digital facsimile in archive search are not yet fully leveraged in more general historical document archive tools, and hope to learn from problems detected in the newspaper archive space and finally provide inspiration for future research in that area by presenting our facsimile- and OCR-focused prototype.

Hebert et al. [21] devise a Platform for Indexing and Valorizing Journal Archives (PIVAJ). To put it more simply, PIVAJ employs a pipeline that takes archive newspaper facsimile pages, extracts text through a collaborative OCR correction tool and automatically segments pages into separate articles [22] via supervised machine learning modelling techniques (i.e., conditional random fields). The collected and categorized data is finally presented in a search user interface, which contains a few interesting ideas. While having features such as ad-hoc recognition error corrections and adaptive zoom resolution for facsimile detail views, it still employs conventional text-based document summaries in the search result list and lacks depth in terms of search exploration user experience.

Kettunen et al. [23] take PIVAJ a step further and leverage its capabilities by allowing users to store and catalogue newspaper clippings dynamically, including extracted and potentially corrected text created from the previously automatically segmented articles.

NewsEye [24] is a research project dedicated to discovering new ways of interacting with digital newspaper archives. Under the umbrella of this project, Jean-Caurant and Doucet [25] introduce a proof of concept user interface. What ties in to our proposed prototype is the incorporation of facsimile snippets in the search results. While the snippets are not dynamically created, but rather represent article segments, using the facsimile directly as surrogates in the search result is akin to our philosophy of interacting with the source upfront instead of hiding it behind extracted text. Aside from that, the concept offers interesting features, such as a customizable user workspace, where inserted articles can be manually labelled based on relevance, and trained topic models that distil topics from datasets. These topic models enable the discovery of new topics and the consecutive creation of fully automated topic distribution reports in natural language.

3.3.2 Spatial Archive Data

Caserio et al. [26] demonstrate an archival exploration interface built on top of semantic metadata. This enables users to investigate historic individuals, events, and places through interactive maps and cross-linked entries, which further promote exploration.

SpaceWars, proposed by Gutehrlé et al. [27] provides a related map-like exploration of historical events but achieves it through different means. Instead of relying on semantic metadata, the pipeline behind SpaceWars searches through digitized text from historical newspaper archives (NewsEye) to find historical geospatial references and link the source articles to the visual map locations.

3.4 Multilingual Search User Interfaces

The digitized historic archives processed in this work span a range of different languages. While in their work Chu & Komlodi [28] stress the distinction between cross-lingual and multilingual retrieval to signify the user's multilingual skills or a tool's capabilities to translate queries, we use the terms synonymously to indicate both retrieval of documents in languages apart from the query language and the support of multiple languages when formulating a search query. Aside from that, Chu & Komlodi devise TranSearch, a multilingual search interface concept offering automated query translation, manual translation reformulations for polyglots, and a highly customizable result layout. Based on their concept, they evaluate different layout options in a qualitative study and find that simplicity, visibility of result item language, and layout customizability are useful aspects in multilingual search interfaces.

Similarly, Steichen & Freund [29] conduct a crowdsourced evaluation with over 800 participants and later on, Ling et al. [30] conduct a lab-based user study with 25 participants with both studies evaluating the same multilingual search interface layout archetypes. Participants had a choice of four layouts, a tabbed interface to switch between languages, a panel interface showing all result languages in distinct panels side-by-side, and two interfaces with single lists, a fully interleaved option, and one with languages

being grouped in the single list. Both studies found clear language separation to be preferable to mixed language results, and participants overall had the best experience with a single list containing separate groups of languages.

CHAPTER 4

Import Process for Annotated PDF Facsimile Documents

The importing and processing of input documents represents an integral step in making our digital archive exploration concept possible. As elaborated in Chapter 2, our import workflow is currently built to be able to process PDF files specifically. Incoming PDF files are digitized facsimile documents originally from historical archives. During the import process, no optimizations are employed that are geared towards documents from such archives specifically, but the synonym mappings used to extend the search query contain domain-specific phrases and are briefly mentioned in Section 5.2.2. Additionally, to be able to offer all features in our user interface prototype, documents should also be processed in an OCR tool, or otherwise annotated with positional and textual metadata, before feeding them to our import tool. The metadata contained within the resulting

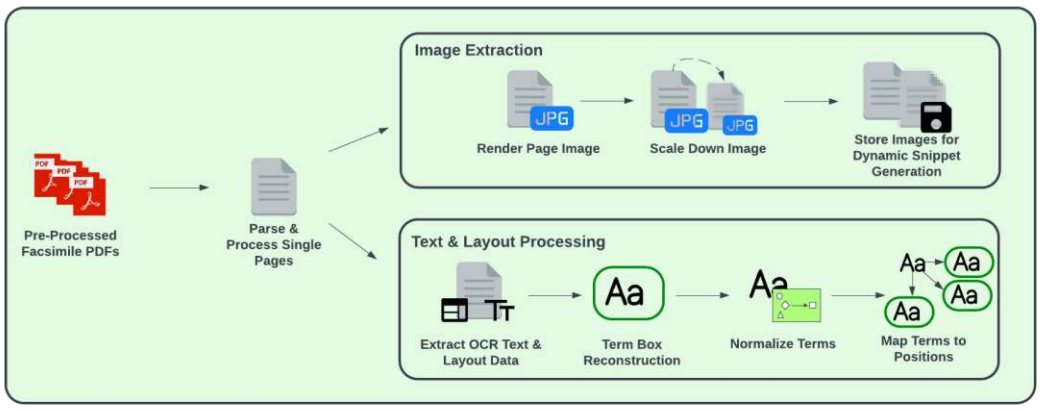


Figure 4.1: Facsimile PDF Import Process Steps

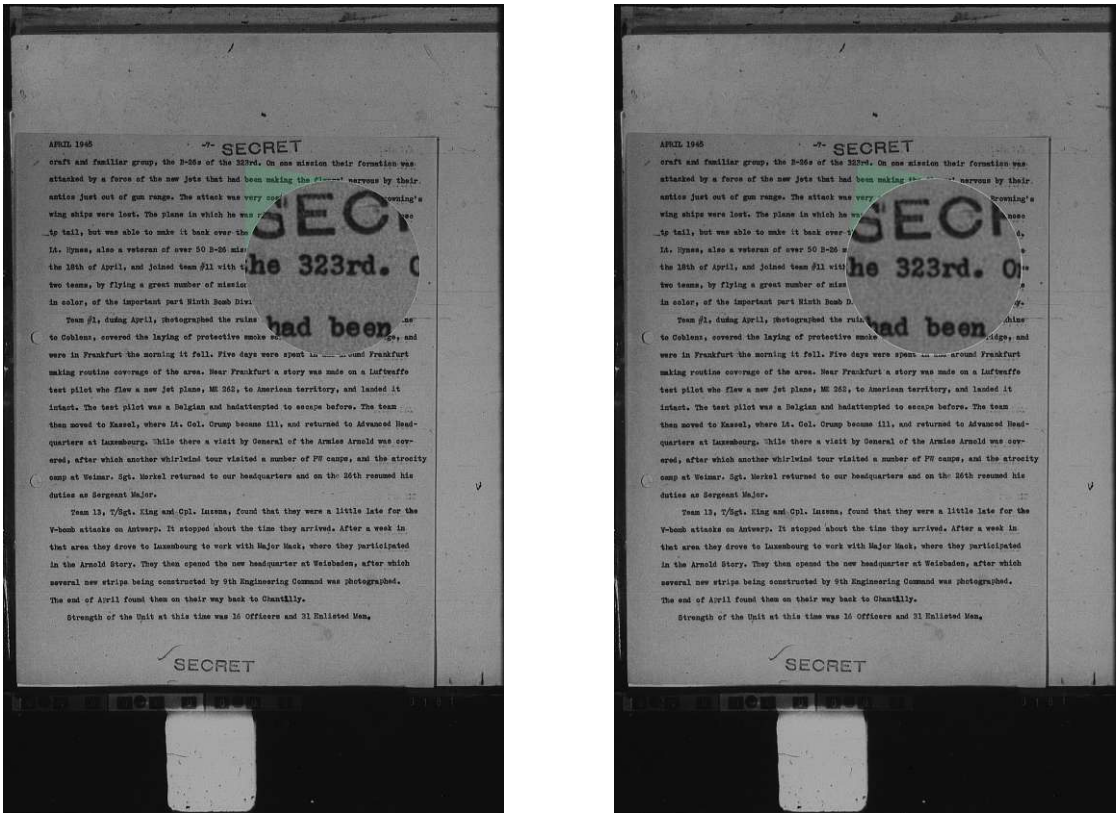
PDFs, which we intend to use as input for our search application, should then contain information about the recognized or transcribed document text and where each character of a word is located inside the digital facsimile's bounds. Once our import process is kicked off with an input document according to the previously established specifications, there are several steps to be taken. Figure 4.1 showcases the different processing steps.

We take deliberate measures during the import process in preparation to make further processing more convenient and efficient because we front-load some of the most resource-intensive tasks such as parsing the PDF files and extracting images from them. PDF files are parsed page by page, as source documents could potentially contain dozens or hundreds of pages. Since we treat document pages separately, this can be a low effort mitigation technique against exceeding potential memory limits for large files loaded and parsed all at once, while still allowing for future distributed approaches, or approaches aimed at higher resource capacities, to scale up our existing process.

4.1 Image Extraction

After an individual document page is parsed and loaded into the runtime, we need to convert the page to an image. Based on how our user interface prototype is designed in accordance with the requirement to digitally replicate archive exploration, the facsimile representations of the documents are a vital component of interacting with search results and the relevant documents themselves. With our plans to dynamically crop relevant page sections into image snippets, we have to run several file modifications on each page during runtime. To reduce the overhead of these file operations, rather than working with the source PDFs directly, we convert the individual PDF pages into an image file format such as JPEG, even though we initially lose the inherent interaction capabilities of already textually annotated PDFs.

The first step includes a direct conversion from the singled out document page to a JPEG file in the source file's native resolution. As we have previously stressed in Chapter 2 when talking about performance requirements, the development process contained periodic performance assessments throughout various aspects of our prototype design. One of the results from initial dynamic snippet creation testing showed a noticeable increase in response time and peak memory load for extracted images with higher native resolutions. To reduce the required resources for post-import snippet generation, we added another processing step to scale down the image resolution for all processed pages. Images are scaled down to 25% of the native resolution up to a lower threshold of either the width or height reaching 1500 pixels, while still preserving the aspect ratio. An example facsimile document page can be seen before and after scaling down in Figure 4.2. Even though at a glance, both versions provide a similar level of readability, the magnified sections show visibly blurrier line edges in the scaled down page. Still, the reduced resolution and resulting file size leads to faster load times and a significantly lower processing footprint for snippet generation, and therefore greatly outweighs the slight step-down in sharpness. In our tests throughout development, we found this scale factor to strike an acceptable



(a) Original resolution (2656 x 4288) (b) Scaled down resolution (1328 x 2144)

Figure 4.2: Example document before and after downscale

balance between improved performance and quality preservation.

The higher quality image is still widely used in our prototype to display full document pages, while the lower resolution version is used solely for snippet generation purposes. The snippet creation performance is described in more detail in Chapter 6.

4.2 Text & Layout Data Processing

If the imported PDF is the result of OCR processing, or was annotated through other means, the loaded and parsed PDF page should contain a collection of the recognized characters and their respective position on the page. With PDFMiner[1], the PDF parsing tool chosen for this work, we can extract the full text and all character positions from each individual document page. The page text is in turn used for building our text-based search index, which generates the search results that serve as the baseline for query extensions (Chapter 5) and dynamic snippet generation (Chapter 6).

[1] https://github.com/pdfminer/pdfminer.six

The initial composition of the character position data is structured in boxes representing lines or layout regions detected based on layout analysis ordered from top to bottom and the characters inside the line boxes ordered based on the reading direction (default: left to right). This mostly flat and ordered structure is beneficial for efficiently constructing an interactive visual overlay of our facsimiles because we want to allow users to manually highlight text sections in the correct reading order in our interface. However, another goal is the automated highlighting of relevant sections in the document pages from the search result. An inverted mapping from the words to the respective positions in the text lets us retrieve relevant words and text sections more efficiently.

For this purpose, we iterate through the line boxes and construct word boxes from the character boxes that constitute a word. We tokenize the list of characters into words based on whitespace characters pre-established as likely word separation signals, and also choose to filter out special characters, such as dashes, slashes, and ampersands, that hinder indexing and retrieval of similar word constructions. In the next step, we use the detected language resulting from analysing the full-page text to be able to categorize morphologically similar words in the text. According to the detected language, we strip down the tokenized words by stemming them to the most likely morphological root form and normalizing any format inconsistencies. Once the words are in their respective base forms, we distribute all collected word positions to the appropriate word roots to create our inverted mapping. Listing 4.1 shows a simplified approach for the previously described process in Python language notation. In Chapter 6 we go into more detail on how we leverage this data structure to quickly visualize relevant text sections.

Listing 4.1: Tokenization and inverse position map construction

```
word_position_map = {}
# Fill above dictionary/map with a mapping from word stems
# to lists of original words and their bounding box
for line in lines:
  for character_data in line:
    # Extract char and char position
    character, character_box = character_data
    word = ''
    word_box = (0, 0, 0, 0) # Bounding box coordinates
    if character not in separator_characters:
      word += character
      # Create new box bounds including new character
      extend_word_box(word_box, character_box)
    else:
      # End of word reached
      # Normalize and stem word
      word_stem = normalize_and_stem(word)
      # Map word stem to the original word and its position
      word_position_map[word_stem].append(word, word_box)
```

We make a deliberate decision to logically separate the extracted data. This decision results in different storage mechanisms for the raw page text data and the processed bounding box data. This allows the text-based index to provide a fully functional search experience without being coupled to our visual concept. Our facsimile-based visual approach can build upon results from the text-based index, but in partially decoupling these two sets of data, we leave the possibility of future work leveraging our search index and query processing, while potentially pursuing a novel visualization approach.

CHAPTER 5

Query Processing

After introducing our document import process, we elaborate how the gathered data is persisted and queried. Our earlier established goals are to support several languages for incoming documents, and for user queries launched in our search prototype. We present our approach to support multiple input and query languages and different modes of search in our prototype in the following subsections.

5.1 Query Structure

Our proposed query process allows two basic modes of operation. Either mode A, a simple search query that contains a single search phrase, or mode B, a query composed of multiple independent search phrases. The distinction behind these two modes can be observed on the user-facing side and also how they are handled internally in our query processing and during query extension.

The intention behind a simple single phrase query is to start a new independent search. While the query phrase can contain several words, partial matches of the whole phrase are enough for documents to be ranked and retrieved. Each word from the query *can*, but does not have to be included in the retrieved documents, even though any matched word contributes to the overall ranking of the documents. Retrieval ranking currently uses BM25, a ranking function that relates the frequency of relevant terms in a document to the overall occurrence frequency of the terms across all documents in the index, which in our case contains individual pages of digitized archive documents. Query mode A can therefore be equated to placing each word in a query phrase into a logical disjunction.

On the other hand, query mode B comes with the intention to let users refine existing searches of mode A. When users have already launched a query but find an interesting new topic they want to search within the bounds of the existing query, they can fine-tune the query with an additional subquery. Mode B queries therefore are composed of mode

A subqueries. While internally, the subqueries are handled like mode A queries, each individual subquery needs to have at least one corresponding match in the retrieved documents. This choice leads to the retrieved documents always being a subset of either subquery and an intersection of all provided subqueries, which in turn equates to a logical conjunction.

Figure 5.1 shows a comparison of the two query modes based on how they would match the same document page. In the first scenario, we have a single query in query mode A, searching the phrase 'ambush attack', while the second query builds on the first phrase to build a composite query in query mode B adding the phrase 'night'. Due to the permissive nature of query mode A, the document page would result in a match based on several terms matching with 'attack', even though no mention of 'ambush' is present. While this suffices for the first scenario, the restrictive nature of query mode B filters out the page, since no match for the second part of the composite query can be found, which in turn makes sense when a user wants to refine the initial query by potentially finding documents that report on ambush attacks happening at night.

On top of both query modes, our query processing offers the ability to apply property filters based on metadata properties stored alongside the indexed page text. Such filters can be appended to existing queries through a simple conjunction, and therefore it is easy to extend the current query structure with additional filters in the future. Currently, this includes the option to filter based on the document source language or to restrict the search to one single document altogether. Restricting the search to a single document can be helpful in exploring topics within a large document consisting of several dozens of pages.

5.2 Semantic Query Extension

As we have detailed in Chapter 4, our import process detects the most likely language of a document page before feeding the page text into our search index. This source language information is also persisted to the index as part of the metadata registered for each page. Aside from this information being used for enabling user-driven result filtering based on preferred languages, it can also be used to weave in language-specific and synonym extensions into the previously described query structure. We do not further distinguish between different source languages when feeding documents to the index. In Appendix B, we demonstrate how we highlight query extension mechanisms in our user interface prototype.

5.2.1 Term-based Translation

One measure to increase our recall across a multilingual document corpus is to extend the basic structure described in Section 5.1 with term translations. In place of the basic query terms in the initial structure, we set a list of translations of the term, which are in turn equivalently ranked during retrieval. For this purpose, incoming query phrases are run

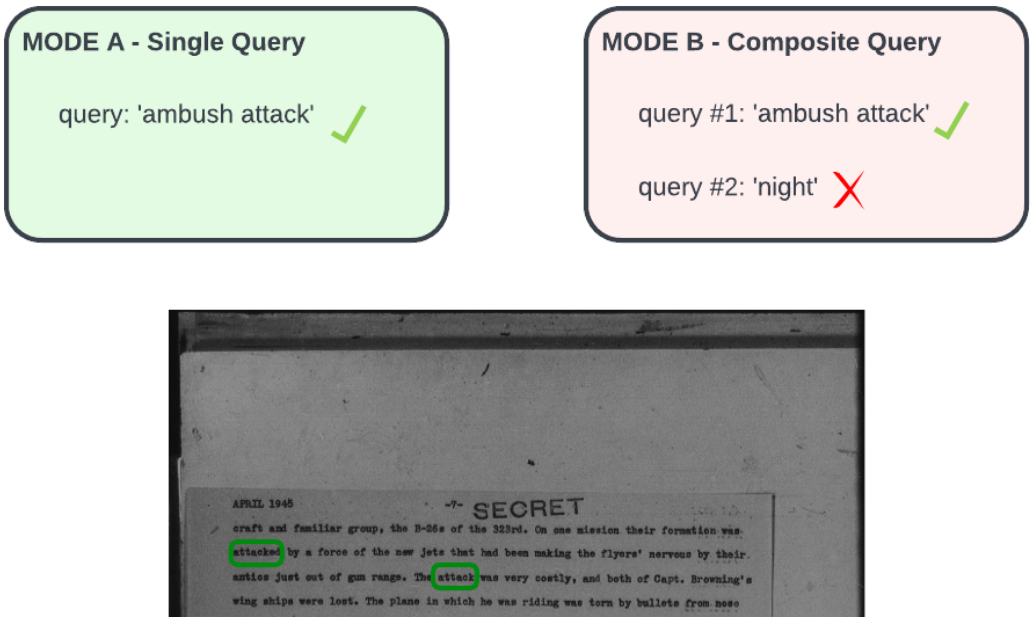


Figure 5.1: Match comparison of query modes based on a common document page

through language detection to determine how to translate the query terms. Longer queries yield an increase in detection accuracy, but the translation process falls back to English as a starting language. Each term in a query is translated individually into all available languages and added to the list of the equivalent term translations that are considered for finding matching document pages. Our translation process is based on a lexicon extraction model devised by Choe et al. [31] that yields a dataset of over 3500 bilingual lexicon language pairs. Within our solution, we have chosen to support 10 languages, which were extracted from a digitized archive test data excerpt used throughout development. While it would be ideal to support all possible language pairs to enable direct translation between all languages, we have resorted to using a reduced set of languages pairs. With word2word [1], the tool wrapping the previously mentioned lexicon dataset, each language pair can be loaded separately into memory. Each additional language pair constructs an interactive data structure during runtime that in turn increases the runtime memory load, and the intention to support all bidirectional language pairs to directly translate between 10 languages increases the memory overhead and the initial loading time of the translations service. The number of overall language pairs can be calculated through combinatorics, by viewing the process of pairing languages in one translation direction as a *variation without repetition*. With our base set to choose from being 10 languages and our choice containing 2 languages for each pair in one direction, we have the following

[1] https://github.com/kakaobrain/word2word

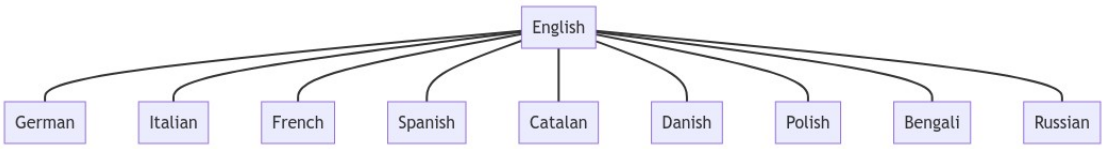


Figure 5.2: Supported language translation pairs for query extension

result:

$$\frac{n!}{(n-k)!} = \frac{10!}{(10-2)!} = 90$$

This means a full adoption of all possible pairs between the 10 languages would result in 90 pairs. Our reduced variant using only bidirectional language pairs from and to English results in 18 language pairs and an overall memory overhead reduction of 80% compared to a full adoption. Figure 5.2 shows the supported bidirectional translation pairs. With this approach, at the cost of having to translate incoming terms to English first before further translating them to the target language, we can reduce the needed memory overhead significantly.

Keeping the index mostly language independent creates the possibility to handle translation-based query extension in two different ways. More precisely, the discussion revolves around whether to indiscriminately retrieve all translated terms from documents in all indexed source languages, or to target each specific source document language with the translated terms in that respective language. Both options come with their benefits and disadvantages. The language-independent variant greatly benefits in cases, where documents contain foreign language words or phrases, that otherwise would not be retrieved. On the other hand, there is a high potential for retrieving false positives, where phrases in foreign languages have a matching word stem, but an entirely different meaning. One such example that surfaced during a prototype test iteration was the English noun 'war' matching with the conjugated form of the German verb 'sein', 'war', which equates to the past tense form of 'to be'. This effect could be counteracted by letting users manually strike such false positives dynamically in query results and adjust the query accordingly, or by defining a blacklist of cross-retrieved words beforehand.
In contrast, the language-specific version yields a higher rate of true in-language matches, while losing the ability to retrieve terms in different languages than the document language. To pick up the previous example, an English input query containing 'war' would search for 'war' in English document pages exclusively, while the German translation 'Krieg' would be retrieved from German documents only. Such a language-specific approach requires some additional query extension overhead. The produced overhead consists of the requirement to branch up each individual translated term and pair it up with a document language filter, and finally extend the query with each term-filter-pair through a conjunction. This proposed language-specific approach serves as mitigation to prevent users from having to circumvent internal query issues by having to manually strike bad cross-language matches, even if they miss an occasional fringe document with this variant.

5.2.2 Synonym Extension

The synonym data structure is represented as a collection of base phrases, which are each in turn mapped to a collection of domain-specific multilingual synonyms of the base phrase. Similar to the approach for extending terms with their translations, terms within the query are compared against synonymized phrases. Any synonym matches result in an extension of the query structure. Synonyms of terms are combined with their source terms in the query, and any detected matches in target documents are ranked equivalently to a source term match. Multilingual synonym extension, next to the term translations, provides an additional layer of increased recall across document source languages, but, in this instance, is aimed at domain-specific phrases, which we could not account for through conventional translation alone. In our user interface prototype, synonym matches are highlighted in a specific manner to visually distinguish them from terms and translations. This visual design choice, and also the way we inform the user of query extensions, are showcased in Appendix B. All term translations and found synonyms are subsequently passed on as metadata in the search results, since this information is vital for snippet generation and highlighting.

CHAPTER 6

Dynamic Generation of Relevant Facsimile Page Snippets

Digitized facsimile are the foundation of our visualization concept and as such can be fully viewed and explored in our user interface prototype. However, utilizing high-resolution facsimile in a user interface that should ideally feel quick and responsive can have its downsides. In Chapter 4, we detailed how we prepare and visually downgrade the gathered facsimile data to remedy later processing bottlenecks. This chapter in turn covers the processing of these prepared facsimiles. The core idea of our processing is to dynamically crop snippets from relevant document pages that contain the most important sections of a page, instead of rendering full pages for relevant result hits. This step is taken to visually declutter the search result overview by preventing a noisy result view containing full document pages. Additionally, we can defer rendering the high-resolution original facsimile to an optional detail view that can be reached via the individual search result snippets.

There are several steps that comprise the snippet generation process. Starting with a user query, we retrieve a list of relevant document pages and query extension metadata during the index polling step. Based on the relevant pages, we load and restructure the previously stored page images and mappings from document words to box positions in the page during the next step. During snippet candidate collection, we find relevant terms and phrases that match the query terms, their translations or any relevant synonyms. We construct boxes that wrap the candidate word boxes with a vertical padding and spanning the document width. In the following step, we combine any overlapping or adjacent candidate boxes to prevent redundancy in the finally generated and presented snippets. Figure 6.1 shows an overview of the mentioned steps, while the following Sections will highlight each step of the process in detail.

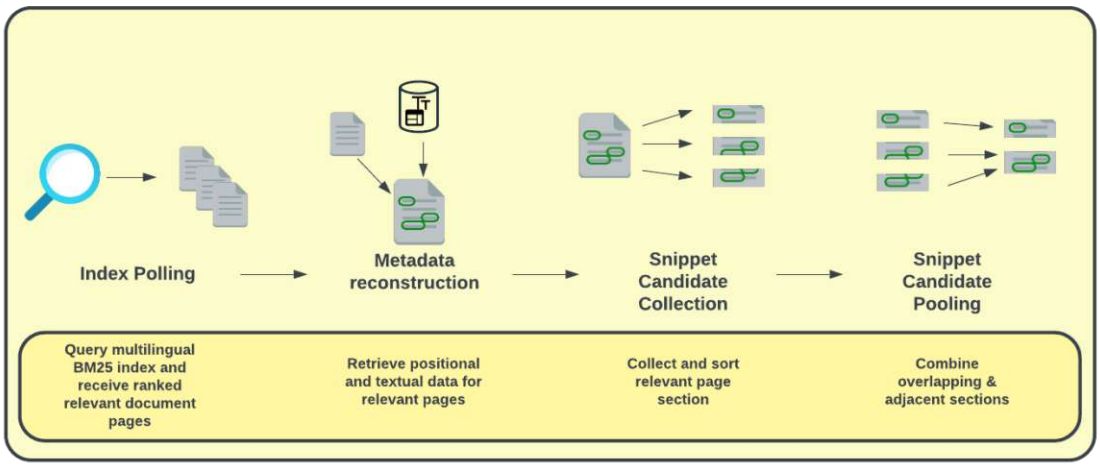


Figure 6.1: Steps in the snippet generation process

6.1 Preliminary Measures

The initial two steps described in this section are intended to gather and prepare the data required for the snippet generation.

6.1.1 Index Polling

The process of creating page snippets on the fly starts by launching a query at our previously introduced search index. We briefly outline our querying and ranking process in Section 5.1 and provide a description of our index in Appendix A. Since our index is structured in a way to catalogue individual pages of each recorded facsimile document, a search query results in a ranked paginated list of relevant document pages. While the index stores the text of the document page and additional information for the snippet generation, we use the ranked pages as a starting point to further retrieve page-specific OCR metadata that is extracted during the document import phase. Even though our chosen search engine, vespa[1], features the ability to return highlighted sections in the retrieved documents, we do not leverage this feature for our prototype. We opt not to utilize or adapt such built-in functions in favour of implementing our own highlighting solution based on the translation and synonym query metadata provided in the search result.

6.1.2 Metadata Reconstruction

Based on the search result from our index, we want to construct snippet-based result entries for each relevant document page. As previously noted in Section 4.2, we extract OCR metadata during the facsimile PDF import process that contains bounding boxes

[1] https://vespa.ai/

which surround recognized words. We create an inverted mapping that groups all bounding box positions based on word stems present on the given page. These mappings and also the scaled down page images are loaded for each retrieved page individually. Aside from the page-specific mappings, we have the query metadata at our disposal. The metadata contains all translated terms and found synonyms that were used in extending the initial search query. All these terms are stemmed and normalized according to their assigned language to easier find matching stems in the word boxes created during document import. Additionally, the mapped word boxes are re-linearized into an ordered list, again ordered from top to bottom and left to right based on their positions in the document page. In a later stage, this ordered list lets us generate word boxes that give users of our UI prototype the possibility to mark text across these boxes in the natural flow of the source text.

6.2 Candidate Collection

A single page can contain several relevant hits and therefore needs to be examined for every possible relevant word or phrase based on the initial search query. This requires iterating through the list of ordered words of each page and trying to gather all relevant words. Whether a word or phrase is relevant can be determined based on three criteria, either they can be directly matched to a query term or one of the translated terms, or the query page text contains synonyms of pre-defined domain-specific terms as explained in Section 5.2.2. Matching based on a query term is quite straightforward because all page words as well as the terms and translations of a query are available in their language-processed stemmed form and can be compared quite effortlessly.

Contrary to that, found synonyms can come in any of several supported languages, are potentially not fully normalized, or can contain several equivalent sub-phrases which need to be considered and matched as separate phrases. For instance, some phrases have the same synonyms linked to them and are grouped together in the synonym data source (e.g. 'university/academy/college'). Synonyms also do not need to be of equivalent length to their matched original phrases, meaning single words can be synonymized into phrases consisting of several words. Based on the synonym data structure, we do not have access to the definitive source language of the provided synonyms. Due to the short phrase length of the provided synonyms and translation inaccuracies that come with short translation sources, we do not attempt to determine the source language of the phrases and therefore do not apply any language-specific normalization before gathering matches in the relevant page texts. Given these circumstances, we conduct a best effort approach to find matches based on synonyms by processing sub-phrases within synonyms separately and comparing synonyms both against the normalized terms of a page, as well as the original page text.

Each word or full phrase that is matched in any of the previously described ways, is further processed. Here, our previously described inverted mapping from single terms to the respective bounding boxes comes into play. Using this mapping, we can quickly

retrieve all positions of relevant terms and consider these relevant boxes as snippet candidates. Even though such terms can be part of a longer relevant phrase, we treat each bounding box individually at first. This is because adjacent boxes that we view as neighbouring bounding boxes in a text sequence can have slightly different positions and might not even be on the same text line. Snippet candidate boxes start with an area surrounding the relevant word box. As a measure to provide uniform snippet sizes across a single page, the width of the candidate boxes are extended to the whole width of the surrounding page. The candidate boxes are padded with 3% of the overall page height above and below the relevant word box. Figure 6.2a demonstrates a document page excerpt after the initial candidate collection process, with all the candidate boxes (outlined with green borders) repeatedly overlapping.

6.3 Snippet Candidate Pooling

After our candidate collection process, we are faced with a list of horizontal page slices surrounding relevant words. Since all candidates are also padded and a single horizontal slice can represent a text line, which might contain several relevant words, individual relevant boxes can overlap or otherwise be very close to each other on the page. However, our snippet generation is intended to declutter the search results, therefore displaying a number of potentially overlapping or even duplicated sections would not contribute to the goal of having a clear result overview. Accordingly, we want to examine the candidate boxes including their paddings for overlapping sections and combine them into larger snippets, until we are left with a set of distinct boxes that do not overlap any more. By also considering the padding of the candidate boxes, we ensure that relevant word boxes, which are vertically very close to each other without overlapping, also are pooled together into a combined snippet. Through this measure, we prevent generating several adjacent snippets which on the original page are only separated by negligible amounts of vertical white space. Listing 6.1 describes an approach by iterating over all candidate boxes and finding overlapping or adjacent bounding boxes and combining them into a pooled box that surrounds several relevant terms. To visually demonstrate the candidate pooling process, Figure 6.2 shows a processed excerpt of an example document page based on the query *"combat activity during war"*. More specifically, Figure 6.2a displays the initial candidate boxes enveloping relevant terms (additionally highlighted in green) and the surrounding horizontal slices that are vertically padded and span the whole document width (outlined with green borders). Here we can clearly see one example, where multiple terms are located in the same line of the document and would therefore lead to a duplicate snippet. Additionally, we can see multiple cases in which the padded space around terms is overlapping or where adjacent padded candidate boxes are only separated by a small amount of vertical white space. If we employ a process as seen in Listing 6.1, the resulting pooled boxes for the same document and query can be seen in Figure 6.2b. We can see that the described process leads to a reduced number of snippet boxes and additionally removes any redundancy within the combined subset. Once we have eliminated any snippet overlaps, the created snippet dimensions are finally cropped

out of the rendered image of the respective document page. We can see the resulting cropped snippets for the previously described query example in Figure 6.3b, while Figure 6.3a shows the overlapping and redundant snippets that would result from a process without candidate pooling. Snippet results contain all word boxes within their bounds, with the relevant word boxes being highlighted accordingly. Due to the padding being relative to the page height and not based on individual line height, some word boxes that are partially visible within the visual snippet are cut off from the provided list of word bounding boxes.

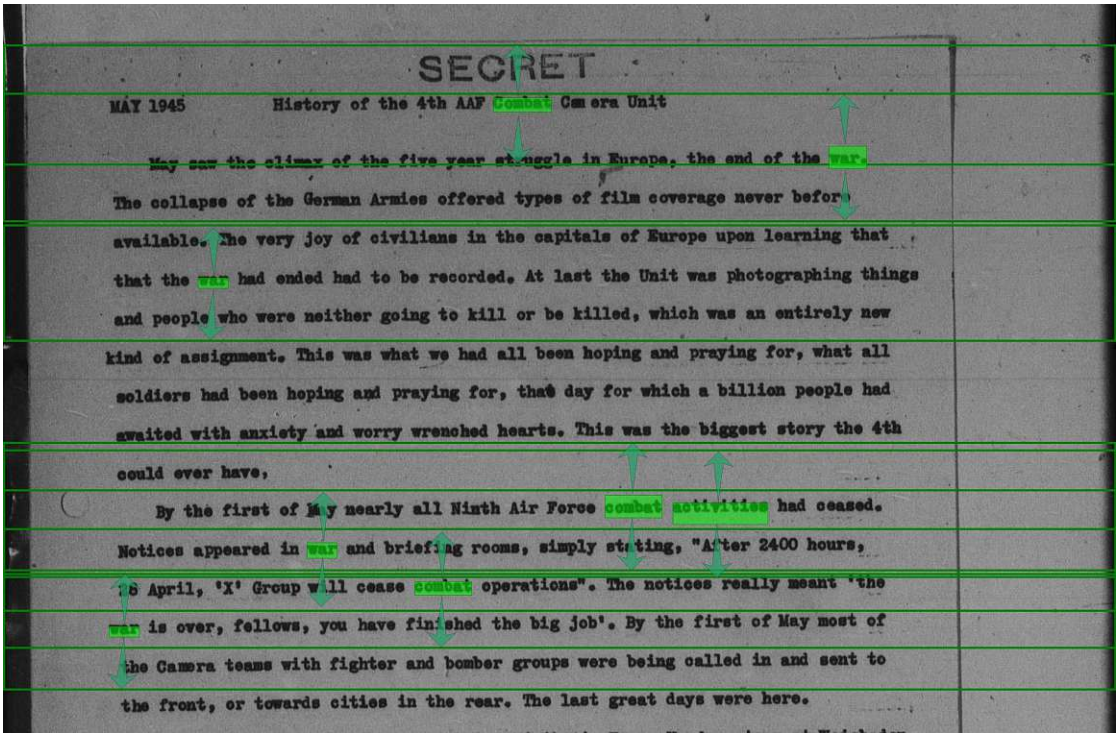
Listing 6.1: Pooling of overlapping or nearby candidate boxes

```
word_position_map = {}
checked_boxes = []
candidate_boxes.sort() #  sort based on ascending y-position
for candidate in candidate_boxes:
  for checked in checked_boxes:
    # check if boxes are overlapping
    # or within a lower_bound distance from each other
        if (checked.y0 <= candidate.yo <= checked.y1)    or
    (candidate.y0 <= checked.yo <= candidate.y1)         or
    (abs(candidate.y0 - checked.y1) <= lower_bound)      or
    (abs(checked.y0 - candidate.y1) <= lower_bound):
      # combine boxes that are overlapping or nearby
      combined = Box(
            candidate.x0,
            candidate.x1,
            min(candidate.y0, checked.y0),
            max(candidate.y1, checked.y1)
      )
          checked_boxes.pop(checked)
          checked_boxes.append(combined)
        else:
          checked_boxes.append(candidate)
```

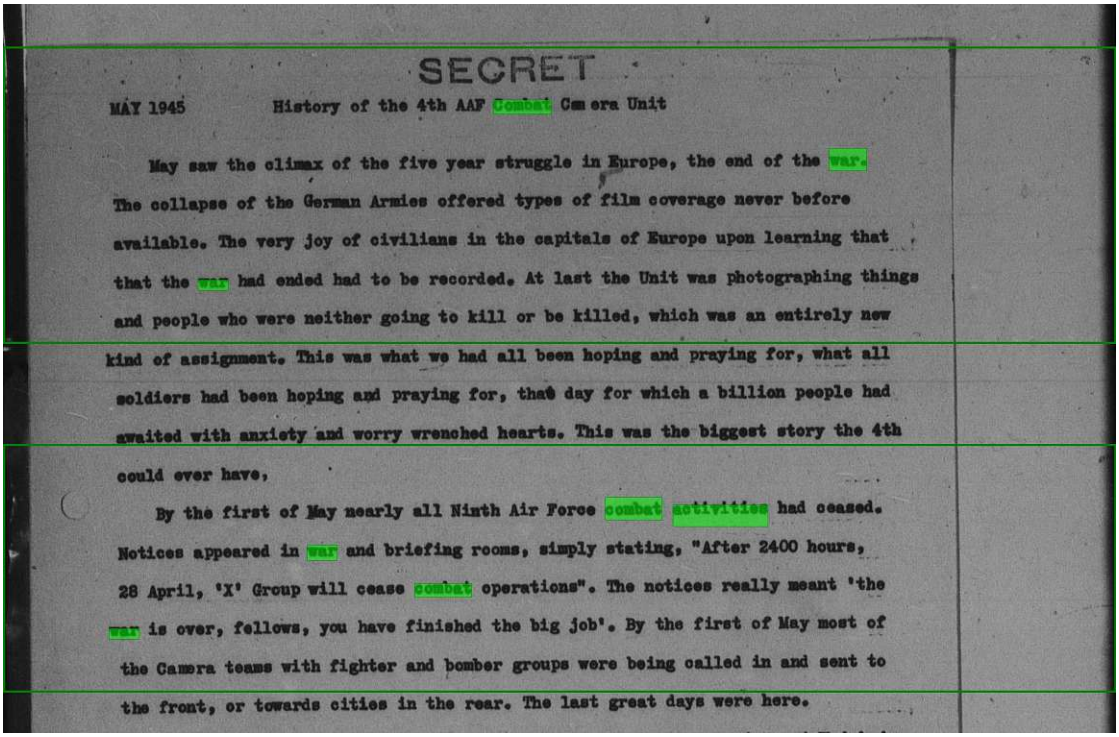
6.4 Performance Sampling

One concern when generating snippets from the source documents is the processing speed and the memory overhead during processing. As has been previously mentioned in Chapter 4, we create scaled down copies of the page images created from the source documents. Working with smaller images in cases where the quality downgrade can be tolerated grants us the possibility to improve the response times and generates less memory overhead while analysing and cropping the pages into snippets.

For this purpose, we created a test setup that allows us to monitor the isolated snippet

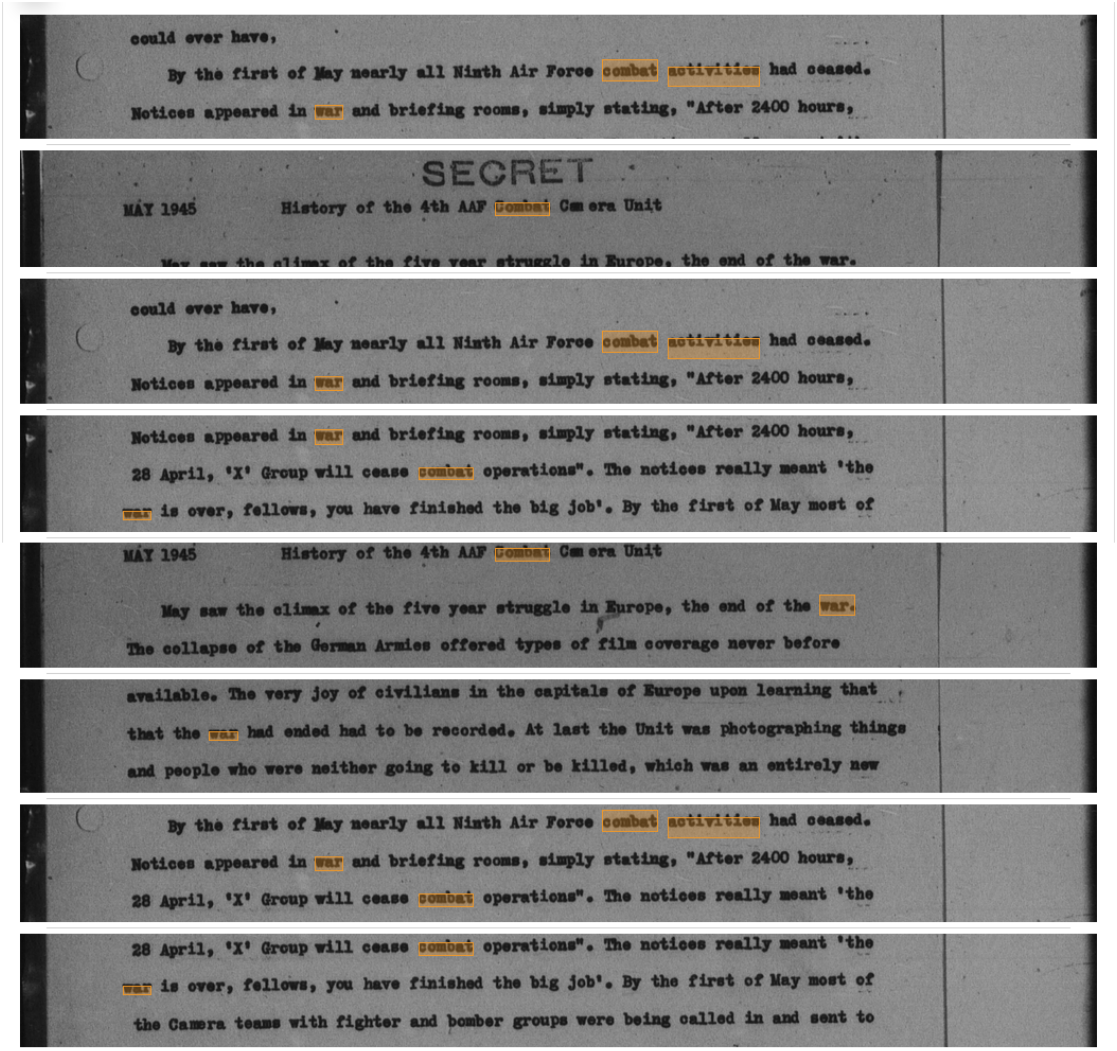


(a) Overlapping candidate boxes (green highlight) including padding (dark green border & arrows)

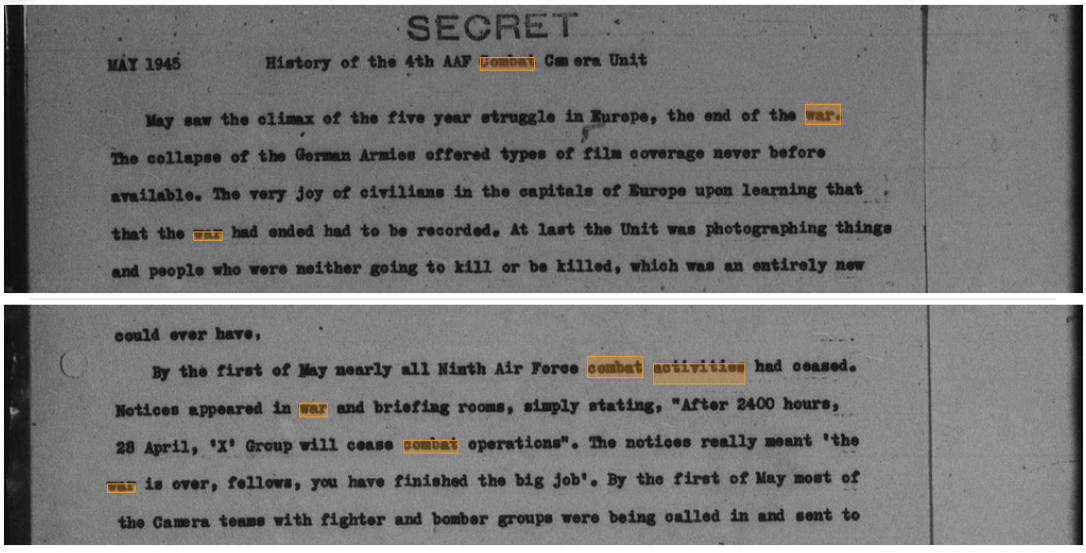


(b) Combined term boxes with surrounding padding after candidate pooling

Figure 6.2: Relevant term boxes with paddings before (a) and after (b) pooling for example query and document page



(a) Overlapping and redundant snippets



(b) Pooled snippets

Figure 6.3: Generated snippets for same query and document page without (a) and with (b) candidate pooling enabled

Table 6.1: Runtime and peak memory usage for snippet generation with different source resolution

sample	resolution	filesize	runtime (ms)	peak memory
A (original)	2672×4272	1.28 MB	98 – 155	107.4 MiB
A (downgrade)	1336×2135	335.6 kB	27 – 42	69.2 MiB
B (original)	3905×5007	2.48 MB	133 – 174	136.6 MiB
B (downgrade)	1952×2503	689.62 kB	30 – 71	76.5 MiB

generation process and fetches only a singular relevant document page for a given query. Within each test run, we can toggle between working with the original page in its original resolution or the downscaled version. This allows us to compare performance metrics between two versions of the same document page and query. Among the used tools to evaluate performance are the Werkzeug Application Profiler Middleware[2] to capture the overall endpoint response time and gain insight into runtime distribution, and filprofiler [3] to record peak memory usage. We evaluated the performance for a sample of two document pages out of the collection of documents that we had at our disposal during development and testing. These two pages are among a sub-collection of documents captured at a high resolution, and could therefore benefit greatly from being processed in a scaled down version for snippet generation. For each profiling tool and both versions of both sampled pages, we ran our isolated query setup for 10 iterations. Table 6.1 contains details for the inspected document pages and the metrics observed during testing.

6.4.1 Sample A

Comparing both versions of sample A, we are looking at an average response time reduction of 72.73% from 126.5 ms to 34.5 ms and a median reduction of 70.59% from 102 ms to 30 ms. Response time improvements for sample A range from 82.58% in the best-case scenario to 57.14% in the worst-case. The peak memory consumption could be reduced by 35.57% from 107.4 MiB to 69.2 MiB.

6.4.2 Sample B

Response times for Sample B could be reduced by 67.1% from 153.5 ms to 50.5 ms on average and by 76.17% from 138.5 ms to 33 ms when looking at the sample medians. The improvements range from 133 ms – 71 ms (46.62%) in the worst case to 174 ms – 30 ms (82.76%) in the best case. Peak memory consumption sampled across 10 iterations could be reduced from 136.6 MiB to 76.5 MiB (44%).

[2] https://werkzeug.palletsprojects.com/en/2.2.x/middleware/profiler/
[3] https://pythonspeed.com/fil/

6.4.3 Performance Summary

By halving both width and height for source images with high resolutions and therefore working with a fourth of the source resolution, we manage to cut down response times for snippet generation by up to 82% in ideal conditions and roughly 70% on average for our two sampled document pages. Additionally, peak memory consumption can be reduced by over a third in both cases. This can be achieved while also preserving a resolution that provides enough quality for the contents to be properly grasped in smaller document page surrogates in the search result. On the other side of the spectrum, source documents with low resolutions only are scaled down to a lower bound of 1500×1500, since they already inherently are being processed faster than documents captured in higher resolutions and should also not be downgraded beyond legibility.

CHAPTER 7

Conclusion

Given the recent improvements in text digitization and post-processing, which in turn leads to a higher throughput of digitized historic documents, researchers are investigating how to improve their research and exploration of archival documents in a digital environment. Together with researchers of the Visual History of the Holocaust project, we discussed the requirements for a research workflow that involves the use of digitized facsimile documents in an initial meeting. After analysing the received input from the initial meeting, we fleshed out specific requirements to adhere to throughout our work. By involving the researchers in a recurring feedback loop during our iterative design and development process, we ensured that the initial high-level requirements were properly applied and adjusted in our resulting technical implementation. As a consequence, we proposed several tools and components that together aim to satisfy the established requirements, which were described extensively in Chapter 2.

7.1 Requirements Review

One category, namely input processing, covered the requirements of preserving digitized archival facsimile quality, providing language support and a source-agnostic interface. Instead of using an input format of a specific digitization tool, our input processing workflow supports all text-annotated PDF files, which allows imports regardless of the source tool and also to optionally support digital-born documents in addition to digitized archival documents. Processed documents are preserved and stored in their original quality and in a downgraded version to allow better performance for our dynamic snippet generation. Additionally, extracted text inputs and OCR-based layout data are processed and indexed for later search in 10 different languages.

Another discussed and agreed-upon important aspect was the overall search experience provided by our solution. One core demand from the historians in that category was to make the research workflow as intuitive and fluent as possible, while trying to replicate

the experience of exploring topics in an archive with several documents at hand by using digital facsimile as an interactive starting point for exploration. By adhering to these requirements, we implemented a search user interface prototype. Searches launched in our prototype are passed through a query extension mechanism that translates search requests into 10 languages and additionally enriches the query with domain-specific synonyms of search terms. Our prototype uses dynamically generated and auto-highlighted image snippets of facsimile documents, which are relevant to the search. These snippets and also full-page views of documents are fully interactive via the positional metadata extracted during the input processing. This means that all facsimile representations in the prototype can be manually marked to create excerpts and also be used to start new searches or refine searches to dive deeper into a topic. Users can also quickly navigate between past searches to change topics. Phrases that are automatically highlighted provide tooltips to show the linguistic base (stem) form of a relevant word, and info-boxes show the translations and found synonyms of a search task to provide further explanation to the found results and highlighted phrases.

Each component within our solution stands on its own, and we have also been asked by VHH project partners at some point in our recurring feedback loop to provide a self-contained UI-independent version of our solution to allow leveraging our process for use in other user-facing tools. The feedback rounds and internal testing rounds have also yielded some performance observations that we were able to address along the development process. Among other small improvements, the facsimile scaling ratio for dynamic snippet generation has been adjusted for the best tradeoff between quality and response times and the generation process modified to be rendered on the server-side. Additionally, the input processing was adjusted to load and process document pages one by one to prevent memory spikes for single-machine setups.

7.2 Limitations & Future Work

The form of input data tokenization and the defaulting to parsing PDFs from left to right described in Chapter 4, of course, relies very heavily on input text in languages, where words can easily be determined and separated based on whitespace placement and are not read from right to left. As our work is heavily focused on the visualization aspect of the proposed concept and only represents an initial prototype aimed at inspiring further research, we leave specific tokenization refinements for advancements in language support in the import process up to future work.

Moreover, our input processing has some additional drawbacks when dealing with other specific layout circumstances. Hyphenated words that span multiple lines from an OCR standpoint are treated as separate entities, and can therefore sometimes lead to cases where words are not properly normalized and fed to the index. Multi-column layouts, such as newspaper columns or scans of book spreads with both the left and right page visible, would require additional layout analysis steps to prevent grouping such columns together logically and interpreting them as single cohesive lines.

While we managed to satisfy most of the established requirements with our UI prototype, there are still some open challenges that we suggest future contributors to pursue. Our query translation mechanism picks the best match out of several translation suggestions for each query term. Given the complexity and inconsistencies of natural languages and the fact that we translate word by word in a phrase, we cannot ensure that the translation will always capture the full context of the source query. Therefore, we propose to enable the option to let users alter translated query terms on the go. In Section 2.3.1 we even go a step further and suggest exposing the native vespa YQL[1] query and letting expert users modify the query parameters on the lowest level. Our prototype would also benefit from additional performance improvements such as more efficient result item and list caching and the implementation of some UI shortcuts to improve overall research fluidity.

As we have mentioned in Section A.2.2, we have laid the foundation for a more experimental retrieval method involving the representation of words in the form of neurally trained word embeddings in vector space. These word embeddings and the relations between different embeddings in vector space allow for a more context-based approach to information retrieval than conventional word stem matches.

In the form of BERT, Devlin et al. [32] introduce an approach for transformer-based language representation pre-training. BERT manages powerful bidirectional context through a masked language model and next sentence prediction in pre-training. The model's very task-specific approach to fine-tuning allows BERT to be at the forefront of tasks in the areas of natural language processing and, consequently, information retrieval. Multilingual BERT (M-BERT)[2] is a modified version of traditional BERT pre-trained on Wikipedia in over 100 languages, without any language-specific identifiers, but still performs well with translated as well as zero-shot fine-tuning approaches. For future research building on our contributions, we suggest to use Multilingual BERT as a starting point to incorporate neural context-based re-ranking.

[1] https://docs.vespa.ai/en/reference/query-language-reference.html
[2] https://github.com/google-research/bert

APPENDIX A

Implementation Details

A.1 Architecture

Figure A.1 shows a visual overview of the ArchAIvist prototype project architecture. Users solely interact with the platform through our Angular-based search frontend (5). This module forwards user queries to the backend and visualizes the resulting relevant facsimile documents. All the following backend modules are docker containers orchestrated in a docker-compose configuration (1). The processing service, a lightweight Flask Python API, (2) parses the incoming user query for our vespa search index (3) and is responsible for creating query-relevant snippets from the facsimile document images. An additional endpoint processes import data for the search index while also extracting image representations and OCR data for query-time retrieval. Our search index application (3) is built on top of the scalable and extensible vespa engine. Customized search components enrich the query terms with translations into multiple languages and by matching semantically related phrases through a synonym map. Our last backend container serves as an API for single-term translations through the word2word[1] Python library, built on top of a Flask Python web service (4).

A.2 Indexing

The document indexing process is made up of several steps. This process can be set into operation by either feeding input documents (in PDF format) via an API endpoint or an internal import script. Subsequently, all information needed for later overlaying (highlighted) text over relevant snippets and full document pages is extracted and stored. Finally, the document text is fed into our search engine of choice, vespa[2]. Vespa is a

[1] https://github.com/kakaobrain/word2word
[2] https://vespa.ai/

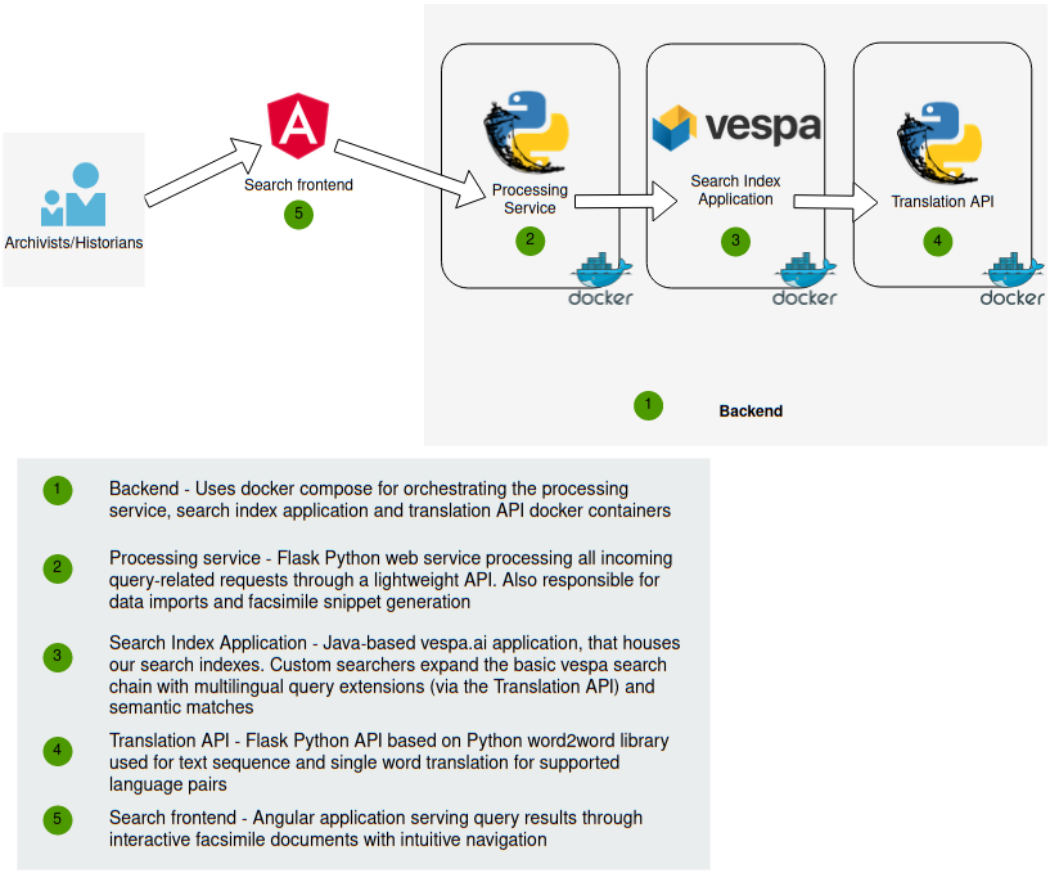


Figure A.1: Architectural overview

flexible tool with well-documented support for neural language models, and therefore suits both a traditional approach with an inverted index and more complex natural language processing (NLP) approaches potentially employed in future work.

A.2.1 Text Overlay Preparation

The following steps work for PDF files with existing OCR information, but can be adjusted to different input configurations. The input documents are split up into their individual pages to then be converted and stored as image files via the pdf2image[3] library. These images are important for the frontend visualization, as they act as a background for the OCR overlay and are the basis for the snippet generation. Additionally, we use PDFMiner[4] to extract terms and their bounding box positions for each individual document page. A language detect method is applied on the whole page corpus to reduce the words to their proper linguistic word stem. Next, the bounding boxes are stored in

[3] https://github.com/Belval/pdf2image
[4] https://github.com/pdfminer/pdfminer.six

an inverse index style mapping from stemmed document terms to bounding boxes on the document page. As a result, we can later on quickly retrieve all positions of relevant terms on a page or in a snippet.

A.2.2 Baseline Index & Retrieval

The baseline index and the resulting retrieval method serve as a strong initial basis for users looking for more conventional results based on term matches. Moreover, as the name suggests, it can also be used as a performance baseline for evaluating a more experimental concept based on neural dense retrieval that we propose and discuss in Section 7.2. Vespa provides a very hands-off approach for automatically building a basic inverted index for bag-of-words style retrieval, such as BM25. Part of the vespa configuration process is defining a schema for the processed documents. Our schema is intentionally kept simple and represents a single document page:

- **id**
- **language:** allows for language-based filtering
- **parent_doc:** document title reference to group individual pages
- **page:** incremental page numbers inside documents
- **body:** fully extracted text content of a document page
- **collection:** semantically group different documents into separate collections

Automatic indexing of such fields can be enabled through the 'indexing' attribute. Of these fields, currently only the 'body' field is indexed for BM25 retrieval. BM25 is a ranking algorithm which scores documents based on the frequency of occurrences of query terms (= term frequency), but favours rare terms through an inverse document frequency (IDF) calculated across the whole document corpus.

Since we are dealing with a multilingual document corpus, we must account for language discrepancies between the query language and document languages. To this end, we have implemented a custom searcher component in our vespa application that extends the original incoming query thanks to the translation API mentioned in the architecture section. When processing a query in our backend, our custom searcher tries to detect the source language and translates all query terms into all other supported document corpus languages. We can therefore use this extended multilingual query to allow for a more accurate ranking of document pages in other languages. In addition to translating all query terms, they are matched against a synonym data structure that maps domain-specific base terms to synonyms in different languages. The custom searcher returns a result very similar to the default vespa JSON result format[5], which additionally contains translations and synonym metadata collected during query extension.

[5] https://docs.vespa.ai/en/reference/default-result-format

APPENDIX B

User Interface Prototype

We present our user experience concept and the resulting prototype based on a typical user workflow: Our user wants to delve into a specific topic from the historic documents that were previously digitized and handled by our import process. After typing in a search phrase, the application starts working away in the background and promptly presents a list of results. An example result list can be seen in Figure B.1. Each item in this list represents a document page from the archive that matches the given query – either through exact term overlaps or synonym phrases across various languages. The results inform the user, which document the relevant pages are from and show cropped snippets from a digitized facsimile representation of the page with relevant passages already highlighted. After clicking on one of the result items, the user is now confronted with a high-resolution facsimile view of the full page with all relevant text passages already visually highlighted B.2. The page detail view contains a shortcut to all other relevant pages from the parent document, and also navigation buttons to leaf through the document page by page. Any potentially interesting text passage can be marked (via an invisible overlay of OCR-extracted plain text placed on top of the facsimile image). After marking a text section, a context tooltip is displayed on top of the selection, which lets users copy, search, or filter the marked text passage (Figure B.3). Like the initial query from the beginning of our example scenario, our prototype takes this passage, and launches a new search or filters the existing search based on the marked passage. The difference between searching or filtering based on a manually marked passage are the two different query modes explained in Chapter 5. Where the tooltip 'search' launches a new query in mode A, the 'filter' option launches a query in mode B that refines the current query results based on the marked passage. As previously mentioned, ArchAIvist both looks for exact or translated term matches across languages and tries to find matches in a pre-compiled selection of multilingual synonyms and again returns a new result list of relevant document snippets. These snippets, like the full facsimile pages, contain fully selectable text that can again lead to the contextual tooltip, which allows new searches

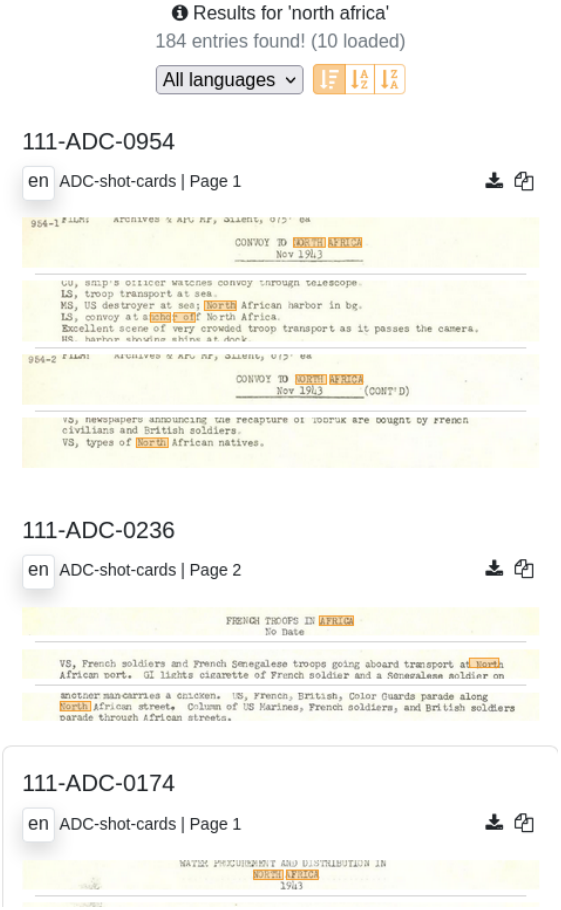


Figure B.1: Search result list containing relevant pages with facsimile snippets

and refinement. All search result lists can be individually filtered via the initially detected document language and sorted based on their relevance ranking or document name. The prototype's search history, lets users trace back past searches and explored documents, and users can also swiftly scroll horizontally through their history at any time. The text contents of the dynamically generated snippets can be copied and saved elsewhere and each search result also lets users download the original PDF document that was imported. Additionally, the application features an 'explore' mode, which shows several searches or document pages side-by-side, or 'focus' mode where each search result or detail view is scaled to the available screen width (Figure B.4).

Finally, we will revisit parts of the described user workflow to summarize the inner workings of the query process in combination with the UI prototype in light of all explained processes from previous chapters and the implementation details highlighted in Appendix A. Once the user has typed in a query, a request to our processing service gets launched. This service serves as a slimmed down intermediary to the very comprehensive

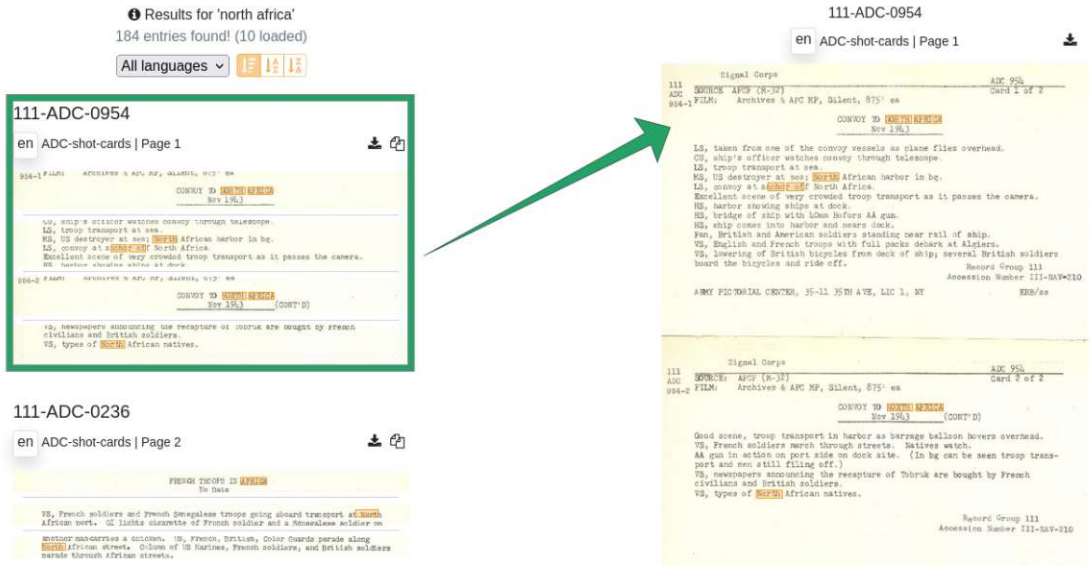
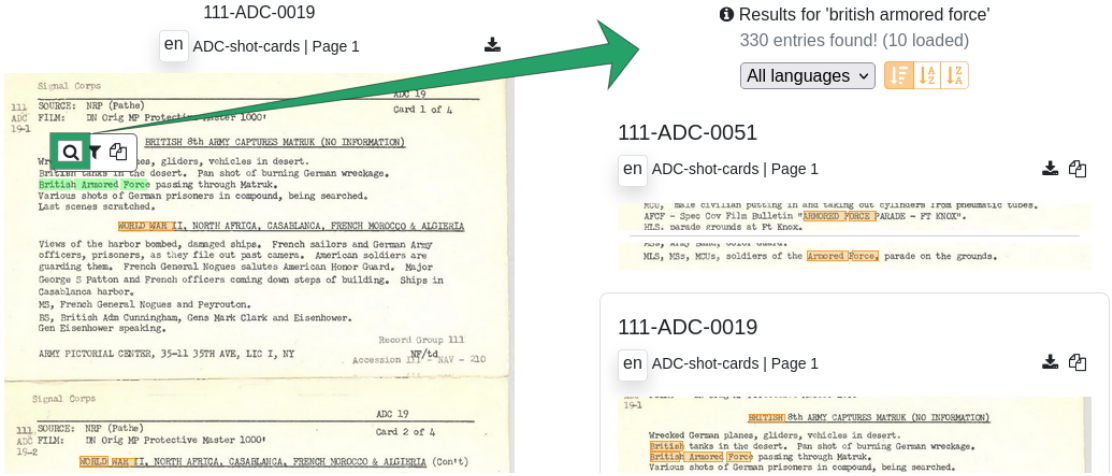
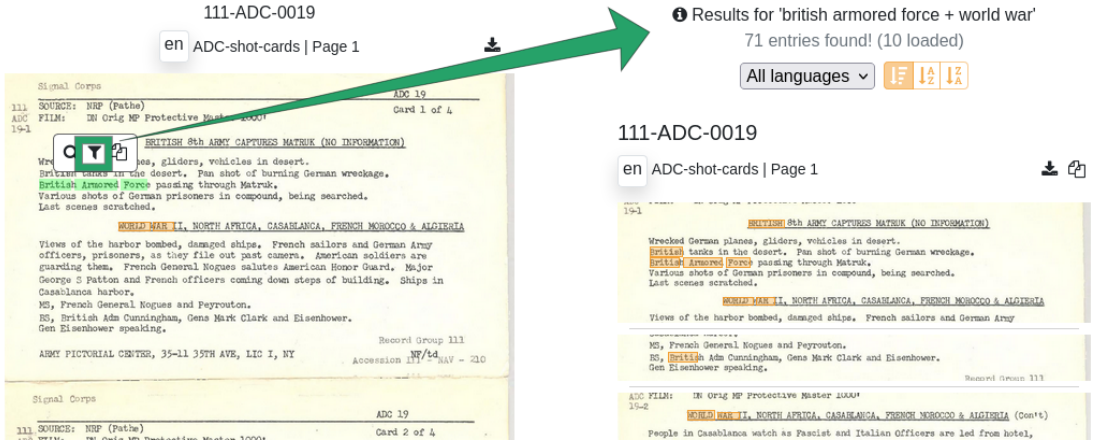


Figure B.2: Clicking a specific page in the result list opens a detail view of the full page side-by-side

API vespa applications provide out of the box. The parsed request gets forwarded to our multilingual search chain. As previously touched on, our custom searcher builds an extended multilingual query from the search terms and also includes matching synonym phrases. This extended query and the ranked relevant document pages are then returned to our intermediary service API. With the inverted index of stemmed query terms to positional bounding boxes, we can quickly locate the exact positions of relevant terms on a document page and cut out snippets around these terms. The service API then returns all retrieved information, including IDs for the server-side generated snippets in the response and relevant bounding box data, to the frontend. From here on, the frontend can start with visualizing the query results. Relevant document pages are displayed in a vertical list with an infinite scroll feature, incrementally loading the latest items when the bottom is reached. The result items are cards containing the document name, the page number and a collection of query-relevant interactive snippets. Clicking on one of the result cards lets the user delve deeper into a single document or page on a new horizontal search beam next to the previous beams. Every search or document detail exploration adds on to the chain of search history items, that can always be examined on the left, and lets users quickly horizontally navigate to previous searches or documents (Figure B.5). Document detail beams display a full image of the facsimile document page for more thorough examinations. Moreover, these detail views of single pages can serve as a stepping stone to extend the user's search chain. To further encourage archive exploration, our angular application renders an invisible overlay of term bounding boxes on top of the image snippets and full-page images. These bounding boxes show a tooltip with the characters recognized by OCR at the position when hovering over them (Figure B.6). For instance, this can be useful to sift through hardly legible text passages, or

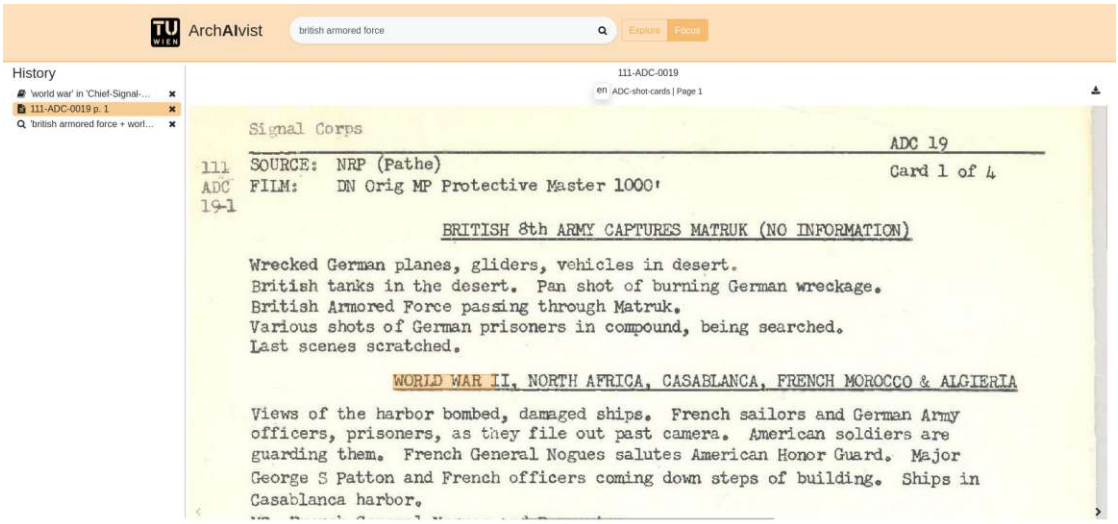


(a) Launch new search based on the marked text passage

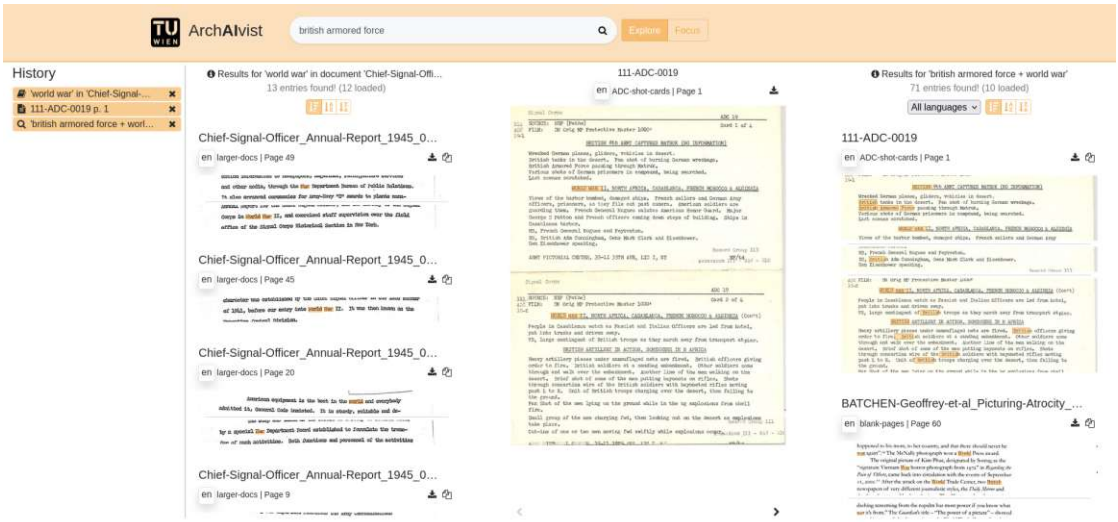


(b) Refine the previous search based on the marked text passage

Figure B.3: Continue searching or refine searches through a context tooltip after manually marking text sections



(a) Focus – Scales the contents to the fully available width to investigate single results or documents



(b) Explore – Gives an overview of the past searches and documents to promote extending the search chain

Figure B.4: Comparison of the two layout modes

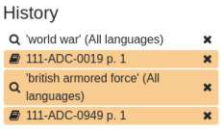


Figure B.5: Interactive history results

Figure B.6: OCR box overlay text tooltip

simply to detect OCR misses and errors. Additionally, these bounding boxes enable the user to mark text passages directly in the facsimile renderings. Our application also visually highlights term boxes that are relevant to the current search. Upon marking a text passage, a tooltip opens positioned just above the passage, which enables the user to either start a new search query with the text selection or copy the text to the clipboard.

List of Figures

Bibliography

[1] S. Oberbichler, E. Boroş, A. Doucet, J. Marjanen, E. Pfanzelter, J. Rautiainen, H. Toivonen, and M. Tolonen, "Integrated interdisciplinary workflows for research on historical newspapers: Perspectives from humanities scholars, computer scientists, and librarians," *Journal of the Association for Information Science and Technology*, vol. 73, no. 2, pp. 225–239, 2022. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/asi.24565.

[2] D. Force and B. Wiles, ""Quietly Incomplete": Academic Historians, Digital Archival Collections, and Historical Research in the Web Era," *Journal of Contemporary Archival Studies*, vol. 8, Dec. 2021.

[3] A. Hawkins, "Archives, linked data and the digital humanities: increasing access to digitised and born-digital archives via the semantic web," *Arch Sci*, Dec. 2021.

[4] E. Late and S. Kumpulainen, "Interacting with digitised historical newspapers: understanding the use of digital surrogates as primary sources," *JD*, vol. 78, pp. 106–124, Sept. 2021.

[5] P. Tranouez, S. Nicolas, V. Dovgalecs, A. Burnett, L. Heutte, Y. Liang, R. Guest, and M. Fairhurst, "DocExplore: overcoming cultural and physical barriers to access ancient documents," in *Proceedings of the 2012 ACM symposium on Document engineering*, DocEng '12, (New York, NY, USA), pp. 205–208, Association for Computing Machinery, Sept. 2012.

[6] M. Marx and T. Gielissen, "Digital weight watching: reconstruction of scanned documents," *IJDAR*, vol. 14, pp. 229–239, June 2011.

[7] S. Colutto, P. Kahle, H. Guenter, and G. Muehlberger, "Transkribus. A Platform for Automated Text Recognition and Searching of Historical Documents," in *2019 15th International Conference on eScience (eScience)*, pp. 463–466, Sept. 2019.

[8] T. Wilkinson, J. Lindström, and A. Brun, "Neural Word Search in Historical Manuscript Collections," Tech. Rep. arXiv:1812.02771, arXiv, Mar. 2020. arXiv:1812.02771 [cs] type: article.

[9] C. Neudecker, K. Baierer, M. Federbusch, M. Boenig, K.-M. Würzner, V. Hartmann, and E. Herrmann, "OCR-D: An end-to-end open source OCR framework for historical printed documents," in *Proceedings of the 3rd International Conference on Digital Access to Textual Cultural Heritage*, (Brussels Belgium), pp. 53–58, ACM, May 2019.

[10] M. Hämäläinen and S. Hengchen, "From the Paft to the Fiiture: a Fully Automatic NMT and Word Embeddings Method for OCR Post-Correction," in *Proceedings - Natural Language Processing in a Deep Learning World*, pp. 431–436, Oct. 2019. arXiv:1910.05535 [cs].

[11] R. Therón, C. Seguín, L. de la Cruz, and M. Vaquero, "Highly interactive and natural user interfaces: enabling visual analysis in historical lexicography," in *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*, DATeCH '14, (New York, NY, USA), pp. 153–158, Association for Computing Machinery, May 2014.

[12] X. Chen, W. Zeng, Y. Lin, H. M. AI-maneea, J. Roberts, and R. Chang, "Composition and Configuration Patterns in Multiple-View Visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, pp. 1514–1524, Feb. 2021. Conference Name: IEEE Transactions on Visualization and Computer Graphics.

[13] F. Windhager, P. Federico, G. Schreder, K. Glinka, M. Dörk, S. Miksch, and E. Mayr, "Visualization of Cultural Heritage Collection Data: State of the Art and Future Challenges," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, pp. 2311–2330, June 2019. Conference Name: IEEE Transactions on Visualization and Computer Graphics.

[14] O. Hoeber, "Information Visualization for Interactive Information Retrieval," in *Proceedings of the 2018 Conference on Human Information Interaction & Retrieval*, CHIIR '18, (New York, NY, USA), pp. 371–374, Association for Computing Machinery, Mar. 2018.

[15] O. Hoeber and X. D. Yang, "HotMap: Supporting visual exploration of Web search results," *Journal of the American Society for Information Science and Technology*, vol. 60, no. 1, pp. 90–110, 2009. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/asi.20957.

[16] T. Khazaei and O. Hoeber, "Supporting academic search tasks through citation visualization and exploration," *Int J Digit Libr*, vol. 18, pp. 59–72, Mar. 2017.

[17] S. Shukla and O. Hoeber, "Visually Linked Keywords to Support Exploratory Browsing," in *Proceedings of the 2021 Conference on Human Information Interaction and Retrieval*, CHIIR '21, (New York, NY, USA), pp. 273–277, Association for Computing Machinery, Mar. 2021.

[18] C. di Sciascio, V. Sabol, and E. E. Veas, "Rank As You Go: User-Driven Exploration of Search Results," in *Proceedings of the 21st International Conference on Intelligent*

User Interfaces, IUI '16, (New York, NY, USA), pp. 118–129, Association for Computing Machinery, Mar. 2016.

[19] P. Martin-Rodilla and M. Sánchez, "Software Support for Discourse-Based Textual Information Analysis: A Systematic Literature Review and Software Guidelines in Practice," *Information*, vol. 11, p. 256, May 2020.

[20] M. Ehrmann, E. Bunout, and M. During, "Historical Newspaper User Interfaces: A Review," 2017.

[21] D. Hebert, T. Palfray, S. Nicolas, P. Tranouez, and T. Paquet, "PIVAJ: displaying and augmenting digitized newspapers on the web experimental feedback from the "Journal de Rouen" collection," in *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*, DATeCH '14, (New York, NY, USA), pp. 173–178, Association for Computing Machinery, May 2014.

[22] D. Hebert, T. Palfray, S. Nicolas, P. Tranouez, and T. Paquet, "Automatic article extraction in old newspapers digitized collections," in *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage - DATeCH '14*, (Madrid, Spain), pp. 3–8, ACM Press, 2014.

[23] K. Kettunen, T. Pääkkönen, and E. Liukkonen, "Clipping the Page – Automatic Article Detection and Marking Software in Production of Newspaper Clippings of a Digitized Historical Journalistic Collection," in *Digital Libraries for Open Knowledge* (A. Doucet, A. Isaac, K. Golub, T. Aalberg, and A. Jatowt, eds.), Lecture Notes in Computer Science, (Cham), pp. 356–360, Springer International Publishing, 2019.

[24] A. Doucet, M. Gasteiner, M. Granroth-Wilding, M. Kaiser, M. Kaukonen, R. Labahn, J.-P. Moreux, G. Muehlberger, E. Pfanzelter, M.- Thérenty, H. Toivonen, and M. Tolonen, "NewsEye: A digital investigator for historical newspapers," in *15th Annual International Conference of the Alliance of Digital Humanities Organizations, DH 2020*, (Ottawa, Canada), July 2020.

[25] A. Jean-Caurant and A. Doucet, "Accessing and Investigating Large Collections of Historical Newspapers with the NewsEye Platform," in *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*, (Virtual Event China), pp. 531–532, ACM, Aug. 2020.

[26] M. Caserio, A. Goy, and D. Magro, "Smart Access to Historical Archives based on Rich Semantic Metadata:," in *Proceedings of the 9th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*, (Funchal, Madeira, Portugal), pp. 93–100, SCITEPRESS - Science and Technology Publications, 2017.

[27] N. Gutehrlé, O. Harlamov, F. Karimi, H. Wei, A. Jean-Caurant, and L. Pivovarova, "SpaceWars: A Web Interface for Exploring the Spatio-temporal Dimensions of WWI

Newspaper Reporting," *CEUR Workshop Proceedings*, Oct. 2021. Publisher: CEUR Workshop Proceedings.

[28] P. Chu and A. Komlodi, "TranSearch: A Multilingual Search User Interface Accommodating User Interaction and Preference," in *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '17, (New York, NY, USA), pp. 2466–2472, Association for Computing Machinery, May 2017.

[29] B. Steichen and L. Freund, "Supporting the Modern Polyglot: A Comparison of Multilingual Search Interfaces," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, (New York, NY, USA), pp. 3483–3492, Association for Computing Machinery, Apr. 2015.

[30] C. Ling, B. Steichen, and A. G. Choulos, "A Comparative User Study of Interactive Multilingual Search Interfaces," in *Proceedings of the 2018 Conference on Human Information Interaction & Retrieval*, CHIIR '18, (New York, NY, USA), pp. 211–220, Association for Computing Machinery, Mar. 2018.

[31] Y. J. Choe, K. Park, and D. Kim, "word2word: A collection of bilingual lexicons for 3, 564 language pairs," *CoRR*, vol. abs/1911.12019, 2019.

[32] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv:1810.04805 [cs]*, May 2019. arXiv: 1810.04805.