Institut für Maschinelle Sprachverarbeitung

Universität Stuttgart

Pfaffenwaldring 5B

D-70569 Stuttgart

Master Thesis

Multimodal OCR Post-Correction on German Historical Documents

Nianheng Wu

Studiengang:	M.Sc. Computational Linguistics
Prüfer*innen:	Prof. Dr.Sebastian Padó
	Prof. Dr. Roman Klinger
Betreuer:	Prof. Dr.Sebastian Padó
Beginn der Arbeit:	28.03.2023
Ende der Arbeit:	28.09.2023

Erklärung (Statement of Authorship)

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst habe und dabei keine andere als die angegebene Literatur verwendet habe. Alle Zitate und sinngemäßen Entlehnungen sind als solche unter genauer Angabe der Quelle gekennzeichnet. Die eingereichte Arbeit ist weder vollständig noch in wesentlichen Teilen Gegenstand eines anderen Prüfungsverfahrens gewesen. Sie ist weder vollständig noch in Teilen bereits veröffentlicht. Die beigefügte elektronische Version stimmt mit dem Druckexemplar überein.[1]

(Nianheng Wu)

[1]Non-binding translation for convenience: This thesis is the result of my own independent work, and any material from work of others which is used either verbatim or indirectly in the text is credited to the author including details about the exact source in the text. This work has not been part of any other previous examination, neither completely nor in parts. It has neither completely nor partially been published before. The submitted electronic version is identical to this print version.

List of Figures

List of Tables

Contents

1 Introduction

Digitization of historical documents (such as magazines, academic papers, and newspapers) is crucial to today's humanities research. On the one hand, digitized versions of documents would not be limited by their physical conditions and can be easily accessed by more researchers worldwide. On the other hand, as more quantization and analysis methods in mathematics and computer science are being adopted in humanity fields, digitized archives that are directly readable by computers have become the foundation of this process (Constantopoulos et al., 2002).

One of the most essential steps in digitization is Optical Character Recognition (OCR). This step transfers documents in image form into text form. The quality of OCR is of vital importance. Although the technology of OCR has improved a lot through the last decades, most off-the-shelf OCR systems are designed for general purposes (such as handwritten scripts and scene text). They are still not robust enough to handle the varieties of layouts, fonts, paper colors, and scan quality in historical document scans, as well as the different, ancient wordings and phrases in the contents (Mittal and Garg, 2020). Besides, the orthography of many languages, including German, has usually evolved across multiple centuries and has different features in every period. Therefore, an OCR system trained on corpora from one era could perform worse on datasets from other eras (Gloning and Young, 2004). Because of these reasons, researchers who need to digitize historical documents now sometimes need to manually transcribe them into text, which is very expensive and time-consuming.

One obvious solution is to train and improve an OCR system using those historical documents. It would take images of these documents as input and predict the text on the pages. This step would require a massive amount of data and computation resources. For reference, the SOTA OCR system TrOCR (Li et al., 2021) was trained on nearly 700M textlines and used 32 GPUs with a memory of 32 GBs for pre-training and 8 GPUs for fine-tuning. Such resources are far beyond reach in many research projects.

Therefore, an OCR post-correction system became the natural choice of many. It was introduced as the next step in the digitizing pipeline to denoise the OCR-ed text, increase transcription accuracy, and reduce manual effort for post-correction. A typical OCR post-correction system would take in noisy OCR-ed text and output the corrected ones. Most prior works treat OCR post-correction as a translation problem, and the use of visual aids remains unexplored here (See section 3.2 for details). These years, large-scale pretrained models have achieved astonishing results on various downstream tasks (Kalyan et al., 2021; Han et al., 2022), and brought new options and possibilities to this topic. In this work, I will present a multimodal OCR post-correcting system that 1) Works on German historical documents; 2) Utilises pretrained model ByT5's encoder and decoder (Xue et al., 2022); 3) Combines a CNN backbone from CLIP (Radford et al., 2021) for visual feature extraction; 4) Uses adapter(Houlsby et al., 2019) blocks to achieve efficient transfer learning and adapt the text-to-text model to multimodality. The goals of this work are:

1. **To explore the described system**;
2. To answer the question: **How much can the OCR post-correction system benefit from visual information as input?**

In the second chapter, I will review related works and give the readers a comprehensive research background. Various related technologies will be mentioned there, including OCR, OCR post-correction, transformers, multimodal learning, adapters, and an overview of German historical documents. Following that, the details of the dataset and the way of collecting and processing it will be explained in chapter three. In chapter four, the model structure will be presented. Then, the following chapter will elaborate on the experiments, including the main experiment with the new system presented in this work and two baseline models. The results will be shown in chapter 6, and finally, chapter 7 will summarize the conclusions.

2 Research Background

As mentioned before, to solve the problem of erroneous OCR-ed text, there are two obvious options: 1. Upgrading OCR systems, and 2. Utilizing OCR post-correction. Other than the fact that OCR systems are quite data-hungry and heavy on computation resources, the existing OCR systems are usually designed for more general purposes, meaning they are trained on datasets like modern newspapers, receipts, handwriting pictures, or scene text from real life. In contrast, OCR post-correction systems can use a lot more linguistic knowledge for the text we have. Customizing strategies is usually necessary for text in a specific domain or genre, and OCR post-correction allows us to do that. In the following paragraphs, I will briefly introduce the background of OCR and summarise previous approaches for OCR post-correction. Then, I will revisit the related technology, including CNN, transformers, and adapters. Lastly, I will describe the related background of German language history.

2.1 Optical Character Recognition

OCR is a vast field with several varied applications such as scene text recognition (Singh et al., 2021), handwriting recognition (Memon et al., 2020), documentation transcription (Neudecker et al., 2019). However, prior works rarely handle historical documents, which usually contain many fossil words, dialects, and terminologies. Besides, historical documents are often printed in unique fonts such as Fraktur or complicated layout (Riedl et al., 2019), making it more challenging for OCR systems. Until now, the state-of-the-art systems that can handle Gothic fonts in German are Tesseract (Kay, 2007) and commercial OCR system ABBYY FineReader, but their accuracy is still limited (Heliński et al., 2012).

Through the years, the quality of OCR systems has been improved drastically. Some pretrained OCR models that made use of encoder-decoder transformer structure (Vaswani et al., 2017), such as TrOCR (Li et al., 2021) and MaskOCR (Lyu et al., 2022) achieved state-of-the-art results on modern receipt images e.g. SROIE(Huang et al., 2019), handwriting datasets like IAM (Marti and Bunke, 2002), and scene text datasets. Both models were trained only on modern English and Chinese datasets using vision transformer (Dosovitskiy et al., 2020). Besides, the data-hungry nature of vision transformers means that one must train them on hundreds of millions of related German textlines to perform well on this specific task. This would require a long time and extensive computational resources – neither of which is possible in the case of this project, unfortunately.

2.2 OCR Post-Correction

In regards to text correction, there are mainly two approaches that are widely used for OCR post-correction: statistical and lexical.

Statistical approaches often estimate the distribution of errors inferred from training data. There are multiple ways for the estimation. In the Competition on Post-OCR Text Correction hosted by ICDAR 2019, a Spanish team applied weighted finite-state transducers to a noisy channel model. The system consists of an error model estimated from the training corpus, an n-gram language model derived from a dictionary, and a post-processing module to determine the best token sequence.(Rigaud et al., 2019). In Perez-Cortes et al. (2000), the authors combined stochastic finite-state automaton with the Viterbi Algorithm to perform Error-Correcting Parsing(ECP). Similar estimation methods are often used to solve machine translation problems. In Schulz and Kuhn (2017), the authors combined two statistical machine translation models with other lexical-level correction models to form a multi-modular OCR post-correction system. After neural networks became more popular, more projects were explored using neural machine translation models for OCR post-correction (Mokhtar et al., 2018; Hämäläinen and Hengchen, 2019; Schaefer and Neudecker, 2020). In Rigaud et al. (2019), Clova AI from South

Korea presented a BERT-based OCR post-correction system. It uses BERT as an error detector to find the character- and token-based errors. For correction, they applied a character-level sequence-to-sequence model with attention mechanism. In another research Lyu et al. (2021), the authors propose a neural approach based on a combination of recurrent (RNN) and deep convolutional networks (ConvNet) to correct OCR transcription errors. This method achieved state-of-the-art accuracy.

Lexical approaches usually involve a dictionary of correct words and a system that measures the distance between the erroneous words and their potential correct candidates. One solution is profiling. Reffle and Ringlstetter (2013) built a two-channel profile from OCR-ed historical text. The profile includes global information on typical recognition errors in the text and local information on particular tokens. As an extension to this work, Fink et al. (2017) added an adaptive feedback mechanism to support the system with adaptive human insights and new historical patterns. Some other works focus on the source of dictionaries. In Bassil and Alwani (2012), the authors used Google's online spelling check toolkit to give correct suggestions. Besides, Clematide et al. (2016) applied crowdsourcing to aggregate more comprehensive lexical resources.

Statistical approaches will be adopted in the case of this project for two reasons: 1. Neural networks have become the dominating approach in NLP; 2. The fact that there is a large proportion of irregular spellings in German historical documents rules out the option of using a lexical approach.

2.3 Transformers and Multimodal Learning

Transformer is a model structure that was proposed in Vaswani et al. (2017). It uses self-attention mechanism to model relationships in sequential data. This mechanism is very good at aligning and selectively focusing on relevant parts of the data, and it has already become one of the cornerstones of modern NLP technology. Almost all large-scale pretrained models nowadays are based on Transformer: BERT

(Devlin et al., 2018) takes the encoder part of the Transformer and is pretrained with Masked Language Model (MLM) and Next Sentence Prediction tasks. GPT (Radford et al. (2018), Radford et al. (2019)), on the other hand, takes only the decoder part of the Transformer and can produce sequence auto-regressively. Other than these, there are models like XLM (Lample and Conneau, 2019) that focus on cross-lingual ability, and RoBERTa(Liu et al., 2019) that aims to improve training techniques based on BERT.

To deal with the extensive vocabulary, these models use Byte-Pair-Encoding (BPE) (Sennrich et al., 2015) or its variations as the encoding method, which splits words into subword units. These subword-based models achieved outstanding performance on various tasks. However, word representation is very fragile: even minor typos or irregular spellings could drastically change the BPE tokens, leading to inaccurate representations. This problem would be particularly severe in the context of German historical documents, which contain many irregular spellings (See section 3.5). Targeting this problem, Boukkouri et al. (2020) proposed a character-based pre-trained model CharacterBERT. It dropped the BPE system altogether and used a Character-CNN module to represent the words. Similarly, ByT5(Xue et al., 2022) feeds UTF-8 bytes directly into the model without text preprocessing. Compared to CharacterBERT, which only supports English, ByT5 covers 100+ languages, including German.

Multimodal learning refers to the process of learning representations from different modalities with the same composed model. Vision language models are a type of multimodal model that deals with pictures and text at the same time. In recent years, pretrained vision language models have advanced at a breakneck pace and dominated the mainstream techniques in many vision-language (VL) tasks, e.g., visual question answering (VQA), visual captioning (VC), visual commonsense reasoning (VR). ViLBERT (Lu et al., 2019) is one of the first appeared visual language pre-trained models. It learns joint representation of images and text using a BERT-like structure with separate Transformers for vision and language that attend to each other. Following that work, VisualBERT (Li et al., 2019)used a single Transformer for both image and text input. These vision language models are pretrained with tasks such as masked region classification (MRC), cross-modal masked language modeling (MLM), and image-text matching (ITM). These tasks help the models to learn the alignment between descriptive text and image and focus on aligning the objects in scene images to the related text.

In the context of OCR post-correction, such pretraining could be of very limited use because most of those models were pretrained with scene pictures and content captions inferred from the scenes. Whereas OCR images contain text content, and the text is the literal content from the picture without any inference. Therefore, one can argue that the model is pretrained to have a different ability than what is required in OCR post-correction. In Radford et al. (2021), the authors proposed a contrastive training strategy and a multimodal model CLIP. Its visual encoder part is ResNet (He et al., 2016) or Visual Transformer (ViT) (Dosovitskiy et al., 2020). The model showed relatively good OCR ability in English. Therefore, I will use the pretrained visual encoder from CLIP in this project. CLIP encoder would only accept square image inputs, and the default cropping method for images that are not square is cutting it in the middle. It also comes with different input sizes ranging from 224px to 448px for the ResNet versions and 224px to 336px for the ViT versions. ResNet50x16 version will be chosen here for two reasons: 1. In Shen et al. (2021), the authors show that ViT struggles with localization; 2. To preserve as much information in the images as possible, I would like to choose a version of CLIP where the acceptable input size is big enough but would still fit into the limited available computation resources. ResNet 50x16 version accepts images of 384px x 384px, which became the best choice for this project.

2.4 Adapters

Adapter tuning (Houlsby et al., 2019) is a method for efficient transfer learning. This method inserts small neural modules called adapters into the inner layers of the pretrained model. Only the adapters will be trained during training time, and the other parameters in the model remain frozen. In contrast to fine-tuning, the most common way for transfer learning in NLP that involves adaptation on all parameters, adapter tuning is very efficient in regards to computation, storage, and memory resources while yielding competing results (Houlsby et al., 2019; He et al., 2021; Mahabadi et al., 2021). It is worth noticing that the total training time of a model using adapter tuning is typically 100%-150% of the training time using fine-tuning (He et al., 2021). Although the system trains faster for every epoch, because of having fewer trainable parameters, adapter tuning usually needs more epochs to converge. In recent years, adapters tuning was adopted and evaluated by multiple projects: in Pfeiffer et al. (2020b), adapters showed great ability in multilingual model adaptation and series of downstream tasks including Named Entity Recognition (NER), question answering (QA), and commonsense reasoning (CSR). In He et al. (2021), the authors experimented with different kinds of adapters and ways of using them, but this research only focused on text input. In Eichenberg et al. (2021), the author explored inserting adapters on top of the attention and feed-forward layers of a GPT-based decoder to adapt the text-based model and a pretrained ResNet module from CLIP into a multimodal system. They achieved relatively competitive results on a range of VL benchmarks. Sung et al. (2022) further explored combining pretrained text-to-text models and an image encoder with adapters and adapter variants for tasks like VQA, VC, VR, and so forth. The system achieved comparable results to fine-tuning.

There are some popular variants of adapters: The original bottleneck adapter (Houlsby et al., 2019); Hyperperformer (Mahabadi et al., 2021) that improves the efficiency of adapters by generating their weights via a hyper-network; Compacter

(Karimi Mahabadi et al., 2021) that reduces the parameters by using Kronecker products and low-rank parameterization; Besides, in He et al. (2021), the authors showed four different adapter structures depending on the way they modify head attention: parallel adapter, multi-head parallel adapter, scaled parallel adapter, and mix-and-match adapter. The mix-and-match adapter, which combines the favorable designs behind the three others, achieved the highest performance on summarization and machine translation tasks. All of them are available on *AdapterHub.ml* (Pfeiffer et al., 2020a).

2.5 German Text in Historical Documents

A popular periodization on the evolution of the German language divides its history into the following stages: Old High German (c. 750-c. 1050), Middle High German (c. 1050-c.1350), Early New High German (c.1350-c.1650), New High German (c.1650-c.1945) and Contemporary German (c. 1945-now) (Gloning and Young, 2004). This work will focus on the New High German era. During this time, the standardization of written German started to develop (Schottel, 1977), and there were no significant changes in phonology and morphology in this period (Besch and Wolf, 2009). Nevertheless, as it was not until the eighteenth century that the standardization of German was completed, in the early phase of the New High German era, one can observe more variants regarding grammar, spelling, and typesetting in historical documents (Langer, 2014). An example is shown in Fig.1: the first word was written as ”Jn” instead of the modern form ”In” while ”In” also exists in this book at the same time. This is because since Old High German time, /i/ and /j/ are inconsistently interchangeable in written language (Must, 1965). Besides, although the written form for umlaut with two dots on top (ä, ö, ü) was already introduced in Early New High German times, it was still not uncommon to see umlaut with a small e on top in prints (e.g., erkl$\overset{e}{a}$rt, today as *erklärt*. It means explain).

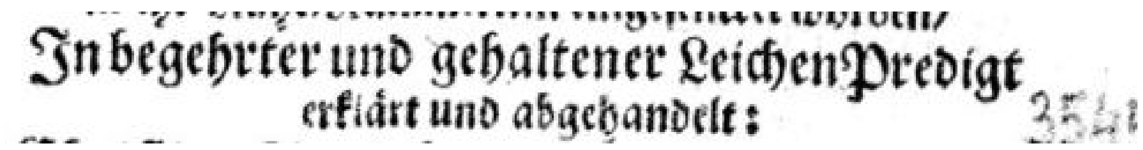

Jn begehrter und gehaltener LeichenPredigt
erklärt und abgehandelt:

Figure 1: "Jn begehrter und gehaltener LeichenPredigt erkl$\overset{e}{a}$rt und abgehandelt: The front page scan of *Adolph, Christian: Himmlischer Hochzeit-Schatz und geistlicher Braut-Schmuck der gläubigen und seligen Kinder Gottes. Zittau, 1664.* This picture and transcription are from Deutsches Textarchive (DTA) dataset (Geyken et al., 2012).

3 Data

3.1 Data Collection

The datasets of this project all come from a publicly available project, namely *Deutsches Textarchive (DTA)* (Geyken et al., 2012). It is a massive corpus with high-quality scans and page-by-page transcriptions of historical documents dating from the 17th century to the early 20th century. Containing more than 1,500 collections in its core corpus and more than 4,000 collections in its extended corpus, it is one of the grounding infrastructures for the research of the New High German language [2]. The documents were manually annotated by German native speakers. For most pages, format-related features like line breaks and footnotes are preserved in the transcription, but some are not (Because some transcriptions come from different sub-project contexts). In Fig. 2, one can see an example of such transcription: The transcription preserved the paragraph breaks in the original print but left out line breaks, some paragraphs at the very top, as well as more complicated layout information, like columns.

The layout problems are disregarded to limit the research focus to the topics

[2]https://www.deutschestextarchiv.de/

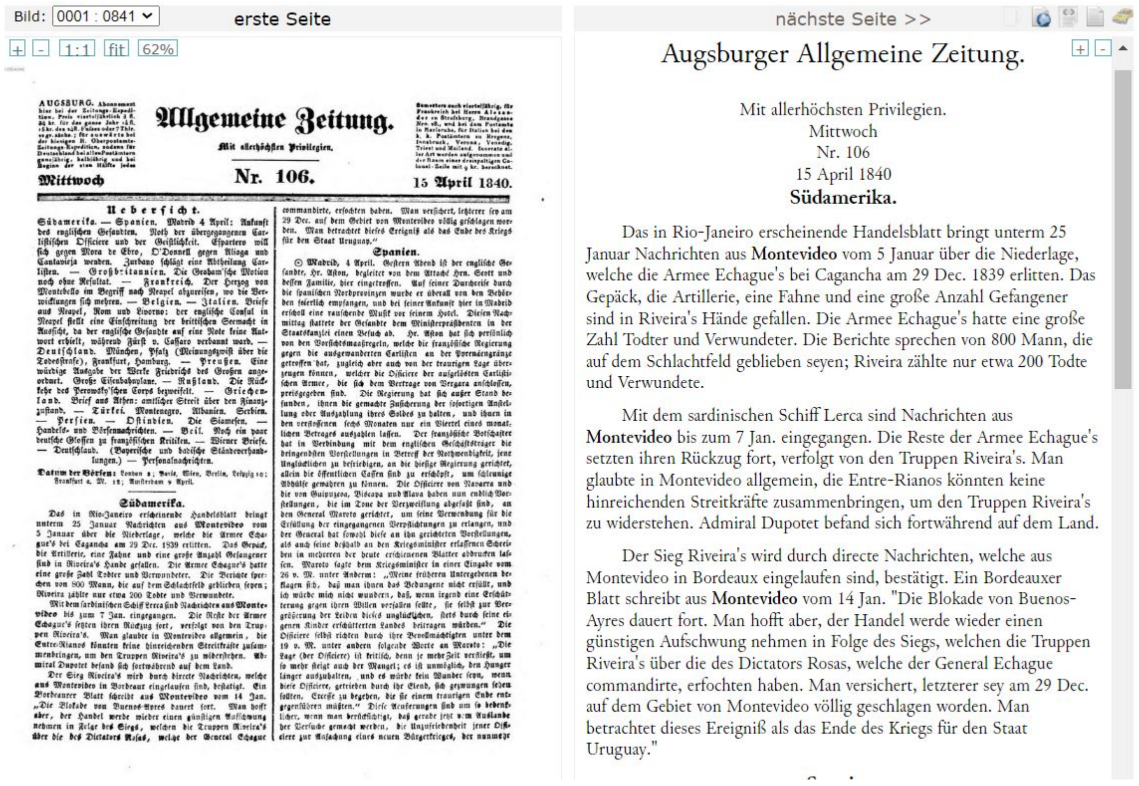


Bild: 0001 : 0841 erste Seite nächste Seite >>

Augsburger Allgemeine Zeitung.

Mit allerhöchsten Privilegien.
Mittwoch
Nr. 106
15 April 1840

Südamerika.

Das in Rio-Janeiro erscheinende Handelsblatt bringt unterm 25 Januar Nachrichten aus **Montevideo** vom 5 Januar über die Niederlage, welche die Armee Echague's bei Cagancha am 29 Dec. 1839 erlitten. Das Gepäck, die Artillerie, eine Fahne und eine große Anzahl Gefangener sind in Riveira's Hände gefallen. Die Armee Echague's hatte eine große Zahl Todter und Verwundeter. Die Berichte sprechen von 800 Mann, die auf dem Schlachtfeld geblieben seyen; Riveira zählte nur etwa 200 Todte und Verwundete.

Mit dem sardinischen Schiff Lerca sind Nachrichten aus **Montevideo** bis zum 7 Jan. eingegangen. Die Reste der Armee Echague's setzten ihren Rückzug fort, verfolgt von den Truppen Riveira's. Man glaubte in Montevideo allgemein, die Entre-Rianos könnten keine hinreichenden Streitkräfte zusammenbringen, um den Truppen Riveira's zu widerstehen. Admiral Dupotet befand sich fortwährend auf dem Land.

Der Sieg Riveira's wird durch directe Nachrichten, welche aus Montevideo in Bordeaux eingelaufen sind, bestätigt. Ein Bordeauxer Blatt schreibt aus **Montevideo** vom 14 Jan. "Die Blokade von Buenos-Ayres dauert fort. Man hofft aber, der Handel werde wieder einen günstigen Aufschwung nehmen in Folge des Siegs, welchen die Truppen Riveira's über die des Dictators Rosas, welche der General Echague commandirte, erfochten haben. Man versichert, letzterer sey am 29 Dec. auf dem Gebiet von Montevideo völlig geschlagen worden. Man betrachtet dieses Ereigniß als das Ende des Kriegs für den Staat Uruguay."

Figure 2: An example of the DTA dataset. The document scan is on the left side, and the transcription is on the right side.

mentioned in the first chapter. Therefore, it was decided to focus solely on the correction of textlines. Ultimately, a data point in our target dataset for this project should contain the following information:

1. An OCR-ed sentence, which is the noisy input to the model and awaits to be corrected;
2. The correct transcription provided by DTA, which is the golden label to the model;
3. The picture of this particular textline cropped from the document scan.

Building such a dataset requires accurate alignment between document scan and transcription to textline level. This alignment is particularly tricky: No coordinate information is locating a transcribed textline to the scan. Besides, many transcriptions have complicated layouts, making it even harder to crop. Therefore, a subset of the DTA corpus with a simple layout should be selected.

DTA tried to annotate the corpus by hand on pixel level for OCR research in 2007 and eventually gave up because of the massive workload. Nevertheless, they published a subset of the corpus in 2010 [3]. This subset contains 199 books whose publication year span from 1783 to 1903. It is also claimed to contain accurate coordinate information for every character on each page, which was found to be untrue after further inspection. However, all documents in this subset have an ideal layout: One column per page and enough space between textlines. These conditions make it very easy to crop out textline images. Aside from the advantages of image cropping condition, the publication year span of this subset falls into the most representative eras in DTA [4]. Therefore, I chose this subset as the data source. The scan images are not included in this dataset for download in batches, so I had to trace the link in the annotation and download them with *wget*. The original quality of the scans is 800px.

3.2 Data Preprocessing

After further inspection of the 199 documents, it was brought to my attention that the character coordinates are primarily incorrect and thus impossible to align with literal textlines. Because of this, alignment was included in the preprocessing phase as well. Eventually, this phase contains three major steps: 1. textline extraction; 2. textline image alignment; 3. performing OCR on all textlines.

[3]https://www.deutschestextarchiv.de/download

[4]79.8% of documents in DTA core corpus are from the 18th to the 20th century.

3.2.1 Textline Extraction

The original annotations with transcriptions and line break marks are in XML format. They also include page layout information, for example, if the page is a cover page, if a page contains a picture, or if a textline does not lay straight across a page. This information can help exclude pages with layouts that are too complicated to crop. After this step, 1.34 million textlines were extracted, of which 1.24 million textlines were used as the training set, and the remaining 100k textlines were used in the test set. All books' front and back covers were filtered out, and only their content pages were kept because the complicated layout and decorative pictures of the cover pages could cause alignment difficulties in the next step.

3.2.2 Textline - Image Alignment

This is the most challenging and time-consuming step. The tricky part is to locate a certain textline on the image. To do this, an off-the-shelf document element extractor, docExtractor (Monnier and Aubry, 2020), was employed to mark out every textline on an image. As shown in Fig. 3: a) is the original scan of a page. Based on this scan, docExtractor can produce a mask over every detected textline (see b)). I developed a heuristic algorithm to align every line using the mask images: For a page that contains n pixel rows, first sum up all the luminance l_n of every pixel in every row. This step would end up with an array of luminance sums $L = \{l_1, l_2, l_3..., l_n\}$. Then, the brightest row i with total luminance l_i can be found among L, and a threshold k should be set manually. Finally, the algorithm loops through all pixel rows, marking the rows at least k times as bright as row i. These marked rows are the rows that are blank on the page (therefore, they are the brightest), and the rest would be the ones with textlines. A threshold of 0.99 was set in this case. Through this algorithm, a list T of pixel row chunks that are marked to contain textlines was found. If the length of T is the same as the number of extracted textlines from the page, one can then identify the textline masks to be able to be fully aligned with

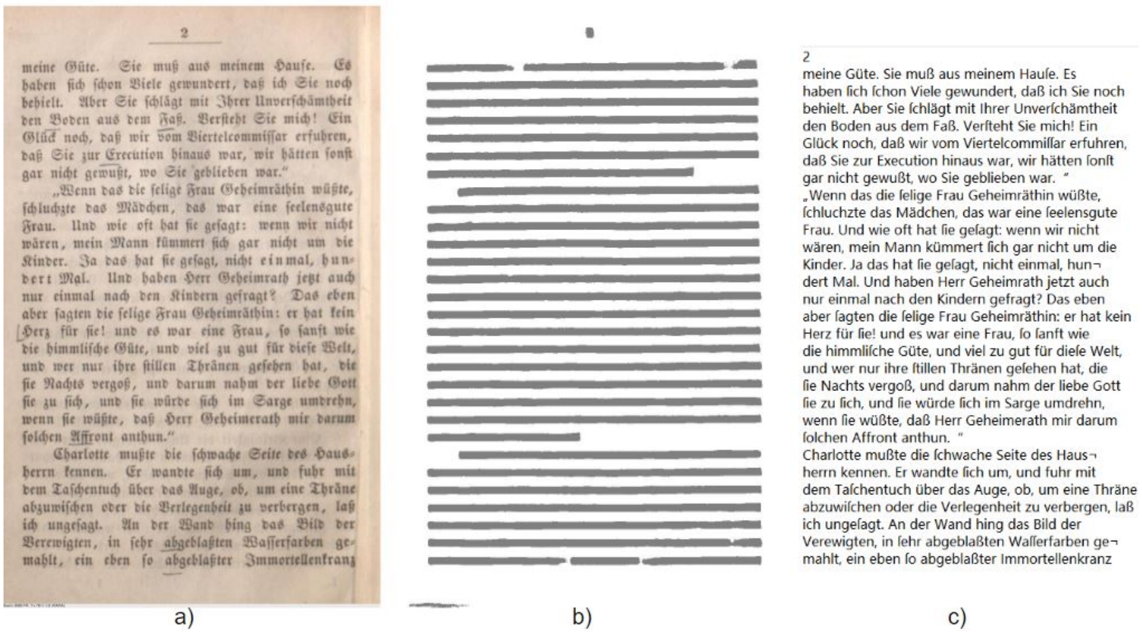
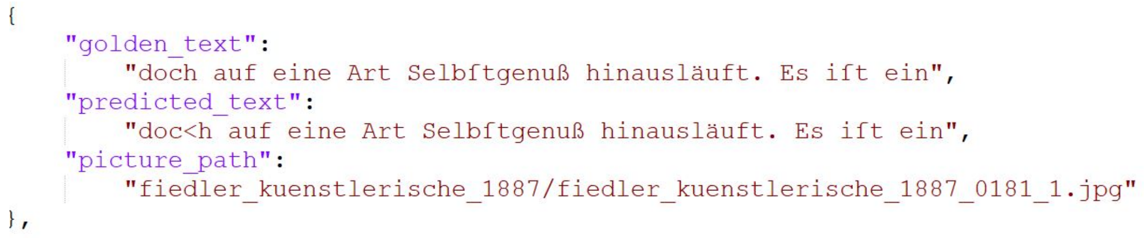

2
meine Güte. Sie muß aus meinem Hauſe. Es
haben ſich ſchon Viele gewundert, daß ich Sie noch
behielt. Aber Sie ſchlägt mit Ihrer Unverſchämtheit
den Boden aus dem Faß. Verſteht Sie mich! Ein
Glück noch, daß wir vom Viertelcommiſſar erfuhren,
daß Sie zur Execution hinaus war, wir hätten ſonſt
gar nicht gewußt, wo Sie geblieben war. "
„Wenn das die ſelige Frau Geheimräthin wüßte,
ſchluchzte das Mädchen, das war eine ſeelensgute
Frau. Und wie oft hat ſie geſagt: wenn wir nicht
wären, mein Mann kümmert ſich gar nicht um die
Kinder. Ja das hat ſie geſagt, nicht einmal, hun¬
dert Mal. Und haben Herr Geheimrath jetzt auch
nur einmal nach den Kindern gefragt? Das eben
aber ſagten die ſelige Frau Geheimräthin: er hat kein
Herz für ſie! und es war eine Frau, ſo ſanft wie
die himmliſche Güte, und viel zu gut für dieſe Welt,
und wer nur ihre ſtillen Thränen geſehen hat, die
ſie Nachts vergoß, und darum nahm der liebe Gott
ſie zu ſich, und ſie würde ſich im Sarge umdrehn,
wenn ſie wüßte, daß Herr Geheimerath mir darum
ſolchen Affront anthun. "
Charlotte mußte die ſchwache Seite des Haus¬
herrn kennen. Er wandte ſich um, und fuhr mit
dem Taſchentuch über das Auge, ob, um eine Thräne
abzuwiſchen oder die Verlegenheit zu verbergen, laß
ich ungeſagt. An der Wand hing das Bild der
Verewigten, in ſehr abgeblaßten Waſſerfarben ge¬
mahlt, ein eben ſo abgeblaßter Immortellenkranz

Figure 3: An example of textline alignment pipeline. Page scan in a) is from DTA corpus, book title: *Alexis, Willibald: Ruhe ist die erste Bürgerpflicht oder Vor fünfzig Jahren. Bd. 1. Berlin, 1852.* b) is the textline mask image of the scan. c) is a screenshot if its transcription in .txt format

the text. One can match the extracted textline m to the corresponding pixel row T_m.

However, this algorithm is only accurate when 1) docExtractor detects textlines perfectly, 2) The image is perfectly straight and not tilted, and 3) There is no error in the transcriptions. Nevertheless, as we all know, life in academia is never that perfect. This algorithm solved alignment issues for 62.6% of the pages; the rest were corrected manually. Finally, I double-checked all alignments manually as well in order to ensure all data points were golden. Finally, I cut the scans into textline images, and every one of them links to their corresponding text.



```
{
    "golden_text":
        "doch auf eine Art Selbſtgenuß hinausläuft. Es iſt ein",
    "predicted_text":
        "doc<h auf eine Art Selbſtgenuß hinausläuft. Es iſt ein",
    "picture_path":
        "fiedler_kuenstlerische_1887/fiedler_kuenstlerische_1887_0181_1.jpg"
},
```

Figure 4: An example entry of the final dataset

1750-1800	1800-1850	1850-1900	1900-1950
25	85	87	2

Table 1: The distribution of books by publication years

3.2.3 OCR on Textlines

The last step is to prepare OCR-ed text as input for the model. Tesseract (Kay, 2007), one of the most popular open-source OCR engines that supports Fraktur font, was chosen. The datasets are in *json* format. In Fig. 4, one can find an example of an entry in the dataset. *golden_text* is the original transcription; *predicted_text* is the text predicted by OCR engine, and *picture_path* records the directory where the textline image was stored. The datasets provide original printed text, OCR-ed text, and corresponding textline images for every data point.

3.3 Dataset Statistics

There are about 1.34 million textlines, among which 100k were separated into the test set, and about 1.24 million were kept for the training set. The dataset is constructed from 199 books and 103 authors, with 178 printed in Fraktur font and 21 in Antiqua. The distribution of books by their publication year is shown in Tabel. 1. The average character length of all textlines is 46.17, with a standard deviation

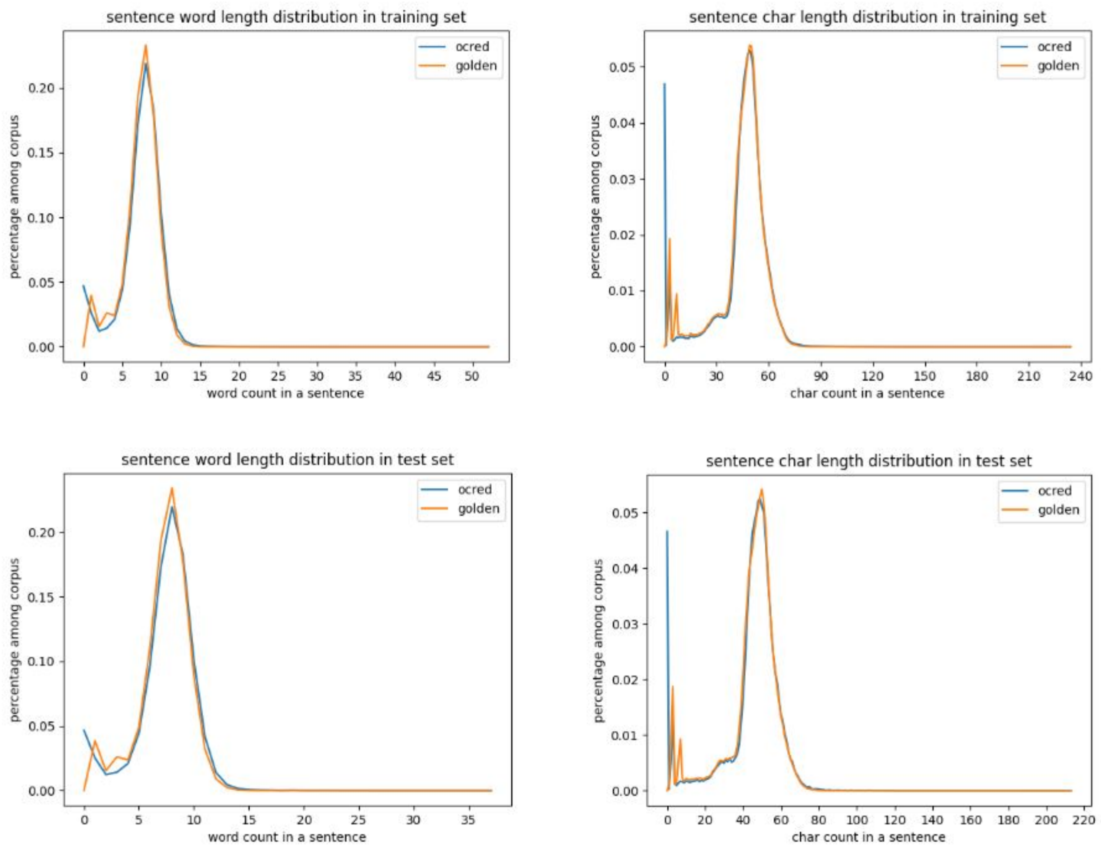


Figure 5: The length distribution of the sentences in the training set and test set

of 7.95. The average character error rate (CER) of the whole dataset is 11.48%, and the average word error rate (WER) is 28.15%. Fig. 5 illustrates the sentence length distribution in the training and test sets. The difference between the golden text and OCR-ed text is bigger when the sentence length is relatively small (word length < 5). More specifically, among the OCR-ed text, there are more sentences with a length of 0 than in the golden text. This is caused by Tesseract when it cannot produce a result with enough confidence. Besides, the peak values for the OCR-ed text are always smaller than the golden text. These observations indicate that the Tesseract OCR engine has the tendency of under-predicting.

4 Methodology

The model structure (See Fig. 6) contains three main modules: An encoder-decoder based **language model**, a **visual encoder**, and a set of **adapter modules**. The parameters of the sentence encoder and the decoder were initialized with a pre-trained ByT5 model from HuggingFace [5], and the visual encoder was initialized with pretrained ResNet from CLIP [6]. Within the sentence encoder and the decoder, there are bottleneck adapters inserted on top of self-attention layers, feed-forward layers, and cross-attention layers.

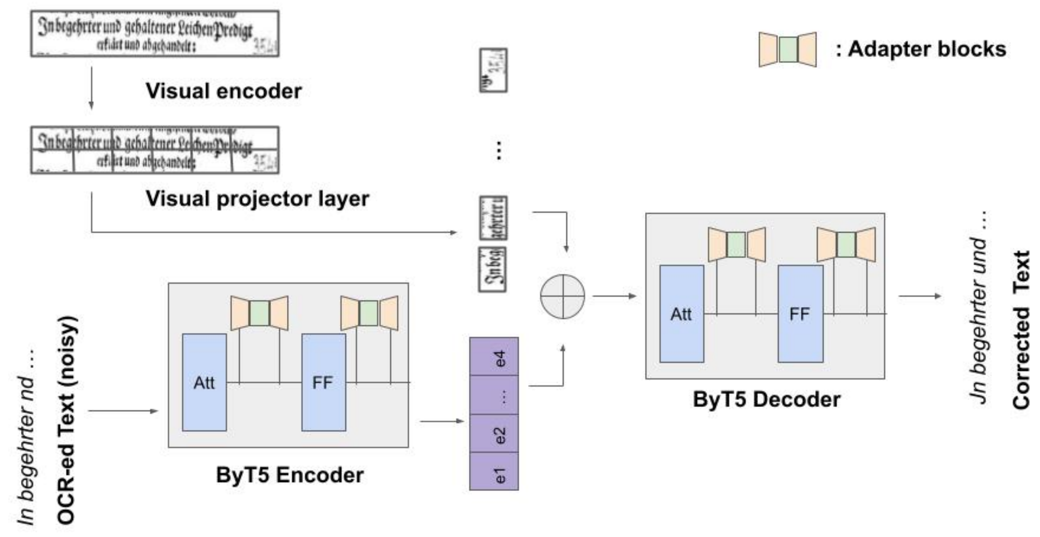


Figure 6: The model structure

[5]https://huggingface.co/docs/transformers/v4.31.0/en/model_doc/byt5
[6]https://github.com/openai/CLIP

4.1 Visual Encoder

A visual encoder V^e extracts the literal semantic information from the textline images. In this work, V^e is first initialized with a pretrained ResNet network from the multimodal model CLIP. A linear transform layer L and a dropout layer D are added on top of it to transform the outputs from ResNet to vectors that are compatible with the language model. For an input image m_i, it is first stretched to a square image through a linear transformation S, then passed through the ResNet encoder E. The feature grids of E before the pooling layer are taken. Therefore, the shape of this feature grid is $N * N$. In this case, I used CLIP-ResNet-Large, thus N=12. After this, these feature grids are flattened into a sequence of N^2 vectors and then transformed by a linear layer L. In the end, dropout regularization is applied to the vectors. The final vectors V^d are of dimension d, the same as the hidden dimension of the language model.

4.2 Language Model

The ByT5-based language model has a vanilla encoder-decoder structure using transformer blocks. It was pretrained with a span-mask denoising task on 100+ languages, including German. The encoder part of the language model takes only UTF-8 bytes inputs and, therefore, has an extremely small amount of vocabulary with a size of only 256. Compared to pretrained models of equal size, ByT5 (small) has only 0.3% of vocabulary-related parameters, while these could take up to 85% of all parameter sizes of a same-size T5 model (Raffel et al., 2020; Xue et al., 2022).

A text input y is firstly converted into a sequence of tokens $t_1, t_2, ..., t_n$. Each token is a character in the sequence y. The word embedding layer W will map every token t_i to its corresponding character embedding w_i. The encoder of the language model will then process the embeddings to a sequence of hidden states $H = h_1, h_2, ..., h_n$. The image feature vectors will be concatenated with the hidden states and fed into the decoder.

4.3 Adapters

Adapters are sets of small modules that can be inserted in between or in parallel to the layers of a transformer model. The adapter used in this work is bottleneck MLP from He et al. (2021). It can be represented mathematically as follows:

$$A(h) = h + \lambda W^{up} f(W^{down} h)$$

In the equation above, $W^{down} \in \mathbb{R}^{d_s \times d_h}$ and $W^{up} \in \mathbb{R}^{d_h \times d_s}$, where d_h is the dimension of the input h, d_s is the size of the downscale bottleneck, f is a activation function, and λ is a scaling parameter. The ratio d_h/d_s is called the *reduction ratio* and is one of the key hyper-parameters in adapter transfer learning. There are typically two kinds of adapters: parallel and sequential. A parallel adapter is set aside of a transformer layer, and the input will be passed through both the adapters and the transformer layers simultaneously. The outputs will be summed up. A sequential adapter is set behind a transformer layer, and the input will be passed through the transformer layer first and then the adapter layer. In He et al. (2021), the parallel adapters showed better performances across all downstream tasks and, therefore, were chosen in this work.

5 Experiment

This work conducted two sets of experiments: 1. The comparison between the multimodal system (CLIP-ResNet + ByT5) in this work and a SOTA baseline; 2. The comparison between the multimodal system and a single modal system with a text-only ByT5 model.

5.1 Setups

The multimodal system took CLIP-ResNet-RN50×16 for the visual encoder. It is a ResNet-50 network using 16 times scaling following EfficientNet scaling rules (Koonce and Koonce, 2021). CLIP was trained only to take square image inputs, and the smaller variants like RN50 and RN50×4 versions would require the textline images to be cropped too small. At the same time, bigger versions like RN50×64 are too large for the computation resources available. For the language model, ByT5-small was chosen because of the limited computation resources. Parallel or sequential adapters were added to self-attention layers, feed-forward layers, and cross-attention layers. The reduction ratio was set to 12 for self-attention adapters and 6 for feed-forward and cross-attention layers following the settings in the MAGMA project (Eichenberg et al., 2021). During the training for the multimodal model, the batch size was set to 128. It was trained for 10k steps with a dropout ratio of 0.1. The learning rate for the visual encoder and language model was set to 1×10^{-6} and 2×10^{-4} each. To fit such a big model into the training device in hand, I utilized *DeepSpeed* [7] *ZeRO* stage 2 (Rajbhandari et al., 2020) to parallelize gradients, models, and optimizer states across multiple GPUs (maximal 3). With the help of *DeepSpeed*, it took around 3.5 days to finish training for each model.

[7] DeepSpeed is a deep learning optimization library: https://github.com/microsoft/DeepSpeed

5.2 Baselines

The first baseline model is from Lyu et al. (2021). This model consists of an encoder combining a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) and a convolutional network. The information from these two networks is joined with multi-layer attention modules. Its decoder is a single LSTM layer. The intuition is that BiLSTM layers will capture the global context of how OCR errors are situated in the text. In contrast, deep convolutional networks capture and enforce local sub-word/phrase context, which is essential for a compound language like German. The following sections will address this model as *baseline*.

The second baseline model is simply a ByT5 model trained on this corpus with adapter tuning. This model only takes textual inputs, and the comparison between this model and the multimodal system will be used to test the helpfulness of visual information. Apart from the visual encoder, all parameters and hyper-parameters remain the same as the multimodal system described in section 5.1.

5.3 Evaluation Metrics

Two standard evaluation metrics were adopted to assess the performance of different systems: 1) Word Error Rate (WER) and 2) Character Error Rate (CER). These two metrics are both derived from the Levenshtein distance (Miller et al., 2009). They measure the number of erroneous units (word or character) caused by substitutions, insertions, and deletions on different levels. WER and CER are also the evaluation metrics in the SOTA baseline work. Therefore, direct comparison is possible.

In previous works, such as Lyu et al. (2021); Rigaud et al. (2019), the average WER and CER were used. It means that the WER and CER for every sentence s_i were calculated independently, and their non-weighted average was taken across all sentences from s_1 to s_n. Specifically:

$$\overline{M} = \frac{\sum_{i=1}^{i=n} \frac{S_i^M + I_i^M + D_i^M}{L(s_i)}}{n}$$

M denotes the evaluation metric that is either WER or CER. S_i^M, I_i^M, and D_i^M refer to substitution, insertion, and deletion on either word error level or character error level for sentence i. L denotes the length of a sentence. This could be problematic when the length of the sentences varies: A good prediction of a short sentence can boost the overall performance more than it should, and a bad prediction could also make a negative impact stronger than it deserves. One can tell from the sentence length distribution in Fig. 5 that a significant number of sentences with lengths derive from the mode. Therefore, I argue that average WER and CER are not accurate enough to describe the performance. The alternative proposed in this work is to calculate WER and CER on the corpus level. It can be described mathematically:

$$M = \frac{\sum_{i=1}^{i=n} S_i^M + I_i^M + D_i^M}{\sum_{i=1}^{i=n} L(s_i)}$$

.

6 Results

6.1 Main Results

The main results are summarized in Table 2. Under the new evaluation metrics that were mentioned in section 5.2, the multimodal system that is proposed in this work achieved the best results by reducing corpus-level WER by 56.2% and corpus-level CER by 56.8% on the basis of the test set, compared to the previous SOTA baseline's 53.5% with WER and 39.5% with CER. Interestingly, under the old average WER and CER metrics, the multimodal model from this work is "outperformed" by the SOTA baseline.

To understand these results, I conducted further inspections. It was then found out that in the dataset, there are a lot of data entries with empty OCR-ed text (4.69% in training and 5.02% in the test set). As mentioned in section 3.3, this happens when the Tesseract OCR engine cannot produce results with enough confidence. The models with or without image inputs behave very differently in this case: For the SOTA baseline model as well as the text-only ByT5 model, the input is just empty, and the models would then have to make a guess based on the same empty samples in the training set, and the guess would always be the same because of the information that is provided to them does not change. In practice, the SOTA baseline model always produced "*4*" for all the data points with empty OCR-ed text input, and the text-only ByT5 model always output "*— 1 —*". Therefore, the WER and CER differences between these two baselines do not accurately describe the systems' performance. However, the multimodal model can give better guesses because of their exposure to image features and limited OCR ability from the training with empty inputs. One can see from Table 2 that while the CER and WER of single modal models remain around 1, the CER of the results from the multimodal model is 0.594, decreased by 40.3% on the basis of the test set. At the same time, the WER remains around 1, indicating that although at times the system can not

Settings	**avg. WER**	**avg. CER**	**stddev. WER**	**stddev. CER**	**cpl. WER**	**cpl. CER**
before	0.281	0.115	0.339	0.339	0.245	0.081
baseline	**0.152**	**0.081**	0.462	0.353	0.114	0.049
t-ByT5	0.171	0.092	0.509	0.443	0.107	0.045
m-ByT5	0.157	0.087	0.740	0.850	**0.100**	**0.035**
before empty	1	1	0	0	1	1
baseline empty	**0.997**	**0.968**	0.048	0.106	**0.997**	0.995
t-ByT5 empty	1.628	1.253	1.011	0.923	1.124	0.964
m-ByT5 empty	1.451	1.179	2.830	3.518	1.054	**0.594**
before non-emp.	0.246	0.072	0.306	0.284	0.225	0.059
baseline non-emp.	0.111	0.038	0.433	0.300	0.092	0.027
t-ByT5 non-emp.	0.100	0.035	0.337	0.309	0.081	0.024
m-ByT5 non-emp.	**0.093**	**0.033**	0.304	0.292	**0.075**	**0.022**

Table 2: The results on the test set. *before* refers to the raw OCR-ed text in the test set before they are corrected. *t-ByT5* is the text-only model with ByT5 model; *m-ByT5* is the multimodal model. *empty* means the performances of models on the part of the test set where the OCR-ed text is simply an empty string. Similarly, *non-emp.* denotes the models' performances on normal non-empty inputs. *avg. WER* or CER shows the performances evaluated on the old average WER or CER metrics; *stddev.* is the standard deviation; *cpl. WER* or CER is the WER/CER on the corpus level.

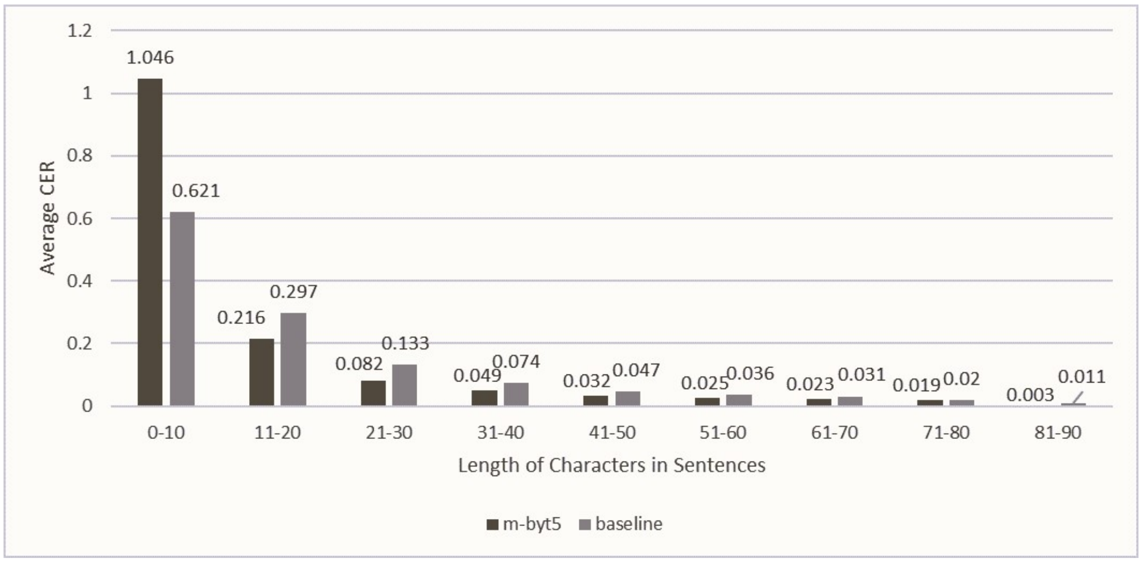


Figure 7: The average CER across sentences of different lengths

get the words correct, it can recognize parts of the characters in the words. For example, for the golden sentence *”Aber, armes Heimathland,”* with empty input, the multimodal system predicted *”Adel, armes heim tplasto,”*, with 1 for WER but 0.292 for CER.

The empty inputs are also why the baseline model ”outperformed” the multi-modal system on the average WER and CER metrics: the baseline produces ”4” as a consistent prediction on all empty inputs. When the WER or CER for each independent sentence is calculated, the number of deletion error types is very low (see Chapter 5.2 for reference), making the average WER and CER lower than the multimodal system. The latter almost always outputs much longer results, leading to higher WER/CER averages. On the other hand, the performance of the multimodal system is the best on every metric when the inputs are not empty. This analysis again supports the argument in section 5.2 that the corpus level WER and CER are better evaluation metrics to describe the performance of these systems rather than the average WER and CER.

Fig. 7 shows the average CER among different length groups of golden sentences. Two observations could be drawn here: First, the multimodal system has a better average CER in every length group except for the 0-10 one. This could be caused by the fact that in the test set, 45.15% of short golden sentences have empty OCR-ed text because of Tesseract's shortcomings. As analyzed previously, the short, consistent output from the baseline model makes the CER look lower. If those sentences with empty inputs were set aside, the average CER for this group would be 0.401 for the baseline and 0.347 for the multimodal system. Thus, it is safe to conclude that the multimodal system outperforms the baseline stably across different length-of-sentence groups. Second, it is also apparent that the models can do a better job working on longer sentences than the short ones. This is also not surprising, considering that the longer the sentence is, the more context and information is provided to the model to predict an accurate result.

6.2 Error Analysis

To analyze the errors, I manually inspected 1,000 erroneous model outputs randomly selected from the results and summarized six types of errors:

1. **Over-Segmentation**: When the model wrongly splits a word into multiple tokens in the prediction. Segmentation-related errors are often caused when the spaces in scans are not distinguishable enough. For example, if a model predicts *Heimat land* when the golden transcription is *Heimatland*.

2. **Under-Segmentation**: The opposite of over-segmentation. It refers to when the model wrongly merges multiple words into one. Like over-segmentation, under-segmentation is also a type of local error, as its effect is usually limited to 1-3 adjacent words in a sentence.

3. **Character Error**: Usually caused by misrecognized characters. It could happen because of the visual similarities between characters or the contextual

inference of the language model.

4. **Over-Prediction**: When the model predicts extra characters that do not align with any of the words in the golden transcription. For example, when the golden transcription is "*und jene Oppofition erhoben*", while the model outputs "*und jene "Oppofition erhoben]*" (with an extra symbol "]" at the end and an extra " before the word Oppofition). This would count as 2 over-predictions.

5. **Under-Prediction**: The model's output lacks some characters it is supposed to predict. For example, the model output "*habe ich befäftigt*", while the golden transcription is "*habe ich befäftigt ge¬*". Because 3 characters are missing, this sentence will contribute 3 under-prediction cases to the statistics.

6. **Over-Prediction on Lines**: This happens when the textline images include some noise from the line above or under. Fig. 8 shows an example textline image with such noise. Its golden transcription is "*latan fucht fich das weiche Wachs der Jugend, um*", but Tesseract predicted it to have two lines:

   Jatan fucht f/ ich das weiche Wachs der Jugend / um
   PE ZISP ED & SN 2.5 7,4

   The content of the latter line looks very random and comes from the tip of the words from the line below.

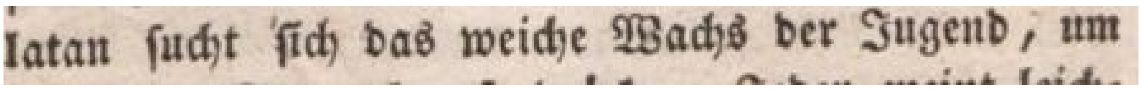

latan ſucht ſich das weiche Wachs der Jugend, um

Figure 8: An example textline image with noise from the line below

Because of so many different types of errors, it is necessary to align the model predictions with the golden transcription to count the frequency of different error types. I used the Needleman-Wunsch Algorithm (Needleman and Wunsch, 1970) implemented by *string2string* (Suzgun et al., 2023) to perform global pairwise sequence alignment on the golden transcription and the noisy text. The algorithm essentially divides a full sequence into a series of sub-sequences and uses the solutions to the sub-sequences to reconstruct a solution back to the full sequence. An example result of the Needleman-Wunsch Algorithm is in Fig. 9. The first line is the golden transcription, and the textline below is its corresponding predicted text. The green parts are correctly predicted, and the red parts are examples of character errors. The symbol ”@” is a necessary placeholder for the alignment when gaps or characters do not exist in the text. The grey underline shows a case of under-segmentation, and the blue underline indicates a case of over-prediction.

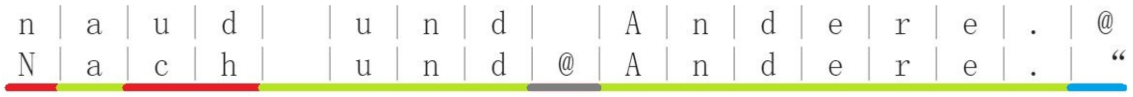

n | a | u | d | | u | n | d | | A | n | d | e | r | e | . | @
N | a | c | h | | u | n | d | @ | A | n | d | e | r | e | . | “

Figure 9: An example of the Needleman-Wunsch Algorithm

As shown in Table 3, over-segmentation is the second common error type in the test set before corrections, but it also seems to be the easiest to correct. After the correction, the amount of over-segmentation errors dropped by 93.0% for the baseline model and 95.2% for the multimodal system. In contrast, under-segmentation is the rarest error type, and it is also the only error type where the baseline model outperforms the multimodal system. The frequency of such errors dropped by 2.2% after the correction of the baseline model but only 1.2% for the multimodal system. An observation in the baseline paper (Lyu et al., 2021) can potentially help understand these results: Word segmentation errors often have local behaviors and local context, such as merging or splitting words. The convolutional neural networks used in this baseline model seem very good at capturing this type of behavior. That could be why the baseline performs very well in segmentation errors in general, and it is even better at correcting under-segmentations than the multimodal system.

Character errors are the most frequent error type. The baseline model can decrease 55.3% of such errors, and the multimodal system can do so by 61.1%.

The performance differences between the baseline model and the multimodal system become larger regarding over-prediction and under-prediction. Especially under-prediction. Over-predictions can happen when the images are noisy or when the language model is over-predicting for contextual reasons. While under-prediction usually happens when a part of the textline images is compromised or not clearly printed. These lead the Tesseract to over- or under-predict and affect the results in OCR post-correction. According to the manual inspection, I found that the beginning and ending parts of the sentences are most prone to these types of errors. This could be caused by two reasons: First, The tokens at the two ends of the sentences are often incomplete words that span multiple rows. This may lead the model to complete the tokens or ignore them. Second, the scans are sometimes noisy. This problem is especially tricky in under-prediction when often the OCR-ed texts are incomplete to begin with. This means that the model has to make up the part that does not exist in its input. With only textual input, the baseline model had difficulty solving this type of error. In contrast, with the help of visual information, the multimodal system can predict much better.

Settings	**Over-seg.**	**Under-seg.**	**Char Err.**	**Over-pred**	**Under-pred**	**Lines**[1]
before	12,919	852	123,892	56,656	44,297	2,228
baseline	899 -93.0%	833 -2.2%	55,402 -55.3%	26,164 -53.9%	31,475 -28.9%	0 -100%
m-ByT5	622 -95.2%	845 -1.2%	48,244 -61.1%	17,789 -69.6%	18,092 -59.2%	0 -100%

Table 3: The frequency of error types before the correction, in the baseline, and after multimodal system correction. [1]: Over-Prediction on Lines. Data points with empty OCR-ed text are excluded in this statistics

Lastly, character errors were and are still the most frequent and challenging error type. This type of error can be caused by visual similarities between characters (See Appendix A for German alphabets in Fraktur font). Fig. 10 compares the multimodal system and the text-only baseline. Some observations can be drawn here: 1. Similar to the finding in Lyu et al. (2021), there are no simple mappings between misrecognized characters. Meaning that these character errors are not only caused by their visual similarities but also contextual reasons, as they are often misrecognized to completely different characters; I will inherit the naming in Lyu et al. (2021) and call them *contextual misrecognitions*; 2. the multimodal system can reduce the majority of the most common character errors with some exceptions; Interestingly, the multimodal system does not decrease the misrecognition between visually similar characters as much as one might think. For example, the multimodal system tends to confuse the character *d* and *v* even more than the text-only model (These two characters are highly similar in Fraktur font). This means the multimodal system is also confused with visually similar characters and cannot accurately tell the subtle differences. This could be potentially improved if the system can use image inputs with better resolutions. However, the multimodal system seems good at correcting contextual misrecognition of the characters with more significant visual differences. From Fig. 10, the multimodal system can decrease the mistaken s from *ſ* by nearly 60%. These two characters are sometimes interchangeably used in the corpus (for example, *sie* and *ſie*). While they are sometimes used in a similar context, they look very different. Therefore, the multimodal system seems to incorporate the information in the image features very well.

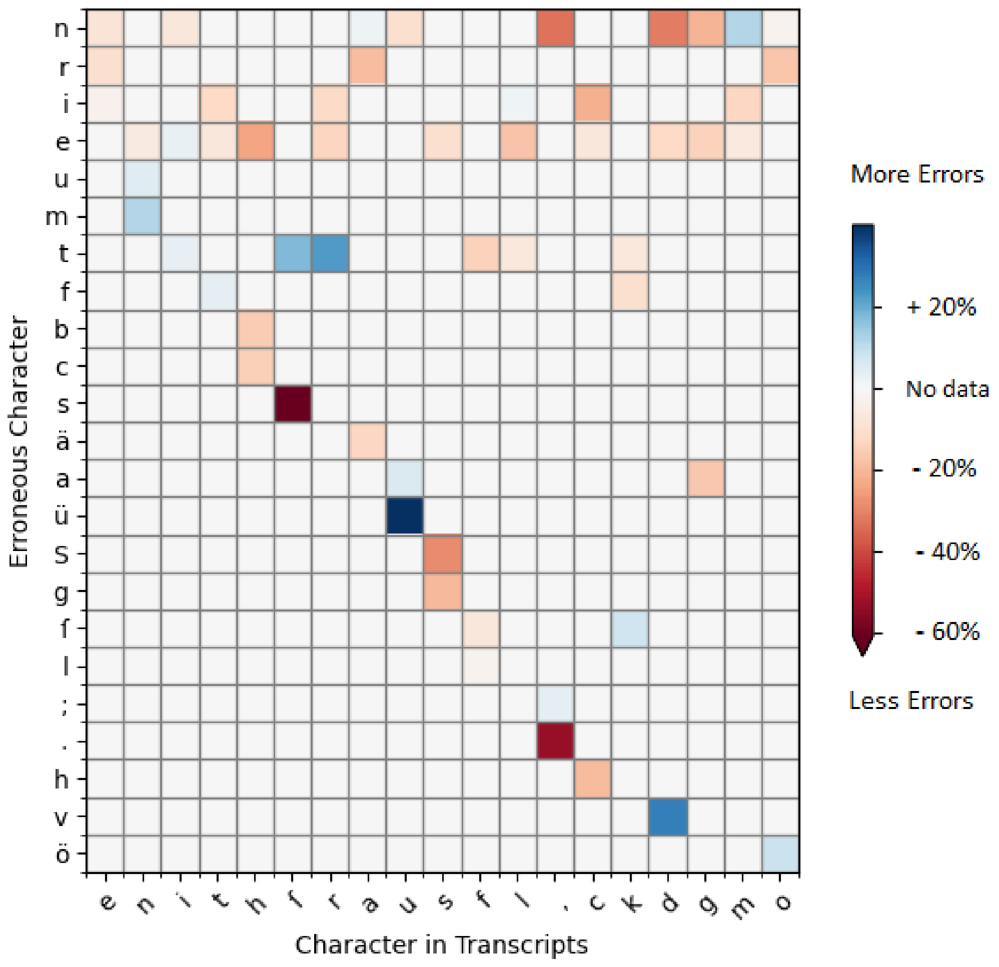


Figure 10: This heatmap shows how using image features as part of the input changes the distribution of the most frequently misrecognized characters. On the x-axis lies the golden characters from the transcripts, and on the y-axis lies the misrecognized characters. I took the most frequent character errors from the text-only ByT5 model and compared the frequency of the same errors among the outputs of the multimodal system. The warmer the color is, the more percentages of mistakes are decreased by the multimodal system.

7 Conclusion

In this thesis, I constructed a dataset based on Deutsches Textachiv (Geyken et al., 2012). It contains 1.34 million transcription textlines, its OCR-ed text from the Tesseract OCR engine, and the aligned textline images correspond to the textlines. I also explored a multimodal OCR Post-Correction system that uses adapter-tuned ByT5 (Xue et al., 2022) and CLIP-ResNet-Larg (Radford et al., 2021). The experiments show that this system outperforms the SOTA system in corpus-level Word Error Rate (WER) and Character Error Rate (CER). In another comparison, it beats a single-modal baseline consisting of only a ByT5 model. Besides, I proposed to use corpus-level WER and CER instead of average WER and CER for more accurate evaluation in OCR post-correction tasks.

The analysis shows that the multimodal system significantly reduced all types of errors. It is better at correcting almost all types of errors than the SOTA baseline. The SOTA baseline is, however, better at correcting under-segmentation errors. One possible reason is that the SOTA model used convolutional networks in their language model and is thus good at correcting local errors like segmentation.

Another comparison between this multimodal system and single-modal baseline shows that with the help of a visual encoder, the multimodal system can correct character errors better in general, and it is especially good at identifying different-looking characters that are interchangeable contextually. Nevertheless, it is not especially good at identifying the nuances between characters that are very similar visually. This is possibly caused by the limit of the CLIP visual encoder: CLIP-ResNet-Large can only accept pictures in the shape of 384px $\times$ 384px. This limitation means that the textline images are less clear and informative than the original scans. For future work, one can try to experiment with a visual encoder that accepts bigger textline images.

Good performance aside, the training of the multimodal system took twice as long as the baseline model. This is because adapter tuning would typically decrease the use of memory and computation complexity of the model with the cost of increasing the time to convergence. Besides, as mentioned in Xue et al. (2022), character-based models like ByT5 would also have higher Floating Point Operations (FLOPs) because of longer sequences than word-based models. Consisting of two pretrained models with a large number of parameters, this multimodal system is also more memory-demanding compared to the baseline model. I tried to compare the performance between this multimodal with fine-tuning and adapter-tuning but failed due to *Out of Memory* error.

Moreover, notice that in this thesis, the visual information and text input were fused together after the text inputs were already passed through the encoder of the language model. Some works from computer science and neuroscience show that models can typically benefit from an early fusion (Barnum et al., 2020; Schroeder and Foxe, 2005; Budinger et al., 2006). This means combining visual and textual information at an earlier stage. For example, concatenate the image features and text input before passing them through the language model encoder. It would also be interesting to see how OCR post-correction task benefits from such early fusion.

8 Acknowledgement

I would like to thank Prof. Dr. Sebastian Padó for the initial idea of this thesis. He also helped me construct the model structure and offered a lot of guidance. Besides, thank Mr. Niklas Wolf and Ms. Zhuge Gao for proofreading this thesis and for their spiritual support during the project. Finally, I thank my mother, Ms. Xiaolan Liu, for her pushing. Without her, I would have procrastinated a lot more.

References

George Barnum, Sabera Talukder, and Yisong Yue. On the benefits of early fusion in multimodal representation learning. *arXiv preprint arXiv:2011.07191*, 2020.

Youssef Bassil and Mohammad Alwani. Ocr post-processing error correction algorithm using google online spelling suggestion. *arXiv preprint arXiv:1204.0191*, 2012.

Werner Besch and Norbert Richard Wolf. *Geschichte der deutschen Sprache: Längsschnitte-Zeitstufen-linguistische Studien*, volume 47. Erich Schmidt, 2009.

Hicham El Boukkouri, Olivier Ferret, Thomas Lavergne, Hiroshi Noji, Pierre Zweigenbaum, and Junichi Tsujii. Characterbert: Reconciling elmo and bert for word-level open-vocabulary representations from characters. *arXiv preprint arXiv:2010.10392*, 2020.

E Budinger, P Heil, A Hess, and H Scheich. Multisensory processing via early cortical stages: connections of the primary auditory cortical field with other sensory systems. *Neuroscience*, 143(4):1065–1083, 2006.

Simon Clematide, Lenz Furrer, and Martin Volk. Crowdsourcing an ocr gold standard for a german and french heritage corpus. 2016.

Panos Constantopoulos, Martin Doerr, Maria Theodoridou, and Manolis Tzobanakis. Historical documents as monuments and as sources. In *Computer Applications and Quantitative Methods in Archaeology Conference (CAA 2002), Heraklion, April 2002*, 2002.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Constantin Eichenberg, Sidney Black, Samuel Weinbach, Letitia Parcalabescu, and Anette Frank. Magma–multimodal augmentation of generative models through adapter-based finetuning. *arXiv preprint arXiv:2112.05253*, 2021.

Florian Fink, Klaus U Schulz, and Uwe Springmann. Profiling of ocr'ed historical texts revisited. In *Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage*, pages 61–66, 2017.

Alexander Geyken, Susanne Haaf, and Frank Wiegand. The dta'base format': A tei-subset for the compilation of interoperable corpora. In *KONVENS*, pages 383–391, 2012.

Thomas Gloning and Christopher Young. *A history of the German language through texts*. Routledge, 2004.

Mika Hämäläinen and Simon Hengchen. From the paft to the fiiture: a fully automatic nmt and word embeddings method for ocr post-correction. *arXiv preprint arXiv:1910.05535*, 2019.

Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on vision transformer. *IEEE transactions on pattern analysis and machine intelligence*, 2022.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Marcin Heliński, Miłosz Kmieciak, and Tomasz Parkoła. Report on the comparison of tesseract and abbyy finereader ocr engines. 2012.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

Zheng Huang, Kai Chen, Jianhua He, Xiang Bai, Dimosthenis Karatzas, Shijian Lu, and CV Jawahar. Icdar2019 competition on scanned receipt ocr and information extraction. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1516–1520. IEEE, 2019.

Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. Ammus: A survey of transformer-based pretrained models in natural language processing. *arXiv preprint arXiv:2108.05542*, 2021.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035, 2021.

Anthony Kay. Tesseract: an open-source optical character recognition engine. *Linux Journal*, 2007(159):2, 2007.

Brett Koonce and Brett Koonce. Efficientnet. *Convolutional Neural Networks with Swift for Tensorflow: Image Recognition and Dataset Categorization*, pages 109–123, 2021.

Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*, 2019.

Nils Langer. Standard german in the eighteenth century. *Norms and use. Rutten et al*, pages 277–302, 2014.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019.

Minghao Li, Tengchao Lv, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. Trocr: Transformer-based optical character recognition with pre-trained models. *arXiv preprint arXiv:2109.10282*, 2021.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

Lijun Lyu, Maria Koutraki, Martin Krickl, and Besnik Fetahu. Neural ocr post-hoc correction of historical corpora. *Transactions of the Association for Computational Linguistics*, 9:479–493, 2021.

Pengyuan Lyu, Chengquan Zhang, Shanshan Liu, Meina Qiao, Yangliu Xu, Liang Wu, Kun Yao, Junyu Han, Errui Ding, and Jingdong Wang. Maskocr: Text recognition with masked encoder-decoder pretraining. *arXiv preprint arXiv:2206.00311*, 2022.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. *arXiv preprint arXiv:2106.04489*, 2021.

U-V Marti and Horst Bunke. The iam-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5(1):39–46, 2002.

Jamshed Memon, Maira Sami, Rizwan Ahmed Khan, and Mueen Uddin. Handwritten optical character recognition (ocr): A comprehensive systematic literature review (slr). *IEEE Access*, 8:142642–142668, 2020.

Frederic P Miller, Agnes F Vandome, and John McBrewster. Levenshtein distance: Information theory, computer science, string (computer science), string metric, damerau? levenshtein distance, spell checker, hamming distance, 2009.

Rishabh Mittal and Anchal Garg. Text extraction using ocr: A systematic review. In *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, pages 357–362, 2020. doi: 10.1109/ICIRCA48905.2020.9183326.

Kareem Mokhtar, Syed Saqib Bukhari, and Andreas Dengel. Ocr error correction: State-of-the-art vs an nmt-based approach. In *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, pages 429–434. IEEE, 2018.

Tom Monnier and Mathieu Aubry. docExtractor: An off-the-shelf historical document element extraction. In *ICFHR*, 2020.

Gustav Must. The symbols for/i/and/j/in german orthography: An historical survey. *MLN*, 80(5):584–595, 1965.

Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970.

Clemens Neudecker, Konstantin Baierer, Maria Federbusch, Matthias Boenig, Kay-Michael Würzner, Volker Hartmann, and Elisa 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*, pages 53–58, 2019.

Juan Carlos Perez-Cortes, Juan-Carlos Amengual, Joaquim Arlandis, and Rafael Llobet. Stochastic error-correcting parsing for ocr post-processing. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 4, pages 405–408. IEEE, 2000.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. Adapterhub: A framework for adapting transformers. *arXiv preprint arXiv:2007.07779*, 2020a.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. Mad-x: An adapter-based framework for multi-task cross-lingual transfer. *arXiv preprint arXiv:2005.00052*, 2020b.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE, 2020.

Ulrich Reffle and Christoph Ringlstetter. Unsupervised profiling of ocred historical documents. *Pattern Recognition*, 46(5):1346–1357, 2013.

Martin Riedl, Daniela Betz, and Sebastian Padó. Clustering-based article identification in historical newspapers. In *Proceedings of the 3rd Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 12–17, 2019.

Christophe Rigaud, Antoine Doucet, Mickaël Coustaty, and Jean-Philippe Moreux. Icdar 2019 competition on post-ocr text correction. In *2019 international conference on document analysis and recognition (ICDAR)*, pages 1588–1593. IEEE, 2019.

Robin Schaefer and Clemens Neudecker. A two-step approach for automatic ocr post-correction. In *Proceedings of the The 4th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 52–57, 2020.

Justus Georg Schottel. *Teutsche Sprachkunst*. 1977.

Charles E Schroeder and John Foxe. Multisensory contributions to low-level,'unisensory'processing. *Current opinion in neurobiology*, 15(4):454–458, 2005.

Sarah Schulz and Jonas Kuhn. Multi-modular domain-tailored ocr post-correction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2716–2726, 2017.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

Sheng Shen, Liunian Harold Li, Hao Tan, Mohit Bansal, Anna Rohrbach, Kai-Wei Chang, Zhewei Yao, and Kurt Keutzer. How much can clip benefit vision-and-language tasks? *arXiv preprint arXiv:2107.06383*, 2021.

Amanpreet Singh, Guan Pang, Mandy Toh, Jing Huang, Wojciech Galuba, and Tal Hassner. Textocr: Towards large-scale end-to-end reasoning for arbitrary-shaped scene text. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8802–8812, 2021.

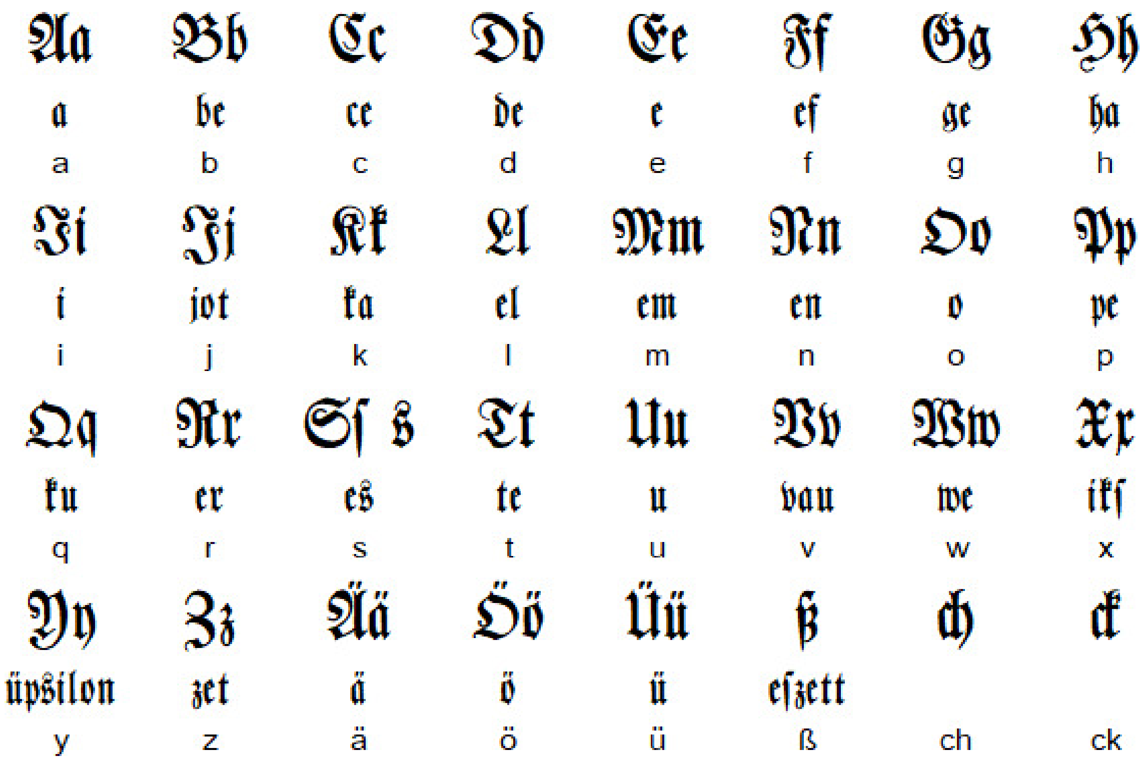
Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5227–5237, 2022.

Mirac Suzgun, Stuart M Shieber, and Dan Jurafsky. string2string: A modern python library for string-to-string algorithms. *arXiv preprint arXiv:2304.14395*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306, 2022.

Appendix A



Aa	Bb	Cc	Dd	Ee	Ff	Gg	Hh
a	be	ce	de	e	ef	ge	ha
a	b	c	d	e	f	g	h
Ii	Jj	Kk	Ll	Mm	Nn	Oo	Pp
i	jot	ka	el	em	en	o	pe
i	j	k	l	m	n	o	p
Qq	Rr	Sſ s	Tt	Uu	Vv	Ww	Xx
ku	er	es	te	u	vau	we	ikſ
q	r	s	t	u	v	w	x
Yy	Zz	Ää	Öö	Üü	ß	ch	ck
üpsilon	zet	ä	ö	ü	eſzett		
y	z	ä	ö	ü	ß	ch	ck

Figure 11: German Alphabets in Fraktur font. This picture was taken from Yale University Library: https://web.library.yale.edu/cataloging/music/fraktur