CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

# Vision Language Model Based Systems for Optical Character Recognition of Historical Swedish Newspaper Material

Master's thesis in Computer science and engineering

Martin Johansson, Selma Waginder

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2026

Master's thesis 2026

# Vision Language Model Based Systems for Optical Character Recognition of Historical Swedish Newspaper Material

Martin Johansson, Selma Waginder

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2026

Vision Language Model Based Systems
for Optical Character Recognition
of Historical Swedish Newspaper Material



Supervisor: Dana Dannélls, Språkbanken Text, University of Gothenburg
Examiner: Krasimir Angelov, Department of Computer Science and Engineering

Master's Thesis 2026
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2026

Vision Language Model Based Systems
for Optical Character Recognition
of Historical Swedish Newspaper Material

Martin Johansson, Selma Waginder
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

# Abstract

The ability to digitize old scanned newspapers plays an important role in improving searchability and making information accessible. To convert text in images into machine-readable form, an Optical Character Recognition engine is employed. In this thesis, a dataset Swedish newspaper material from 1818-1904 is used. The report investigates whether small to medium-sized open-source Vision Language Models are competitive for Optical Character Recognition compared to traditional models. It is found that a fine-tuned Qwen3-VL-8B-Instruct in combination with a simple repetition trimmer is able to outperform the traditional OCR engine Abbyy FineReader version 11.1.16 by 68.5% in terms of CER on this particular dataset and set a new record at 1.930% CER. This thesis demonstrates that the current generation of small open source Vision Language Models are highly competitive with traditional OCR engies for transcription of 19th century Swedish newspaper material. The thesis also thoroughly investigates the particular quirks and failure modes of different OCR systems through a qualitative analysis. Our best model performs no better on the training set than on the test set, suggesting that our fine-tuning was bottle necked by the LoRA adapter size and that one could potentially achieve an even stronger model with a larger adapter.

Keywords: Vision Language Model, VLM, machine learning, Optical Character Recognition, OCR, historical newspapers

# Acknowledgements

We would like to give a massive thanks to our supervisor Dana Dannélls for all the guidance and feedback she has provided us with throughout this project. We would also like to express our gratitude to all of the friends we have made along the way during our time at Chalmers.

Martin Johansson, Selma Waginder, Gothenburg, 2026-02-13

# Contents

# List of Figures

# List of Tables

# 1

# Introduction

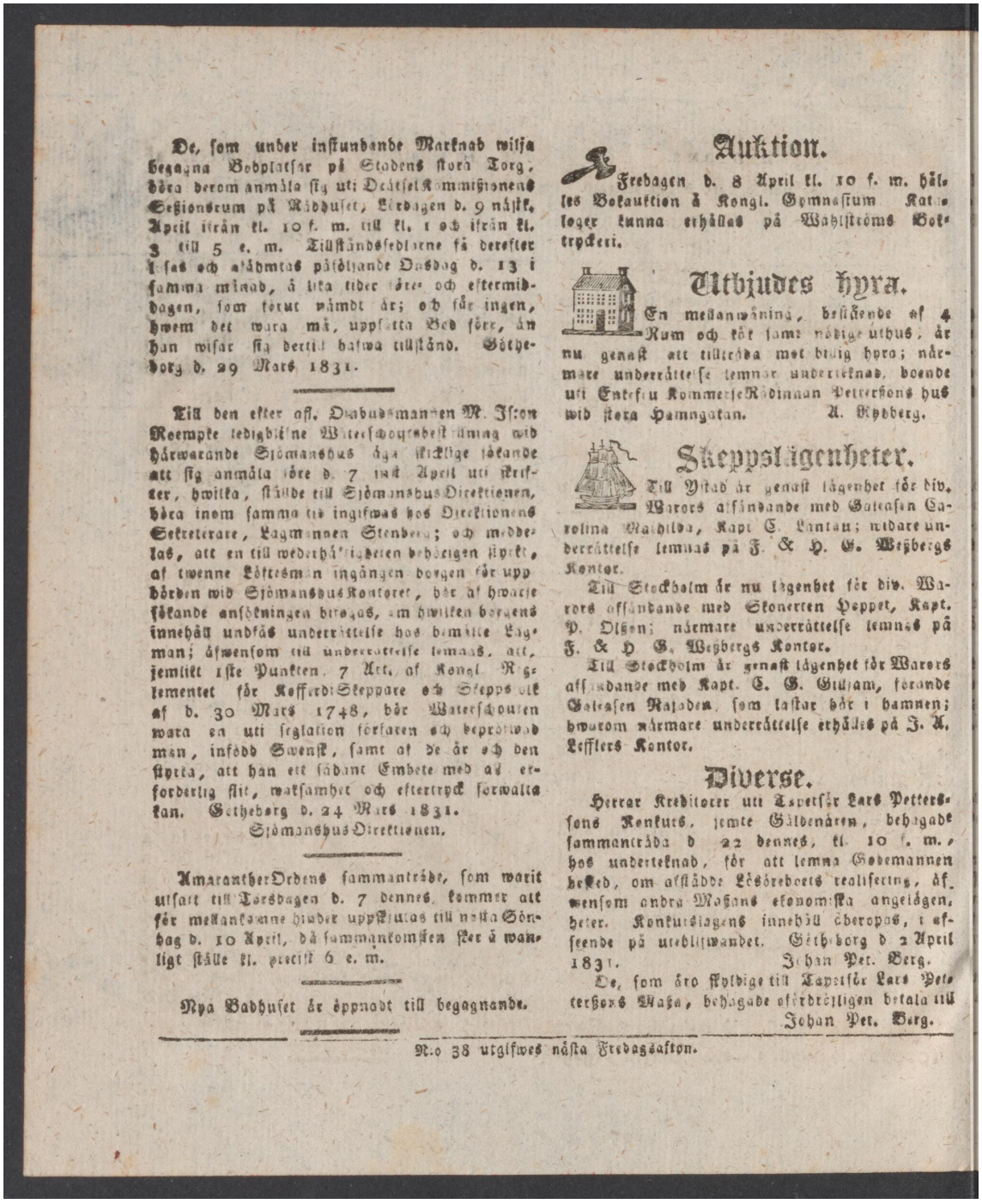
Libraries around the world store vast collections of historical newspapers. Kungliga Biblioteket, the National Library of Sweden, have scanned newspapers dating back to the middle of the 17th century. These collections are invaluable resources for historians and journalists, both professional ones and hobbyists. The accessibility of the archives have previously been limited because of the fragility and irreplaceability of the material. To address this, libraries have in recent decades begun to digitize their collections and make the resources available to anyone with access to the internet.

Digitization is done in two main steps. Firstly, the material is scanned using an image scanner. Secondly, the images of the pages are ran through Optical Character Recognition (OCR) software, which extracts the pure text from the material. This enables data analysis and full searchability of the content. One can for example answer questions that were previously unfeasible, such as how the prevalence of a certain phrase has varied over time.

A current subject of research is the accuracy of OCR engines. The accuracies of OCR on modern material are usually satisfactory, whereas the transcriptions of historical material often suffer from high error rates. The reason for this is multifaceted. OCR engines have primarily been developed using modern text because of the abundant availability of it. Historical material differs from modern in four main ways. Firstly, spelling conventions have varied throughout history. Secondly, Blackletter typefaces, who differ significantly from the Antiqua typefaces used today, were commonly used. Thirdly, the prints of the individual paragraphs were manually placed on the pages, leading to them not being perfectly aligned on the pages. Lastly, the material itself is often degraded after centuries of storage. It often suffers from warping, buckling, ink bleed-through, and sometimes even stains. An example of a scan from 1831 can be found in Figure 1.1.

## 1.1 Aim of the Work

This thesis evaluates the effectiveness of utilizing small to medium sized open-source Vision Language Models (VLMs) as OCR engines for 19th century Swedish newspaper material. It compares both stock and fine-tuned VLMs to traditional OCR engines. Outputs from traditional OCR engines like Tesseract [2] and Abbyy



De, som under instundande Marknad wilja begagna Bodplatser på Stadens stora Torg, höra derom anmäla sig uti Drätsel-Kommissionens Sessionsrum på Rådhuset, Lördagen d. 9 nästk. April ifrån kl. 10 f. m. till kl. 1 och ifrån kl. 3 till 5 e. m. Tillståndsedlarne få derefter i sas och utdömtas påföljande Onsdag d. 13 i samma månad, å lika tider för- och eftermiddagen, som förut nämdt är; och får ingen, hwem det wara må, uppsätta Bod förr, än han wisar sig dertill hafwa tillstånd. Göthe-borg d. 29 Mars 1831.

Till den efter afl. Ombudsmannen M. Jsson Reempke ledigblifne Wattenschoutsbest ämbet med härwarande Sjömanshus äga skicklige sökande att sig anmäla före d. 7 inst. April uti skrifter, hwilka, ställde till Sjömanshus-Direktionen, höra inom samma tid ingifwas hos Direktionens Sekreterare, Lagmannen Stenberg; och meddelas, att en till wederhäftigheten behörigen styrkt, af twenne Löftesmän ingången borgen för uppbörden wid Sjömanshus-Kontoret, bör af hwarje sökande anskaffas, om hwilken borgens innehåll undfås underrättelse hos bemälte Lagman; åfwensom till underrättelse lemnas, att, jemlikt 1ste Punkten 7 Art. af Kongl. Reglementet för Kofferdi-Skeppare och Skepps-folk af d. 30 Mars 1748, bör Waterschouten wara en uti seglation förfaren och beprofwad man, infödd Swensk, samt af de år och den styrka, att han ett sådant Embete med af erforderlig flit, waksamhet och eftertryck förwalta kan. Gothenburg d. 24 Mars 1831.

Sjömanshus-Direktionen.

Amaranther-Ordens sammanträde, som warit utsatt till Torsdagen d. 7 dennes, kommer att för mellankommne hinder uppskjutas till nästa Söndag d. 10 April, då sammankomsten sker å wanligt ställe kl. precist 6 e. m.

Nya Badhuset är öppnadt till begagnande.

## Auktion.

Fredagen d. 8 April kl. 10 f. m. hålles Bokauktion å Kongl. Gymnasium. Kataloger kunna erhållas på Wahlströms Boktryckeri.

## Utbjudes hyra.

En mellanwåning, bestående af 4 Rum och för samt nödige uthus, är nu genast att tillträda mot billig hyra; närmare underrättelse lemnar undertecknad, boende uti Enkefru Kommerse-Rådinnan Pettersons hus wid stora Hamngatan. A. Rydberg.

## Skeppslägenheter.

Till Ystad är genast lägenhet för div. Warors afsändande med Galeasen Carolina Mathilda, Kapt. C. Lantau; widare underrättelse lemnas på F. & H. G. Wetzbergs Kontor.

Till Stockholm är nu lägenhet för div. Warors afsändande med Skonerten Hoppet, Kapt. P. Olsen; närmare underrättelse lemnas på F. & H. G. Wetzbergs Kontor.

Till Stockholm är genast lägenhet för Warors afsändande med Kapt. C. G. Gidlam, förande Galeasen Rajaden, som lastar här i hamnen; hwarom närmare underrättelse erhålles på J. A. Lefflers Kontor.

## Diverse.

Herrar Kreditorer uti Tapetsör Lars Pettersons Konkurs, jemte Gäldenären, behagade sammanträda d. 22 dennes, kl. 10 f. m., hos undertecknad, för att lemna Godemannen besked, om afsiddde Lösöreboets realisering, åfwensom andra Massans ekonomiska angelägenheter. Konkurslagens innehåll åberopas, i afseende på uteblifwandet. Gothenburg d. 2 April 1831. Johan Pet. Berg.

De, som äro skyldige till Tapetsör Lars Petterssons Massa, behagade oförtröttligen betala till Johan Pet. Berg.

N:o 38 utgifwes nästa Fredagsafton.

Figure 1.1: Scan of *Aftonbladet*, 5th of April 1831 [1]. This scan shows the challenges of reading and understanding old newspapers, including stains, ink bleeding, and different spelling and word choice compared to modern texts.

FineReader [3] can be seen in Table 1.1. It exemplifies how much traditional engines struggle with older material. Errors made by the models are marked red.

Table 1.1: An excerpt from *Stockholms Dagblad*, 27th of November 1826 [4], with its corresponding prediction using Tesseract and Abbyy FineReader. Ground truth was transcribed by Grepect GmbH [5]. Errors made by the models are marked red.

| | |
|---|---|
| **Ground truth** | J dag Måndag den 27:de dennes, f. m. från kl. 10, förläljes å denne Stadens Auktions=Kammare |
| **Tesseract** | VR dag Mändag den 27:de dennes, f, m. från fl. 10, förfäljes à denne Stadens Auktions-Kammare |
| **Abbyy FineReader** | I dag Mändagden zyM denneS, f. m. från kl. 10, för'äljeS ä denne Stadens AiiktionS-Kammare |

## 1.2 Research Questions

The research questions investigated in this thesis are the following:

- Are OCR systems based on small to medium sized open-source VLMs competitive with traditional OCR engines for OCR of 19th century Swedish newspaper material?
- To what extent can the performance of VLM based OCR systems be improved by fine-tuning the VLM using a small dataset?

## 1.3 Limitations

The scope of this thesis is focused on 19th century newspapers, with material ranging from 1818 to 1904. This likely means that the performance of the fine-tuned models are poor on newspaper material much older than that. This is unfortunate, but leaves room for further projects on the subject. One of the reasons for the exclusion of 20th century newspapers is that copyright law still applies to most of them (the ones newer than 100 years). Since Swedish spelling has remained mostly the same during the 20th century, performance might still end up being satisfactory on newer material. Either way, this might not be very important since traditional OCR engines perform well on modern texts, at a lower computational cost. The very most recent material will likely not be in need of digitization at all, since the material was already digital from the start. This project is also limited to transcribing the textual content of the material, without any labeling of boldface, italics, headings and so on. The reason for this is that our ground truth does not include such information.

## 1.4 Ethical Considerations

Training and deploying machine learning models require energy and resources. According to Morand et al. [6] (2024) the energy used for training machine learning

models and the manufacturing of the graphics cards used for it has increased massively since 2012, despite reduction strategies and optimizations. With the increased use of machine learning systems, it is not unreasonable that resource consumption will increase, however, it is important to acknowledge the consequences. Morand et al. conclude that the use of machine learning models must both be reduced and be more efficient in order to reduce the environmental impact. With this in mind, this project uses small to medium sized models that consume orders of magnitude less resources than the largest models of today.

Regarding using OCR to increase the accessibility of old newspapers there are a few ethical considerations. There are statements, language and opinions expressed in old newspapers that are considered offensive and inappropriate today, including but not limited to racism, sexism and pro colonialist attitudes. It could be argued that increasing the accessibility of such content is unethical and should be avoided at all costs. Such an argument could have seemed convincing to some people before the internet age when offensive material was not widely available online, but we have a hard time seeing how such censorship would make sense today. We consider the benefit of increased knowledge about our past to outweigh any potential offense someone could take to the attitudes expressed in the newspapers.

## 1.5 Technical Requirements

To reproduce our results, roughly 50 hours on a high end Graphics Processing Unit (GPU) such as an Nvidia A100 80 GB are required. We rented the GPUs used throughout the project through Google Colab [7].

## 1.6 Availability of Models and Code

The fine-tuned VLMs developed in this project are publicly and freely available on Hugging Face[1,2]. Some of the developed code and the dataset partitioning are available on GitHub[3].

[1] `https://huggingface.co/J0hanski/Swe19centOCR-8B`
[2] `https://huggingface.co/J0hanski/Swe19centOCR-2B`
[3] `https://github.com/Martin31313/Swe19centOCR`

# 2

# Background

This chapter begins by describing some of the previous work done on OCR of historical texts. Then it describes how traditional OCR engines work. After that VLMs are discussed, including how they can be fine-tuned. Then the differences between the modern Antiqua typefaces and the historical Blackletter typefaces are discussed. Lastly, the evaluation metrics used to compare the performance of different systems in this thesis are described.

## 2.1 Related Work

In 2021, Dannélls et al. evaluated a two-OCR engine on a set of manually transcribed Swedish newspaper pages from 1818 to 2018 [8] (part of which is the dataset used in this project). Their two-OCR engine was developed in cooperation between Kungliga Biblioteket and the Norwegian company Zissor. The two OCR engines the hybrid system used were FineReader (proprietary) [3] and Tesseract (open-source) [2]. To combine the outputs of the two engines, a custom schema developed using a series of experiments done by Zissor was used. In addition to using the two engine hybrid, they also used a set of custom word lists for different time periods. Neither the hybrid system nor the use of custom word lists were found to improve the performance in general though. The best system for the time period 1818 to 1906 was found to be stock Tesseract without any external word lists. It outperformed all other configurations of systems and word lists in terms of both Character Error Rate (CER) at 11.86% and Word Error Rate (WER) at 18.74%.

To enhance the accuracy of transcriptions produced by OCR, a process known as post-OCR correction can be applied. A post-OCR correction system takes the OCR output as its input, and produces an improved version of the transcription. The process can be thought of as a translation from less correct to (hopefully) more correct language. Post-OCR correction has traditionally been done using dictionaries in combination with statistical techniques. In recent years, great results have been achieved using transformer-based systems. Löfgren did her master's thesis on this subject in 2023 [9]. She fine-tuned a post-OCR correction model based on the ByT5 byte-level Text-to-Text Transfer Transformer developed by Xue et al. (2021) at Google Research [10]. Löfgren trained the model on the material from Dannélls et al. (2021) [8], and was able to reduce both WER and CER across all data from 1818 to 2018, compared to baseline. Löfgren's result on the earlier half of the material,

mostly corresponding to the material used in this project, is shown in Table 2.1. These numbers are according to her report.

Table 2.1: The result of Löfgren (2023) [9] by fine-tuning a post-OCR correction model based on ByT5 with manually transcribed data from Dannélls et al. (2021) [8], excluding results for 20th century text. "Before" refers to WER and CER before applying the post-OCR correction model and "After" refers to after applying it. Löfgren's work was on the the whole period from 1818 until 2018.

| **Years** | **WER (%)** | | **CER (%)** | |
|---|---|---|---|---|
| | **Before** | **After** | **Before** | **After** |
| 1818–1859 | 32.46 | 15.34 | 8.39 | 4.39 |
| 1860–1899 | 16.51 | 8.06 | 4.04 | 2.29 |

Jasonarson et al. (2023) did a similar project as Löfgren, but at a much larger scale, for post-OCR correction of 19th century Icelandic material [11]. In addition to having roughly 50 times more manually annotated material, they also created artificial training data by randomly introducing errors in non-OCR material. Including the artificial material, they had roughly 1000 times the amount of training data of Löfgren. By fine-tuning a ByT5-base, with roughly twice the number of parameters of the ByT5-small that Löfgren used, and by both using the manually annotated material and the manually annotated material plus their artificial material, they achieved a similar proportional reduction in both WER and CER as Löfgren did (although from a lower baseline). The results showed that the artificial material did not improve the model over the one trained on just the manually annotated material. The fact that neither the slightly larger model nor the much larger dataset resulted in a significant improvement in error reduction in comparison to that of Löfgren [9] might indicate that the limit to what can be accomplished with post-OCR correction is near. Further significant reductions in error rates might be unfeasible with post-OCR correction and instead require improving the OCR system itself.

We have searched the internet for scientific papers and reports describing the utilization of VLMs for OCR of historical newspaper material in different languages, but have at the time of writing not found anything that is directly comparable to our project. This is not surprising giving how fast the progress of open source VLMs has been over the past couple of years.

## 2.2 Traditional OCR Engines

There are a multitude of OCR engines publicly available today. These range from commercial engines like FineReader [3] and Tungsten (formerly Kofax) OmniPage [12], to open-source tools like Tesseract [2] and OCRopus [13]. The commercial systems are often able to do the whole process from a scanned newspaper page to a PDF file with transcriptions overlayed automatically. Some of the open-source systems on the other hand require pre-processing before the transcription can be performed. Pre-processing can consist of de-skewing the image if it is tilted or warped in some way, converting it to grayscale, increasing the contrast, converting it

to binary black and white, and finally segmenting the page paragraph by paragraph. After that, the binarized paragraphs are fed to the transcription system one by one.

### 2.2.1 Architecture of Tesseract

One of the traditional OCR engines, Tesseract, is structured so that it first finds the textline of a text [2]. In the detected textlines, it searches for blobs containing words and letters. The blobs are separated letter-wise depending on whether the font is fixed pitch or not. If a blob has fixed pitch text, it is handled word by word and chopped character-wise. When a blob instead has proportional text, it is harder for the model to recognize spacing between words. Thus, word separation is done only after word recognition. Secondly, the model concludes its certainty of which characters there are in a blob. If the certainty is unsatisfactory, the blob undergoes chopping based of concave vertices found in the outline of the blob. In that way, even connected letters can be distinguished. If the certainty is unsatisfactory even after this, an evaluation is made whether ink is missing and causing broken letters.

## 2.3 Vision Language Models (VLMs)

The models used in this project are Vision Language Models (VLMs). The VLM is an extension of the Large Language Model (LLM). Simply stated, VLMs are generally made up of a vision encoder, an LLM, and an interface between them [14]. The vision encoder converts the input image into a set of tokens that are fed to the LLM. The LLM then generates text tokens, that hopefully match the user preference. VLMs are trained through gradient descent using very large datasets of inputs (image + prompt) and the corresponding preferred output text [14]. These datasets are created through a combination of content published on the internet and manual human work.

The LLM is based on the Transformer architecture, which was first introduced in 2017 by Vaswani, A., et al. [15]. It uses a multi-head attention mechanism, and recursively predicts the most probable next token based on all of the previous tokens in the sequence. Tokens are generated with query, key and value vectors, obtained by linear transformations [16]. Since the generation of tokens is dependent of previous tokens, key and value vectors are temporarily stored in something called KV cache. The attention between all tokens in a sequence can be computed in parallel [15]. This makes the Transformer scale much better with compute than older machine learning architectures such as recurrent neural networks.

Through the release of ChatGPT by OpenAI in late 2022, the LLM became one of the main focuses of machine learning research [17]. In early 2023, the VLM was popularized through the release of GPT-4 [18]. The fast public adaptation of these models led many major technology companies (e.g. Google, Meta, Microsoft) to develop their own LLMs and VLMs [17]. Some of these have been open sourced to the public. Common use cases for VLMs are tasks such as image classification [19], image captioning [20] [21], question answering about image content [22] [23] [24],

object localization in images [25], and what we are interested in, text transcription [26] [27] [28] [29].

The open source VLMs that are of interest for this project are the small to medium sized ones that can be run and fine-tuned on a single GPU. Examples of popular open source VLMs are the Gemma 3 and PaliGemma 2 series from Google and the Qwen3-VL and Qwen2.5-VL series from Alibaba Cloud. Both of these series are available in a multitude of sizes for different trade-offs between capability and efficiency.

The Qwen3-VL series is the latest generation of VLMs from Alibaba Cloud, and it is available in both dense and mixture-of-experts variants [30]. Dense means that parameters are active all the time, whereas mixture-of-experts mean that only a subset are active at a time. There are four dense versions with 2, 4, 8 and 32 billion parameters respectively and two mixture-of-experts versions with 30 and 235 billion parameters respectively.

One feature that differentiates the Qwen3-VL and it's two predecessor series Qwen2.5-VL [14] and Qwen2-VL [31] from most other open source VLMs is that they use a dynamic resolution vision encoder instead of a static resolution one. This means that the model gets to see the whole image without any distortions, which is not the case for most other models.

For example, the Google PaliGemma 2 is available with vision encoders with resolutions of $224 \times 224$, $448 \times 448$ and $896 \times 896$ pixels. Images are simply rescaled into the required resolution before they get processed [32]. The more recent Gemma 3 series also uses a fixed resolution of $896 \times 896$ pixels, but it uses an adaptive windowing algorithm for larger images, that simply cuts the large images into multiple smaller ones before feeding them to the vision encoder [33]. The LLaVA-1.5 (Large Language and Vision Assistant) that saw great success in early 2024 also uses a fixed resolution of $336 \times 336$ pixels [34]. As can be read about in Section 3.3, neither the PaliGemma2-3B-Mix-448 nor the LLaVA-1.5 made our 30% CER cut-off for further consideration.

A difference between traditional OCR engines and VLMs is that traditional OCR engines use a set of discrete steps while transcribing (as discussed in Section 2.2.1), while VLMs are a black box from start to end. When a traditional engine makes a mistake, one could in principle inspect each of the steps and find out where in the system the mistake was caused. This is only true to a certain extent though, since the subsystems are black boxes even in traditional engines. You might be able to determine that the mistake occurred in the line detection step, but it might still not be possible to explain why it happened.

## 2.4 Inference and the vLLM Inference Engine

In the machine learning community, using a model to generate outputs is commonly referred to as doing inference with it. Because of the large amount of parallelizable compute required to generate outputs, a Graphics Processing Unit (GPU) is typically

used while doing inference with VLMs. To efficiently run inference on a GPU, the prompts cannot be ran serially, because then only a small fraction of the GPU cores would be utilized at a time. Instead, the prompts need to be batched. It is however not trivial to batch, since it is not known beforehand how many tokens each prompt is going to generate, and therefore how much GPU memory they are going to occupy.

In many datasets (such as the one used in this project), the expected number of tokens generated varies widely across prompts. If a fixed batch size is used when such is the case, it must be chosen very conservatively in order to not run into out-of-memory crashes. Batching with a small batch size is of course more efficient than running the prompts serially, but it still leaves the GPU memory almost empty most of the time, with occasional spikes that almost fill the memory whenever some of the largest prompts are randomly paired together in the same batch. To address this issue, there are advanced inference engines that dynamically regulate the batch size to keep the GPU busy. One such inference engine is called vLLM.

vLLM was developed by W. Kwon et al. (2023) [16] and is a high-throughput inference engine that reduces memory load without losing accuracy, which enables faster processing of VLMs, regardless of the size of the model. vLLM is built so that it achieves near zero waste of KV cache memory, meaning it stores the obtained key and value vectors in a more efficient way and saves memory. vLLM makes use of an attention algorithm called PagedAttention, which stores KV cache in non-contiguous paged memory. According to Kwon et al., vLLM reaches two to four times the speed compared to previous state-of-the-art methods, including FasterTransformer and Orca.

Most of the VLMs utilized in this project are compatible with vLLM. By using vLLM, we were able to evaluate a large set of VLMs while limiting the use of time and resources. We would like to emphasize that the output given a specific model and prompt will be identical regardless if it is run through vLLM or not. The only difference is that the outputs are generated faster through a more efficient utilization of the hardware.

## 2.5 Methods for Fine-tuning of VLMs

Fine-tuning a model refers to starting from an existing model and training it further to be specifically adapted to ones particular use case. Fine-tuning can be done in multiple ways. In this section three methods are described.

### 2.5.1 Full Fine-tuning

The first one is full fine-tuning. Full fine-tuning means that all of the models parameters are updated. This enables radical alteration in the behavior of the model, but requires as much VRAM as the initial training of the model. For a VLM with 7 billion parameters this usually means a node of at least 2 Nvidia A100 80 GB GPUs, and for a 32 billion parameters a node of at least 6 Nvidia A100 80 GB GPUs. It also requires very large datasets to avoid overfitting.

### 2.5.2 Low Rank Adaptation

The second way of fine-tuning is Low Rank Adaptation (LoRA). LoRA was developed by Edward Hu et al. (2021) [35]. In LoRA, instead of updating the model weights themselves, you introduce adapter weights parallel with the original weights and train only the adapter weights. Let's call the original weight matrix $W_0$ with dimension $\mathbb{R}^{d\times k}$. The adapter matrix is then made up as a product of two weight matrices $B$ and $A$, where $B$ has dimension $\mathbb{R}^{d\times r}$ and $A$ has dimension $\mathbb{R}^{r\times k}$, where $r$ is the rank parameter normally chosen so that $r \ll min(d, k)$. The product $BA$ has the same dimension as $W_0$ and can simply be added (element wise) to the original weight matrix.

In an extreme example where you choose the rank to be $r = 1$, it reduces the number of trainable parameters from $d \cdot k$ to $d + k$. For a 7 billion parameter model, using a more realistic rank of $r = 8$, you only train around 8 million parameters instead of the full 7 billion you would train with a full fine-tuning. Only the gradients of the 8 million adapter parameters need to be stored, which greatly reduces the memory requirements, making it possible to fine-tune a 7 billion parameter VLM on a single Nividia A100 80 GB. It also makes it possible to fine-tune using a small dataset without overfitting. The downside of LoRA compared to full fine-tuning is that the possible impact of the fine-tuning is more constrained. For use cases where a complete change of model behavior is sought after, full fine-tuning is the preferred method.

Along with the rank, one usually introduces a second hyperparameter, a scaling parameter $\alpha$ which is a scalar multiplied with the adapter matrix $BA$ to linearly scale the impact of the fine-tuning. Altering $\alpha$ has almost the exact same effect as altering the learning rate with the added benefit of also being tunable after the fine-tuning has been done. The fine-tuned weight matrix is then given by

$$W_{FT} = W_0 + \frac{\alpha}{r}BA \tag{2.1}$$

### 2.5.3 Quantized Low Rank Adaptation

The third way of fine-tuning is Quantized Low Rank Adaptation (QLoRA). QLoRA is similar to regular LoRA, but instead of loading the model parameters with full 16-bit resolution, you quantize them to 4-bit resolution. This lowers the memory requirements by a factor of four, enabling the fine-tuning of models with 32 billion parameters on a single Nvidia A100 80 GB. The downside with QLoRA compared to regular LoRA is a potential performance reduction because of the lower resolution of the parameters.

## 2.6 Blackletter and Antiqua Typefaces

Our material mainly consists of two different categories of typefaces, namely Blackletter and Antiqua. Today, Antiqua typefaces are used almost exclusively, whereas

in the 19th century and earlier, Blackletter typefaces were also widely used. Blackletter typefaces is a quite diverse category, and it can be further decomposed into Fraktur and Schwabacher typefaces. In this thesis, we do not distinguish between Fraktur and Schwabacher, and exclusively refer to them as Blackletter fonts. The properties of Blackletter typefaces described in this section may not hold every single Blackletter font, but are general trends. Newspaper clippings exemplifying Blackletter and Antiqua can be seen in Figure 2.1.


Från porslinswurmens tid.

*Verkligheten är underbarare än dikten.*

I en förfallen gammal prestgård i Leicestershire hade i mera än 40 år stått ett par porslinsvaser — icke synnerligen beundrade eller uppskattade af egaren — hwilka under den tiden dammats af kanhända ett par tjog owarsamma pighänder och ändå icke slagits sönder. Huset war fullt med dylika saker af synbarligen lika eller större wärde och såg mera ut som ett porslinsmagasin än ett gammalt ärewördigt prestbus på landet.

Andra våningen,

i f. d. Bronanderska huset vid Gustaf Adolfs torg, bestående af sju stora och välbelägna rum med kök, stor tambur och dithörande skåp, badrum, serveringsrum, pigkammare, handkammare, garderober, skafferi, veranda, rymlig och välförsedd källare, egen vind samt del i stor torkvind och brygghus m. m., uthyres att tillträda den 1 April 1889.

Våningen, som ej är lämplig för återuthyrning, innehafves nu af hr Otto Magnus och kan beses, efter tillsägelse hos värden, från kl. 12 till 2.

(9899.) 3

Figure 2.1: Newspaper clippings from *Hvad Nytt*, 14th of June 1889 [36] and *Göteborgs Handels- och Sjöfartstidning*, 12th of November 1888 [37], showing different typefaces. The left image displays a heading and body text in Blackletter with a subheading in Antiqua, while the right image displays Antiqua.

### 2.6.1 Characters that Look Similar in Blackletter Typefaces

An issue with Blackletter typefaces is that there are pairs of letters that look visually similar. There are examples of this in Antiqua typefaces as well such as lowercase “l” and uppercase “I”, but these are usually easily distinguishable based on context. With Blackletter typefaces, this is a much larger issue. This subsection describes some of the most obvious cases.

Figure 2.2 shows the non-existent difference between “I” and “J” in Blackletter. They need to be distinguished purely based on context. Figure 2.3 displays the similarities between “n” and “u”, and shows how they can vary even within the same paragraph. Even if the entire paragraph is in the same condition, the letters vary greatly. Depending on the exact placement of ink, letters can lose their characteristics. In this case, “n” in the left image resemble “u” in the right image.

Figure 2.4 shows examples of characters that many OCR engines interpret as “S”, when in reality they represent “E” and “G”. Another common incorrect prediction by OCR engines is related to the letter “f”. The letter “k” as well as a letter that is not in use anymore, “f”, long s, look similar to “f”. Examples of this confusion can be found in Figure 2.5, where the text in the left image could easily be mistaken for “få folftoma”, compared to its ground truth, which is “så folktoma”, and the right image could be mistaken for “tillbafa.” and “fejor”, instead of its ground truth, “tillbaka.” and “kejor”.

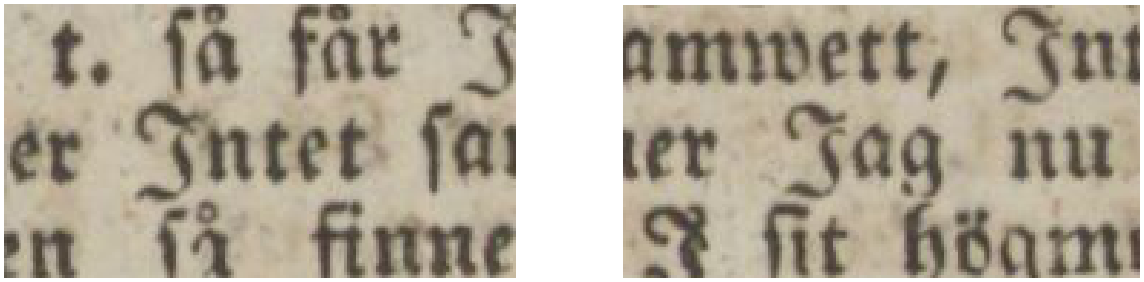


Figure 2.2: Newspaper clippings from *Dagligt Allehanda*, 28th of April 1829 [38], displaying the similarity between letters "I" and "J". The left image illustrates the word "Intet" and the right image illustrates "Jag".

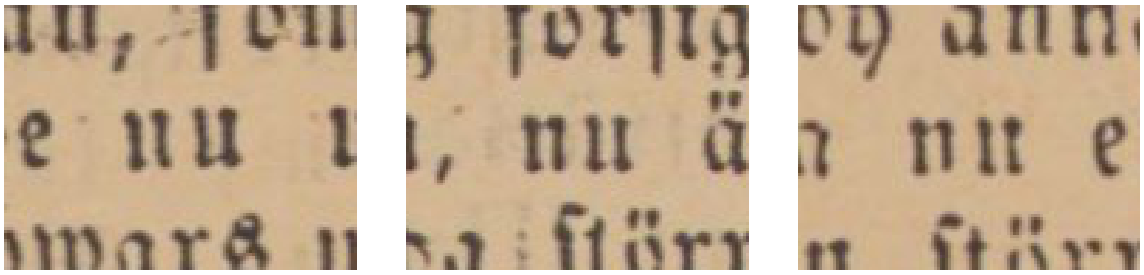


Figure 2.3: Newspaper clippings from *Karlshamns Allehanda*, 5th of July 1873 [39], showing how the word "nu" vary in appearance inside the same paragraph. The left image clearly shows the letter "u", while the clarity decreases with the middle and right image. The image on the left has the weakest representation of letter "n", making it very similar to letter "u" on the right.

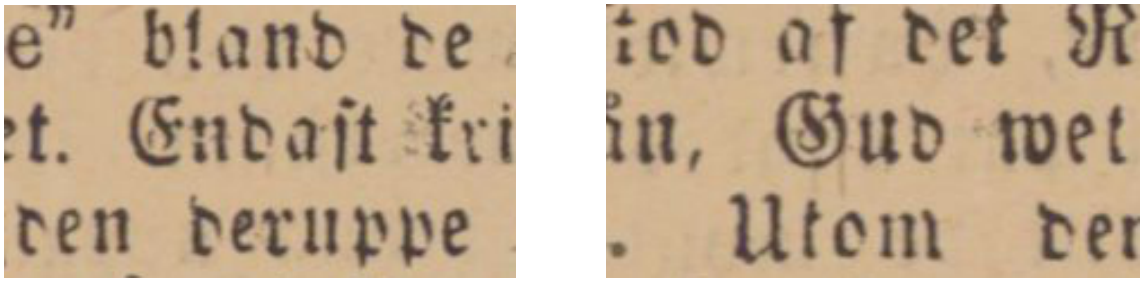


Figure 2.4: Newspaper clippings from *Karlshamns Allehanda*, 5th of July 1873 [39], showing the resemblance between letters. The left image presents the word "Endaft", or as in our ground truth "Endast", and the right image "Gud".

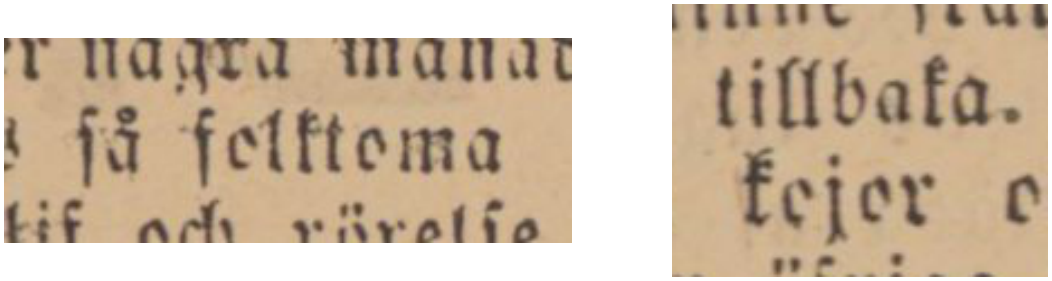


Figure 2.5: Newspaper clippings from *Karlshamns Allehanda*, 5th of July 1873 [39], showing the resemblance between letters. The left image shows "få folktoma" or as in ground truth "så folktoma". The right image shows "tillbaka." and "kejor". In some Blackletter typefaces, the letters "f", "f" and "k" all look similar to "f" in Antiqua.

## 2.7 Evaluation Metrics

We have chosen five metrics to compare the performance of different OCR systems, namely Character Error Rate (CER) and Word Error Rate (WER), Precision, Recall and F-score, where the last three are computed on the word level. The correlations between the last four mentioned metrics are in general high for well performing systems, but we choose to present all of them in the main results, Table 4.1. For the other evaluations, we present only CER and WER, or even exclusively CER, for the sake of brevity. This subsection describes how these metrics are computed using relevant formulae and examples to give the unfamiliar reader some intuition. To illustrate the computation of accuracy, an excerpt from *Stockholms Dagblad* with its ground truth and an arbitrary OCR output (in this case generated by the Qwen2.5-VL-7B-Instruct from Alibaba Cloud [14]) is used and presented in Table 2.2. Also notice that this ground truth most likely contains an error. "I dag" means "Today", whereas "J dag" does not make any sense. There are errors like this one in the ground truth because "J" and "I" look identical in some Blackletter typefaces (as discussed in Section 2.6.1).

Table 2.2: An excerpt from *Stockholms Dagblad*, 27th of November 1826 [4], with its corresponding prediction using Qwen2.5-VL-7B-Instruct. Ground truth was transcribed by Grepect GmbH [5]. Errors made in the output are marked red. "å" has been marked blue as it was not predicted by the model.

| | |
|---|---|
| **Ground truth** | J dag Måndag den 27:de dennes, f. m. från kl. 10, förläljes å denne Stadens Auktions=Kammare |
| **OCR output** | I dag Måndag den 27:de dennes, f. m. från kl. 10, föråfåles dessa Stadens Auktions-Kammare |

### 2.7.1 Word Error Rate (WER) and Character Error Rate (CER)

WER is computed as

$$WER = \frac{S_w + D_w + I_w}{N_w} \tag{2.2}$$

where $S_w$, $D_w$ and $I_w$ are the minimum number of word level substitutions, deletions and insertions required to transform the prediction into the ground truth (or vice versa), as computed by the Levenshtein distance. $N_w$ is the number of words in the ground truth.

Character Error Rate is the corresponding metric on the character level and it is computed as

$$CER = \frac{S_c + D_c + I_c}{N_c} \tag{2.3}$$

where $S_c$, $D_c$ and $I_c$ are the minimum number of character level substitutions, deletions and insertions required to transform the prediction into the ground truth (or vice versa), as computed by the Levenshtein distance. $N_w$ is the number of characters in the ground truth. When computing the CER, spaces are counted the same way as any other character.

Table 2.3: Substitutions, deletions and insertions found in Table 2.2 on word level. The errors made in the prediction are marked red and the corresponding correct words in the ground truth are green.

| Substitutions | | | Deletions | Insertions |
|---|---|---|---|---|
| J | → | I | å → “” | |
| förläljes | → | föråfåles | | |
| denne | → | dessa | | |
| Auktions=Kammare | → | Auktions-Kammare | | |
| $S_w = 4$ | | | $D_w = 1$ | $I_w = 0$ |

Table 2.4: Substitutions, deletions and insertions found in Table 2.2 on character level. The errors made in the prediction are marked red and the corresponding correct letters in the ground truth are highlighted as green.

| Substitutions | | | Deletions | | | Insertions |
|---|---|---|---|---|---|---|
| J | → | I | å | → | “” | |
| förläljes | → | föråfåles | “ ” | → | “” | |
| denne | → | dessa | | | | |
| Auktions=Kammare | → | Auktions-Kammare | | | | |
| $S_c = 9$ | | | $D_c = 2$ | | | $I_c = 0$ |

Using Table 2.3-2.4 and counting the total number of words and characters in ground truth, WER and CER are computed as

$$WER = \frac{4+1+0}{16} = \frac{5}{16} \approx 0.313 \tag{2.4}$$

and

$$CER = \frac{9+2+0}{92} = \frac{11}{92} \approx 0.120 \tag{2.5}$$

CER being significantly lower than WER is not just a quirk of this particular example. It is generally the case.

When computing the WER and CER over multiple text segments, the contribution of each segment is weighted by the length of the segment, and then the sum is divided by the total length of all segments. This is critical, since each error should be weighed equally no matter which segment it appears in. If one were to simply average across all error rates, one would overweigh errors in shorter segments and underweight errors in longer ones.

### 2.7.2 Precision, Recall and F-score

A higher Precision, Recall and F-score indicate a better performance of a model. They range from $0-1$, or $0-100\%$. Precision and Recall are computed as

$$Precision = \frac{TP}{TP+FP} \qquad Recall = \frac{TP}{TP+FN} \tag{2.6}$$

where $TP$ is True Positives, $FP$ is False Positives and $FN$ is False Negatives. $TP$ represents the number of correctly predicted words, $FP$ represents words in the prediction but not in ground truth, and $FN$ is the number of words in ground truth that are not in the prediction. F-score is the harmonic mean of Precision and Recall, which is computed as

$$F_1 = \frac{2}{Recall^{-1} + Precision^{-1}} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{2.7}$$

Using Table 2.2, we can easily find $TP$, $FP$ and $FN$. In this case, the number of correctly predicted words is $TP = 11$. $FP$ represents words in the prediction but not in ground truth, which is "I", "föråfåles", "dessa" and "Auktions-Kammare", giving us $FP = 4$. Lastly, $FN$ represents words in ground truth but not in the prediction, which is "J", "förläljes", "å", "dessa" and "Auktions=Kammare" and gives $FN = 5$. This results in

$$Precision = \frac{11}{11 + 4} = \frac{11}{15} \approx 0.733 \qquad Recall = \frac{11}{11 + 5} = \frac{11}{16} \approx 0.688 \tag{2.8}$$

$$F_1 = \frac{2 \cdot 11}{2 \cdot 11 + 4 + 5} = \frac{22}{31} \approx 0.710 \tag{2.9}$$

# 3

# Methodology

This chapter describes the approach of this project. It discusses the properties of the dataset and its segmentation, inference. Then it establishes a baseline performance of traditional OCR engines. Thereafter it describes how a set of stock VLMs were evaluated, which models that were fine-tuned, and how the fine-tuning was performed. Lastly it is described how a repetition trimmer function was designed.

## 3.1 Dataset

Dannélls et al. (2021) [8] crafted a dataset of Swedish newspapers between the years 1818 and 2018 for an evaluation of an OCR engine they developed. The dataset consists of two pages from a newspaper from each year. The newspapers were carefully selected to be representative of the whole set of Swedish newspapers throughout the time period in terms of variations in layout and typography. For most of the selected newspapers, the second and fourth pages were chosen because it was found that there were less advertisements and pictures on those pages than on other ones.

The pages were then segmented roughly speaking paragraph-wise. The segmentation was done by an automatic system, making some of the segments empty. The segments were manually transcribed by the German company Grepect GmbH [5] using the double keying method to minimize the number of errors. Double keying means that two people independently transcribe the text, and any disagreement between the transcriptions are reviewed afterwards. We were provided two versions of the ground truth transcriptions. One version where "f", the long-s (see Section 2.6) was transcribed as is, and one version where "f" was replaced by "s". We chose to always use the version where "f" was replaced by "s", since most OCR systems (except Tesseract, see Section 4.1.5) usually do this substitution.

The dataset used in this thesis is the subset of Dannélls' dataset between the years 1818 and 1904, resulting in 174 newspaper pages. The dataset, in addition to newspaper pages from years 1905 and 1906 and some metadata, is publicly available from Språkbanken Text[1,2].

[1] `https://spraakbanken.gu.se/en/resources/svenska-tidningar-1818-1870`
[2] `https://spraakbanken.gu.se/en/resources/svenska-tidningar-1871-1906`

The dataset is provided in the form of layered PDF files, where the first layer contains the image of the page itself, and the rest of the layers each mark a segment for ease of extraction. We were also provided with a Python script for extracting the segments of a page, and saving them as image files.

Figure 3.1 shows a selection of newspaper pages with their corresponding segmentation. The top left image exemplifies the layout of many of the earlier newspapers, while the bottom two images exemplifies the layout of later ones. The upper pages displayed in Figure 3.1 have 16 and 27 segments, while the lower pages have 126 and 351 segments. As can be seen in the images, illustrations in the text are sometimes marked as segments, and other times not. Illustrations that have been marked as segments have an empty ground truth file.

### 3.1.1 Properties of the Dataset

A description of the typefaces Blackletter and Antiqua can be found in Section 2.6. The division of typefaces throughout the years can be seen in Figure 3.2. The use of Blackletter declines over time, while Antiqua increases over time. Before 1832, all of the newspapers in our dataset are in Blackletter, and after 1890 all of them are in Antiqua. Many of the newspapers use both kinds of typefaces, varying from only a few headlines in Antiqua and the rest of the text in Blackletter, to having half of the total text in Antiqua and the other half Blackletter. Newspapers marked as "Both" in Figure 3.2 use each kind of typeface to a substantial degree, meaning not just headlines.

The number of segments and the number of characters in each newspaper can be seen in Figure 3.3. Even though segments vary greatly in size, it is clear from the figure that there is a strong correlation between the number of segments and the number of characters in a newspaper. It is also clear that the amount of text in each newspaper is increasing over time. This is because of the rise of larger pages and more advanced printing technology. The numbers of segments and characters of each typeface are given in Table 3.1. It is clear that the majority of the text is printed in Antiqua.

Table 3.1: Typeface breakdown showing how many segments and characters there are of each typeface. Newspapers containing both typefaces across multiple segments are labeled "Both". Newspapers labeled "Blackletter" sometimes contain headlines in Antiqua, but not more than that.

| **Typeface** | **Number of Segments** | **Number of Characters** |
|---|---|---|
| Blackletter | 4,048 | 894,577 |
| Antiqua | 10,269 | 1,716,752 |
| Both | 2,899 | 538,531 |

Along with the dataset, we were also provided with a manual content classification, where each segment is classified either as "Text", "Image" or "List". The division can be seen in Table 3.2. Only a small fraction of the segments contains formats other than regular text.

Figure 3.1: A selection of newspaper pages with corresponding segmentation. The top left image is from *Wexjöbladet*, 18th of September 1835 [40], the top right image is from *Umebladet*, 14th of December 1861 [41], the bottom left image from *Hallandsposten*, 25th of February 1891 [42], and the bottom right image from *Göteborgs Aftonblad*, 17th of October 1995 [43].

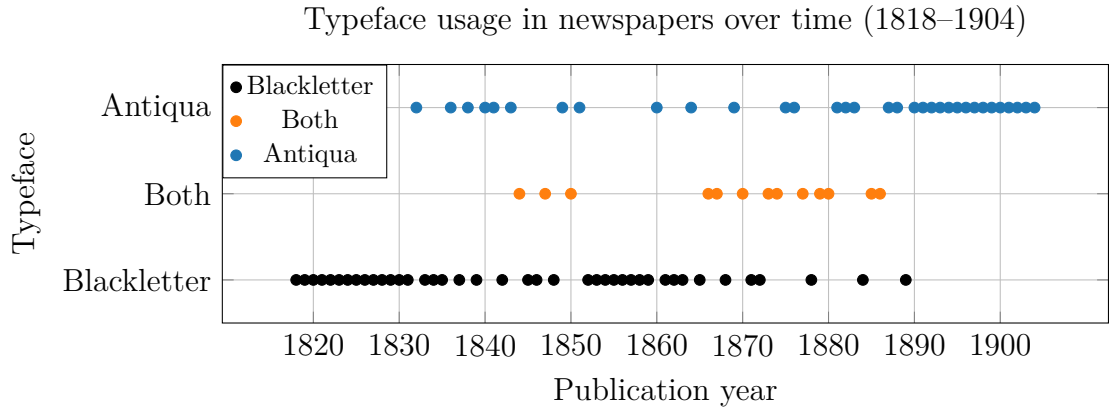



Figure 3.2: Typeface usage in the newspapers in our dataset over time ranging from 1818 to 1904. Newspapers are labeled "Both" if both typefaces are used in multiple segments. Some of the newspapers labeled "Blackletter" do have headlines in Antiqua, but newspapers with more than just headlines in the minority typeface are labeled "Both". Out of the 87 newspapers, 41 are labeled "Blackletter", 33 are labeled "Antiqua", and 13 are labeled "Both".

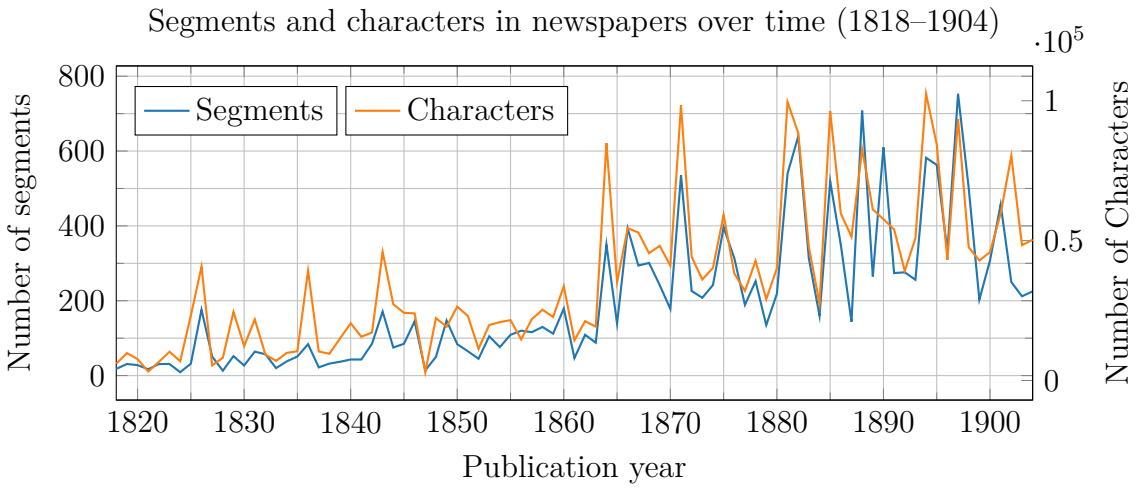



Figure 3.3: Combined visualization of number of segments and characters in newspaper pages from 1818 to 1904, according to our dataset. Number of segments is shown on the left axis. The segments are of varying size and surrounds the paragraphs of a newspaper page. A more complex layout generally generates more segments. Number of characters is shown on the right axis. Even though the segments vary in length, there is a strong correlation between the number of segments and the number of characters in a newspaper.

Table 3.2: Division between "Text", "Images" and "Lists" in the different segments of the dataset. Lists include tables.

| **Type** | **Number of Segments** | **Prop. (%)** |
|---|---|---|
| Text | 16,791 | 97.531 |
| Images | 191 | 1.109 |
| Lists | 234 | 1.359 |
| Total | 17,216 | |

### 3.1.2 Fixing Errors in the Ground Truth

Eleven of the segments from the dataset lacked a ground truth file, and were therefore excluded from the project. Most of these segments contained little or no text, and could be considered mistakes of the automatic segmentation system.

In order to examine the quality of the ground truth, we ran the whole dataset through a VLM called Qwen2.5-VL-7B-Instruct and manually inspected the 10% of the segments that gave the highest error rates. The vast majority of the high error rates were simply due to OCR engine mistakes, but we also found roughly 150 segments where there were substantial errors or inconsistencies in the ground truth. These errors ranged from words in the ground truth not being inside of the image (falling outside of the crop), to there being a complete mismatch between the image and the ground truth (ground truth corresponding to another image). A few segments (that had text in the image) even lacked a ground truth text all together. Luckily, most of these issues were fixable, but four segments were deemed too hard to fix and therefore excluded from the project. This left us with 17,201 segments in total.

Apart from the substantial errors, some amount of single letter errors were also observed during the manual inspection. Most of them were caused by the visual similarity between some pairs of letters in Blackletter typefaces, as discussed in Section 2.6.1. These were left unfixed.

### 3.1.3 Partitioning of the Dataset

The dataset was divided completely randomly segment-wise, where 70% of the data were used for training, 15% for validation and 15% for testing. The training set was used for the fine-tuning itself, the validation set was used to determine when to stop the fine-tuning, and the test set was used for the final evaluation of the performance of all models. The test set consists of 2,518 texts, 30 images and 35 lists, which constitutes a similar proportion of images and lists as the full dataset. The exact partitioning used in the project is available on Github[3] in three separate text files.

## 3.2 Baseline of Traditional OCR Engines

In order to determine the viability of our VLM based OCR systems, we compare them to the best traditional OCR engines we have access to. We were provided the OCR outputs from the proprietary Abbyy FineReader 11.1.16 (released in 2011) [44] and the open source Tesseract 4.0.0 (released in 2018) [45] that were both ran for Dannélls et al. (2021) [8]. Since Tesseract is open source (see Section 2.2.1), we also ran the test set through the latest version (at the time) ourselves, namely version 5.5.1 released in May 2025. We tried both using a combination of the Swedish and German Fraktur language packages, and using exclusively the Swedish language package. German Fraktur is the only Blackletter language package available for Tesseract.

[3] `https://github.com/Martin31313/Swe19centOCR/tree/main/partitioning`

Even though FineReader is a subscription service, they do offer a free trial that lets you run up to 100 images for free. We ran the earliest 100 segments from the test set through FineReader PDF 16, which is the latest version at the time of writing. FineReader PDF 16 performed much worse than FineReader 11.1.16, achieving a CER of 42.200 % in comparison with 20.065 % on these same 100 segments. Because of the poor performance, we decided against purchasing a subscription to run the rest of the test set. It should be noted that almost all of these 100 segments are in Blackletter typefaces, and we cannot rule out the possibility of FineReader PDF 16 performing better on Antiqua.

The outputs generated by the traditional OCR engines were evaluated against the ground truth using the metrics described in Section 2.7. The CER and WER of each system are listed in Table 3.3. As can be seen in the table, FineReader 11.1.16 performed the best and will therefore be considered our main baseline. Out of the Tesseract versions, version 5.5.1 with both the Swedish and the German Fraktur language packages performed the best. Since the FineReader is a proprietary system, we will compare our VLM based systems to both the FineReader 11.1.16, and the Tesseract 5.5.1 with the dual language packages.

Table 3.3: Comparison of the CERs and WERs achieved by the traditional OCR engines.

| **Model** | **CER (%)** | **WER (%)** |
|---|---|---|
| FineReader 11.1.16 | 6.123 | 19.888 |
| Tesseract 5.5.1 (Swe+Frk) | 7.089 | 21.882 |
| Tesseract 5.5.1 (Swe) | 8.065 | 24.410 |
| Tesseract 4.0.0 | 8.139 | 28.559 |

We also considered including a traditional engine in combination with the post-processing system developed by Löfgren [9] in 2023 described in Section 2.1 in the baseline. This could be done in two different ways. Either we could compare the performance of our models to the performance metrics reported in her thesis, or we could run her model ourselves on FineReader 11.1.16.

The first option has three main issues. The first and most important issue is that it would be somewhat of an “apples to oranges” comparison, because even though she used the same dataset as us, she seems to have excluded parts of it in her pre-processing setup while aligning the inputs to the ground truth. Her baseline is lower than that of FineReader 11.1.16, which suggests that the more difficult parts of the material were excluded at a higher rate than the less difficult parts. The second issue is that we do not use the exact same performance metrics as she did, and it is not trivial to convert them without access to the material itself. The third issue is that we could not do an in depth comparison of the strengths and weaknesses of different models without access to the actual outputs of the system.

The second option, to run her model ourselves, comes with another set of issues. Even though her model is open-sourced, her pre-processing setup is not. Her model only accepts up to 128 characters at a time, meaning that we would have to partition all segments longer than that. We attempted to do this in the most naive

way, with unsatisfactory results. Another issue is that we do not know her training/validation/test split, which means that we would end up evaluating the performance of her model on the exact same material that was used to train it.

With all of this in mind, we decided to just compare our VLM based systems to FineReader 11.1.16 and Tesseract 5.5.1. It should however be noted that Löfgren's system does outperform these baselines given the right circumstances.

## 3.3 Running Inference with Stock VLMs

We ran our test set through most open-source VLMs we could find that satisfy the following criteria:

1. Are supported by the vLLM inference engine (see Section 2.4),
2. Have at least some level of community adaptation as measured by number of downloads from HuggingFace,
3. Are not explicitly stated to not be ready to follow instructions,
4. Have at most 32 billion parameters,
5. Are able to run without any major changes to the inference template provided by its developers.

A notable exception to this is Florence-2-large, which was tested even though it is not supported by vLLM because of its small size and good reputation.

The full list of evaluated models is given in Table 3.4. Most models were run with the simple instruction text "Transcribe the text exactly as it appears in the image. Do not write anything but the actual transcription!". Some models were run with a different instruction such as for example "<OCR>", in accordance with recommendations from their developers.

In total, it took roughly 20 hours to run all the models using the vLLM inference engine. For the smaller models an Nvidia L4 GPU was used, and for the larger ones an Nvidia A100 GPU was used. We did however later find out that the inference engine often was unnecessarily constrained by too low of a maximum batch size (we used the settings recommended in the vLLM inference example code [46]). Setting the maximum batch size to "None" allows the engine to freely choose the optimal batch sizes, which make the same inference able to be run in less than half that time.

All models were evaluated using the evaluation metrics described in Section 2.7. The models marked with (*) in the table were not able to achieve a CER below 30% and were therefore excluded from further consideration. The reasons for the poor performances are varied. Some of the models might be developed strictly with object detection or image captioning in mind. Some might lack the visual input resolution required to be able to distinguish the individual letters in longer paragraphs. We also cannot rule out the possibility that some of these models could expect a different

Table 3.4: List of all evaluated stock models including their developer and the number of parameters each model has expressed in billions (B), sorted after the number of parameters in the models. The models marked with (*) performed very poorly (CER above 30%).

| Developer | Model | Parameters |
|---|---|---|
| Microsoft | Florence-2-large | 0.8 B |
| Alibaba Cloud | Qwen2-VL-2B-Instruct | 2 B |
| OpenGVLab | InternVL3-2B | 2 B |
| OpenGVLab | (*) InternVL3.5-2B | 2 B |
| Alibaba Cloud | Qwen3-VL-2B-Instruct | 2 B |
| HyperClovax | (*) SEED-Vision-Instruct-3B | 3 B |
| Alibaba Cloud | Qwen2.5-Omni-3B | 3 B |
| DeepSeek | (*) DeepSeek-VL2-Tiny | 3 B |
| rednote-hilab | (*) dots.ocr | 3 B |
| Google DeepMind | (*) PaliGemma2-3B-Mix-448 | 3 B |
| Alibaba Cloud | Qwen2.5-VL-3B-Instruct | 3 B |
| Google DeepMind | Gemma3-4B-IT | 4 B |
| OpenGVLab | InternVL3.5-4B | 4 B |
| Alibaba Cloud | Qwen3-VL-4B-Instruct | 4 B |
| Microsoft | Phi-4-multimodal-instruct | 6 B |
| omni-research | (*) Tarsier-7b | 7 B |
| Allen AI | (*) Molmo-7B-D | 7 B |
| Llava Hugging Face | (*) Llava-1.5-7B | 7 B |
| Meta Research | (*) Chameleon 7B | 7 B |
| Alibaba Cloud | Qwen2.5-VL-7B-Instruct | 7 B |
| Alibaba Cloud | Qwen2.5-Omni-7B | 7 B |
| Alibaba Cloud | Qwen2-VL-7B-Instruct | 7 B |
| Tiger-Lab | (*) Mantis-Idefics2 | 8 B |
| Alibaba Cloud | Qwen3-VL-8B-Instruct | 8 B |
| Adept AI Labs | (*) Fuyu-8B | 8 B |
| OpenGVLab | InternVL3-8B | 8 B |
| HuggingFaceM4 | (*) Idefics3-8B-Llama3 | 8 B |
| OpenGVLab | InternVL3.5-8B | 8 B |
| OpenBMB | MiniCPM-o 2.6 | 9 B |
| Nvidia | Llama-3.1 Nemotron Nano VL | 9 B |
| Google DeepMind | Gemma3-12B-IT | 12 B |
| Mistral Community | Pixtral-12B | 12 B |
| OpenGVLab | (*) InternVL3-14B | 14 B |
| OpenGVLab | InternVL3.5-14B | 14 B |
| Rhymes AI | (*) Aria | 25 B |
| Google DeepMind | Gemma3-27B-IT | 27 B |
| Baidu | (*) Ernie-4.5-VL-28B-A3B | 28 B |
| Alibaba Cloud | Qwen3-VL-30B-A3B-Instruct | 30 B |
| Alibaba Cloud | Qwen2.5-VL-32B-Instruct | 32 B |
| Alibaba Cloud | Qwen3-VL-32B-Instruct | 32 B |

Table 3.5: Comparison of the CERs and WERs achieved by the stock models which achieved a CER of less than 30%. The models are sorted from lowest to highest CER.

| Model | Parameters | CER (%) | WER (%) |
|---|---|---|---|
| Qwen2.5-VL-7B-Instruct | 7 B | 3.725 | 13.994 |
| Qwen3-VL-8B-Instruct | 8 B | 4.539 | 15.830 |
| Qwen3-VL-30B-A3B-Instruct | 30 B | 4.769 | 16.778 |
| Qwen2.5-Omni-7B | 7 B | 5.108 | 15.151 |
| Qwen3-VL-32B-Instruct | 32 B | 5.934 | 18.378 |
| Qwen2.5-VL-3B-Instruct | 3 B | 6.059 | 20.422 |
| Gemma3-27B-IT | 27 B | 7.850 | 16.673 |
| Qwen3-VL-4B-Instruct | 4 B | 8.138 | 23.120 |
| Qwen2.5-Omni-3B | 3 B | 8.170 | 21.813 |
| Qwen2.5-VL-32B-Instruct | 32 B | 9.459 | 25.340 |
| Qwen2-VL-7B-Instruct | 7 B | 10.022 | 25.764 |
| Llama-3.1 Nemotron Nano VL | 9 B | 10.911 | 29.006 |
| Gemma3-12B-IT | 12 B | 10.950 | 22.332 |
| Qwen3-VL-2B-Instruct | 2 B | 13.419 | 26.761 |
| InternVL3-8B | 8 B | 15.816 | 33.269 |
| MiniCPM-o 2.6 | 9 B | 16.269 | 39.922 |
| Florence-2-large | 0.8 B | 17.266 | 43.996 |
| InternVL3.5-14B | 14 B | 19.567 | 45.968 |
| Pixtral-12B | 12 B | 20.505 | 44.460 |
| InternVL3.5-4B | 4 B | 21.523 | 46.089 |
| InternVL3.5-8B | 8 B | 24.087 | 43.891 |
| InternVL3-2B | 2 B | 24.154 | 50.729 |
| Phi-4-multimodal-instruct | 6 B | 26.036 | 51.804 |
| Qwen2-VL-2B-Instruct | 2 B | 27.377 | 43.386 |
| Gemma3-4B-IT | 4 B | 27.553 | 44.125 |

prompt format than the one we used. Either way, these models were not considered further, but are still included here for the sake of completeness.

Table 3.5 lists the CER and WER of the models that were able to achieve a CER below 30%. The first thing to notice is the total dominance of the Alibaba Cloud Qwen models. The six best performing models are all Qwen models of varying generations, sizes and architectures. The best performing non-Qwen model we tried is the Google DeepMind Gemma3-27B-IT, but it is still beat in terms of CER by the 9 times smaller Qwen2.5-VL-3B-Instruct.

We suspect that the dominance of the Qwen models is because of their dynamic resolution vision encoders described in Section 2.3. Most other models have a square, fixed resolution vision encoder. A lot of our segments are far from square, including wide single row segments and tall long paragraph segments. It does not seem surprising that models with a fixed resolution struggle a lot with such segments, greatly impacting their overall error rates. However, we have not analyzed if the worse performing models actually perform better when limited to segments that are close to

square. We want a model that is able to perform well regardless of segment aspect ratio, making the exact reason for these models' poor performance not important enough for further investigation in this project.

Another thing to notice is the relatively weak relationship between model size and performance. Among the Qwen models, the mid sized Qwen2.5-VL-7B-Instruct and Qwen3-VL-8B-Instruct both outperform their 32B counterparts, and the InternVL3-8B outperforms its 14B counterpart. Three of the eight largest models were not even able to make the 30% CER cutoff to be included in Table 3.5.

## 3.4 Fine-tuning of VLMs

Some models stand out as candidates for fine-tuning. The first and most obvious one is Qwen2.5-VL-7B-Instruct, since it is the best performing stock model. Qwen3-VL-8B-Instruct and Qwen-2.5-Omni-7B also stand out as top tier models of roughly the same size. Qwen3-VL-30B-A3B-Instruct and Qwen3-VL-32B-Instruct also made it to the top five, but since they are much larger than their smaller (and better performing) siblings, their fine-tuning cannot be justified.

Another standout is Qwen2.5-VL-3B-Instruct, which is the best performing model with less than 7 billion parameters. Qwen3-VL-4B-Instruct and Qwen2.5-Omni-3B also stand out as well performing models at a similar size. Among the models with less than 3 billion parameters, Qwen3-VL-2B-Instruct and Florence-2-large stand out as the only two models with a CER below 20%, making them fine-tuning candidates when computational efficiency is prioritized above maximal accuracy. The remaining models are too weak, too large, or both, to justify their fine-tuning.

This left us with the fine-tuning candidates listed in Table 3.6. In the end, we did not end up fine-tuning all of these models. As described in the following subsections, we began by fine-tuning a Qwen3-VL-2B-Instruct to gain some experience with a small model before approaching the fine-tuning of larger ones. Next, we decided between fine-tuning either one of the 3-4 billion parameter models or one of the 7-8 billion parameter ones. Since the fine-tuning of the first model went so well, we decided to go straight for the larger category. After an unsuccessful attempt at fine-tuning a Qwen2.5-VL-7B-Instruct, we ended up fine-tuning two Qwen3-VL-8B-Instructs, one unsuccessfully and one successfully. Because of the relatively small performance gap between our fine-tuned Qwen3-VL-2B-Instruct and our fine-tuned Qwen3-VL-8B-Instruct, we thereafter decided against fine-tuning any of the 3-4 billion parameter models. Finally, we fine-tuned a Florence-2-large with limited success.

Since our dataset is on the smaller end of the spectrum, it is not suitable for full fine-tuning (see Section 2.5.1). That left us with the choice between regular LoRA and QLoRA (see Section 2.5.2 and 2.5.3). Since all of our candidate models were small enough to be fine-tuneable with regular LoRA on a single Nvidia A100 80GB, we decided against using QLoRA because of its potential negative effects on the performance. In order to sustain the models' ability to generalize well, we froze the vision encoding parts of the models, and only introduced LoRA adapters for the language parts of the models (except for the Florence-2-large where the entire

Table 3.6: List of the stock models which were considered for fine-tuning. The models are sorted from lowest to highest CER.

| Model | Parameters | CER (%) | WER (%) |
|---|---|---|---|
| Qwen2.5-VL-7B-Instruct | 7 B | 3.725 | 13.994 |
| Qwen3-VL-8B-Instruct | 8 B | 4.539 | 15.830 |
| Qwen2.5-Omni-7B | 7 B | 5.108 | 15.151 |
| Qwen2.5-VL-3B-Instruct | 3 B | 6.059 | 20.422 |
| Qwen3-VL-4B-Instruct | 4 B | 8.138 | 23.120 |
| Qwen2.5-Omni-3B | 3 B | 8.170 | 21.813 |
| Qwen3-VL-2B-Instruct | 2 B | 13.419 | 26.761 |
| Florence-2-large | 0.8 B | 17.266 | 43.996 |

model including the vision encoder were fine-tuned). The models were fine-tuned one epoch at a time, manually lowering the learning rate by 30-50 % by feel upon display of instability (regression of performance) in the training. The code used to fine-tune the Qwen models is available on Github[4].

Because of the sensitive nature of the task, training and validation loss could not be utilized to supervise the progress of the training. After an initial sharp decrease throughout the first few epochs, the losses stay mostly constant. Instead, we ran inference with the models on the validation set after each epoch in order to supervise the progress.

### 3.4.1 Fine-tuning of Qwen3-VL-2B-Instruct

To gain experience in fine-tuning VLMs without wasting resources, we decided to start off by fine-tuning a smaller model, namely the Qwen3-VL-2B-Instruct. The first step for the fine-tuning is to choose the hyper-parameters. Since the tests done by Hu et al. (2021) [35] indicates that the relation between LoRA-rank and model performance is weak (at least for some tasks), we decided to use a rank on the lower end of the spectrum, namely 8. Since $\alpha$ is just a scaling parameter and varying it mostly does the same thing as varying the learning rate, $\alpha$ was also set to 8 for simplicity. The dropout rate was set to a standard value of 0.1 for some regularization, and weight decay was not used.

To determine a suitable learning rate and effective batch size, some experimental runs were done with a small subset (5 % and 20 %) of the training set. Effective batch size is the number of segments that are used for each parameter update. An initial learning rate of $10^{-5}$ and and an effective batch size of 64 were found suitable.

Figure 3.4 shows the progress of the fine-tuning in terms of CER and WER achieved on the validation set after each epoch. The initial four epochs used a learning rate of $10^{-5}$, epochs 5 to 17 used a learning rate of $5 \cdot 10^{-6}$, epochs 18 to 22 used a learning rate of $3 \cdot 10^{-6}$, epochs 23 to 28 used a learning rate of $2 \cdot 10^{-6}$, and the last three epochs used a learning rate of $10^{-6}$. After epoch 31, the fine-tuning was terminated since no progress had been made in the last six epochs. Epoch 25 was chosen as our

[4] `https://github.com/Martin31313/Swe19centOCR/blob/main/finetune-one-epoch.py`

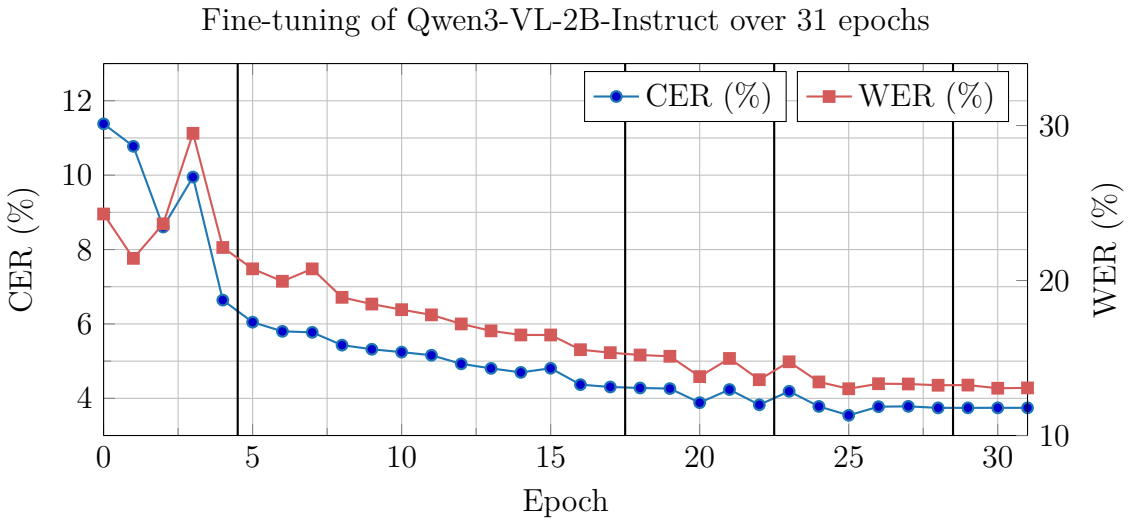



Figure 3.4: CER and WER achieved by each epoch on the validation set during the fine-tuning of the Qwen3-VL-2B-Instruct. Epoch 0 is the stock model. The vertical bars indicate where the learning rate was lowered.

final model since it achieved the lowest CER and WER on the validation set. The model is publicly available on Hugging Face[5] under the name Swe19centOCR-2B.

The training was done on a single Nvidia A100 80GB, using a batch size of 4. The batch size was a bit conservative in relation to the 80GB VRAM, but we decided against increasing it after we had already started the fine-tuning. Each batch took roughly 30 minutes to train and evaluate, meaning that the whole fine-tuning of the model took 16 hours.

### 3.4.2 Fine-tuning of Qwen2.5-VL-7B-Instruct

Since the fine-tuning of the smaller Qwen3-VL-2B-Instruct worked so well, we decided against fine-tuning one of the 3-4 billion parameter models and went straight for the model that performs the best stock, the Qwen2.5-VL-7B-Instruct. We used the same hyper-parameters as for the Qwen3-VL-2B-Instruct. A rank and $\alpha$ of 8, a dropout rate of 0.1, an effective batch size of 64, an actual batch size of 4, and an initial learning rate of $10^{-5}$.

The error rates of the three epochs we trained are given in Table 3.7. The learning rate was lowered to $5{\cdot}10^{-6}$ after the first epoch. After observing successive worsening of the performance after each of the three epochs, we decided to terminate the fine-tuning and try our luck with another model. The reason for the poor results could be one or a combination of the following.

- Too low of a LoRA rank,
- A learning rate that is too aggressive,
- Impatience, the fine-tuning could eventually have turned around.

[5] `https://huggingface.co/J0hanski/Swe19centOCR-2B`

Table 3.7: Error rates achieved on the validation set while fine-tuning the Qwen2.5-VL-7B-Instruct over three epochs.

| Epoch | CER (%) | WER (%) |
|---|---|---|
| Stock | 3.876 | 13.829 |
| Epoch 1 | 4.004 | 14.085 |
| Epoch 2 | 4.731 | 16.275 |
| Epoch 3 | 5.207 | 16.908 |

### 3.4.3 Fine-tuning of Qwen3-VL-8B-Instruct with Rank 8

After terminating the fine-tuning of the Qwen2.5-VL-7B-Instruct, we decided to fine-tune the model that had the second best performance in terms of CER, Qwen3-VL-8B-Instruct. We used a similar set of hyper-parameters as for the previous models. A rank and $\alpha$ of 8, a dropout rate of 0.1, an effective batch size of 64 and an actual batch size of 4. However, the initial learning rate was lowered to $5 \cdot 10^{-6}$ and was reduced to $3 \cdot 10^{-6}$ after the third epoch.

The error rates of the four epochs trained with this setup are given in Table 3.8. This time an improvement in relation to the stock model was seen. However, we were not quite happy with the observed volatility between the epochs and overall slow progress. We suspected that the issue could be related to the LoRA rank. Too low-dimensional parameter space could make it hard to find a smooth path through it. We decided to terminate this training run as well, and train a model with a LoRA rank of 16 instead. It is unclear if this was the right decision. Perhaps the rank was actually too low, or perhaps the fine-tuning would have stabilized eventually, and we were just too impatient.

Table 3.8: Error rates achieved when fine-tuning Qwen3-VL-8B-Instruct over four epochs with a rank of 8.

| Epoch | CER (%) | WER (%) |
|---|---|---|
| Stock | 5.883 | 19.112 |
| Epoch 1 | 5.644 | 19.166 |
| Epoch 2 | 4.783 | 16.365 |
| Epoch 3 | 5.764 | 17.718 |
| Epoch 4 | 5.407 | 16.303 |

### 3.4.4 Fine-tuning of Qwen3-VL-8B-Instruct with Rank 16

For the setup with rank 16, we also increased the scaling parameter $\alpha$ to 16 in order to keep the quotient $\frac{\alpha}{r}$ at 1. The dropout rate was kept at 0.1, the effective batch size at 64 and actual batch size at 4. The initial learning rate was lowered to $3 \cdot 10^{-6}$.

Figure 3.5 shows the progress of the fine-tuning in terms of CER and WER achieved on the validation set after each epoch. The initial six epochs used a learning rate of $3 \cdot 10^{-6}$, epochs 7 to 16 used a learning rate of $2 \cdot 10^{-6}$, epochs 17 to 25 used a learning rate of $10^{-6}$, and the final five epochs used a learning rate of $5 \cdot 10^{-7}$. After epoch

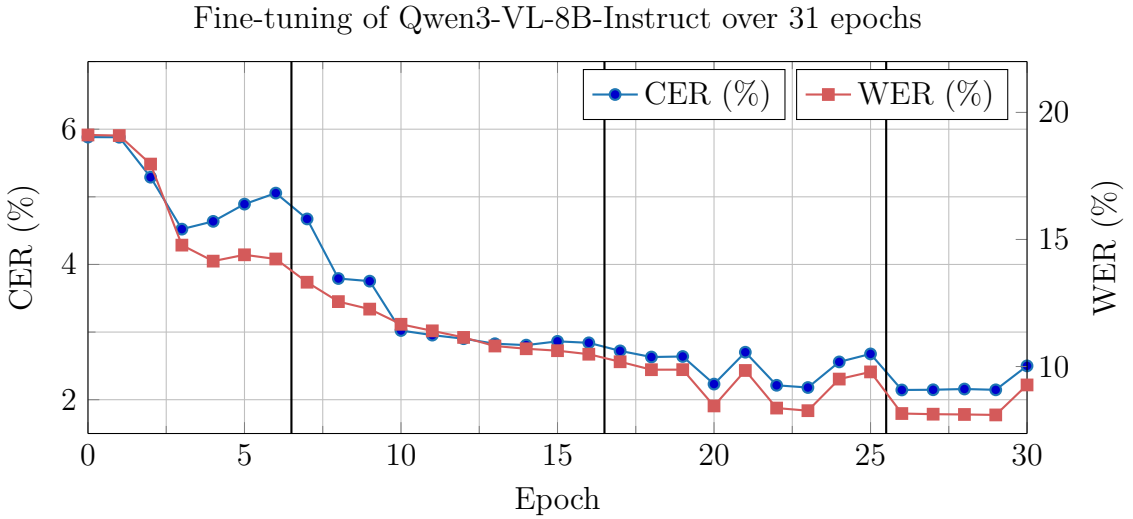



Figure 3.5: CER and WER achieved by each epoch on the validation set during the fine-tuning of Qwen3-VL-8B-Instruct with rank 16. Epoch 0 is the stock model. The vertical bars indicate where the learning rate was lowered.

30 we decided to terminate the fine-tuning since the improvements over the last ten epochs had been modest. Epoch 29 was chosen as our final model since it was the strongest in terms of WER and CER+WER on the validation set. Epoch 26 was the strongest in terms of CER though, by a very small margin (0.02 percentiles). The model is publicly available on Hugging Face[6] under the name Swe19centOCR-8B.

Just like the previous models, a single Nvidia A100 80GB rented through Google Colab was used. The batch size of 4 was well suited. It utilized the 80GB VRAM well without causing too many dropped batches because of memory overflow. Each batch took roughly 60 minutes to train and evaluate, meaning that the whole fine-tuning of the model took 30 hours.

### 3.4.5 Fine-tuning of Florence-2-large

The last model we fine-tuned was Florence-2-large. The training was performed with a LoRA rank of 8, an $\alpha$ of 16, a dropout rate of 0.05 and a learning rate of $10^{-5}$. The CER and WER of the stock model and epochs 5 and 10 are given in Figure 3.6. Since the model is not compatible with the vLLM inference engine, inference with the model was very slow. We therefore chose to only run inference after the fifth and tenth epoch. Instead, we opted to use the training and validation losses displayed in Figure 3.7 to monitor the progress (although the correlation between validation loss and actual model performance is quite weak). The fine-tuning was terminated after epoch 10 since the progress from the fifth to the tenth epoch was so small. It was clear that the model was not going to be able to beat the baseline set by the traditional engines (as described in Section 3.2).

[6] `https://huggingface.co/J0hanski/Swe19centOCR-8B`

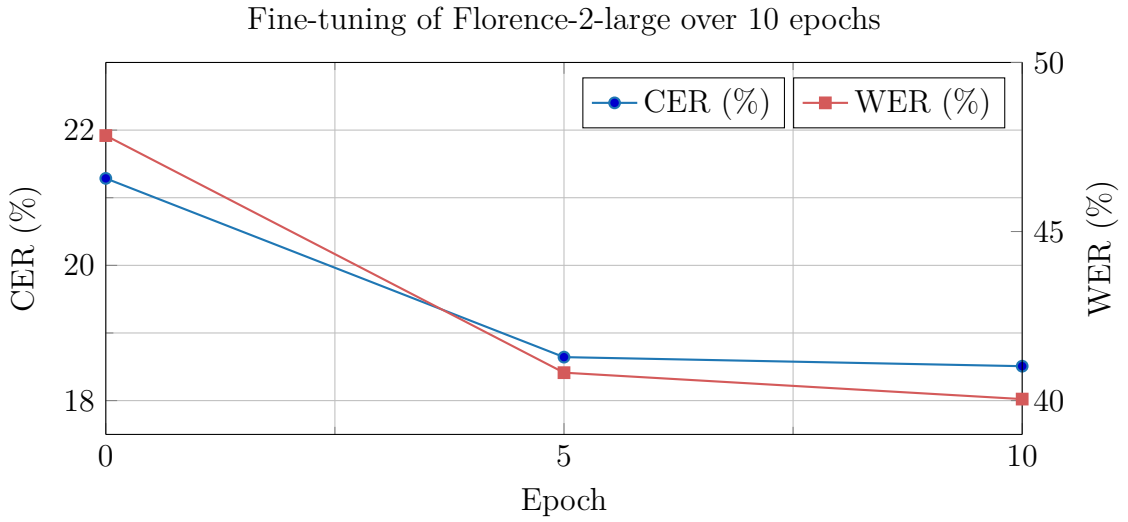



Figure 3.6: CER and WER achieved by ten epochs on the validation set during the fine-tuning of Florence-2-large. Epoch 0 is the stock model.

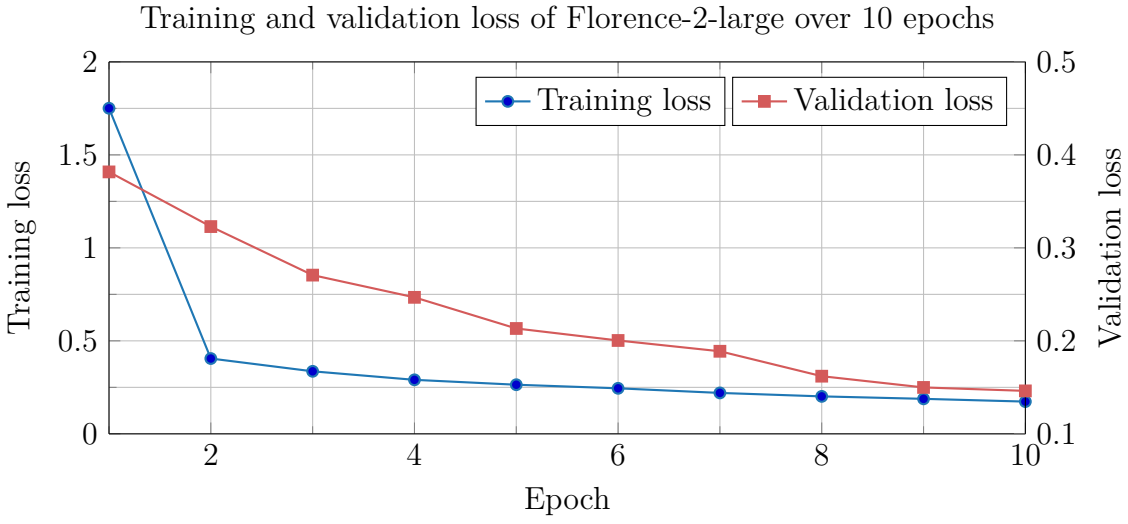



Figure 3.7: Losses achieved on the training and validation sets through the ten epochs of fine-tuning of Florence-2-large.

## 3.5 Repetition Trimmer

In order for this section to be understandable, we first need to clearly define the two terms “repeating string” and “repeating sequence”. Given the string “This is a string - - - ”, the “repeating string” is “- ”, and the “repeating sequence” is “- - - ”.

A common problem with VLMs is that they sometimes get stuck in a loop and start to output the same thing over and over again. Most of the time when this happens, the repeating string is just one or two characters long such as “.....” or “- - - ”, but the repeating string can also be a whole sentence or even a paragraph. When this happens, the model continues to output the repeating string until its max token limit is reached. Even though these failures are rare for stronger models (only happening on the order of magnitude once every 1000 segments on our validation set using Qwen3-VL-8B-Instruct), they end up contributing a significant proportion of the total errors.

To address this issue, we developed a repetition trimmer function that detects and trims repeating sequences at the end of a text. It is available on GitHub[7]. The function takes two parameters, a maximum string length and a threshold. The maximum string length defines the longest repeating string that is detectable, and the threshold defines the number of characters that the repeating sequence has to exceed in order to be detected. You want the maximum string length to be high enough to detect even the longest repeating strings, but not excessively high since the time it takes to run the function grows as approximately the square of it. The threshold is in general less critical than the maximum string length (a large range of values typically work well). You want it high enough not to catch naturally repeating strings at the end of segments, and low enough to catch repeating string even if they are initiated close to the models maximum token limit (and therefore are not allowed to proceed very long). There is also a small performance benefit to having a larger threshold, since texts shorter than the threshold can be returned instantly (there cannot possibly be any repetitions longer than $x$ if there are fewer than $x$ characters in the text).

Let us consider a simple example of how we want the function to behave. Let’s say our predicted text is “abcdefg.....cba.....cba.....cba...”. The model has gotten stuck and has started to repeat the string “.....cba”. We want to choose parameters such that the function catches the whole repeating part and returns “abcdefg.....cba”. If the threshold is chosen too low (in this case below 4), it will catch the substring “.” that repeats three times at the very end and return “abcdefg.....cba.....cba.....cba”. If we instead choose the threshold too high (in this case above 24), the text will be returned uncut since “.....cba.....cba.....cba” has 24 characters. For this particular example, we therefore need the threshold to be in the range [4, 24] to achieve the correct behavior. We also need a maximum string length of at least 8, since the repeating string “.....cba” has 8 characters.

[7] https://github.com/Martin31313/Swe19centOCR/blob/main/repetition-trimmer.py

We inspected the repetition failures of all outputs from the evaluations during the fine-tuning of the models on the validation set. The longest repeating string we found was 236 characters long, so we determined that a maximum string length parameter of 300 is suitable for our dataset in order to have some margin. The shortest failure-created repeating sequence we found was 425 characters long, so we determined that a threshold parameter of 300 was suitable as well, again to have some margin.

# 4

# Results

This chapter is structured as a combined results and discussions chapter. Analyses are presented, and then discussed right afterwards. The analyses are divided into a quantitative and a qualitative part.

The quantitative part consists of an evaluation of the systems' average performance across the test set, an evaluation of performance across time, an evaluation of the extent which the repetition trimmer improves performance, an evaluation of the prevalence case errors, an evaluation of how performance differs between typefaces, and a breakdown of which errors are the most common for the different systems.

The qualitative part analyses the outputs of a selection of segments in detail. First it analyses two segments that the all systems struggled a lot with. Then it analyses two segments of average difficulty. Finally, it compares the performance of different instances of the same two words that occur frequently throughout the test set.

In this chapter we refer to our fine-tuned versions of the Qwen3-VL-2B-Instruct and Qwen3-VL-8B-Instruct as Swe19centOCR-2B and Swe19centOCR-8B respectively. How the fine-tuning was done is described in Section 3.4.1 and Section 3.4.4.

## 4.1 Quantitative Evaluation

Table 4.1 shows the performance of Swe19centOCR-8B and Swe19centOCR-2B, as well as their stock counterparts Qwen3-VL-8B-Instruct and Qwen3-VL-2B-Instruct. The Qwen3-VL-8B-Instruct and Qwen3-VL-2B-Instruct are among the strongest performing open source stock models for their size, as shown in Section 3.3. All four models are run together with the repetition trimmer described in Section 3.5. It also shows the performance of Abbyy FineReader version 11.1.16 and Tesseract 5.5.1, which are considered the baseline as described in Section 3.2. The evaluation metrics used are CER, WER, Precision, Recall and F-score. How they are computed is described in Section 2.7.

As can be seen in the Table 4.1, even the stock Qwen3-VL-2B-Instruct performs on a similar level as the traditional OCR-engines. The Qwen3-VL-8B-Instruct outperforms FineReader 11.1.16 by 47.8% in terms of CER. Our Swe19centOCR-2B outperforms the stock Qwen3-VL-8B-Instruct, which makes it valuable for use cases where computational efficiency is prioritized over accuracy. Our Swe19centOCR-8B is the best performing system and it outperforms FineReader 11.1.16 by 68.5% in

Table 4.1: The evaluation metrics achieved on the test set by our VLM based OCR systems (VLM + repetition trimmer) as well as the traditional OCR engines FineReader 11.1.16 and Tesseract 5.5.1, considered the baseline. All values are in percent. For the Florence-2-large, "(FT)" stands for "fine-tuned" and "(St)" stands for stock.

| Model | CER ↓ | WER ↓ | Precision ↑ | Recall ↑ | F-score ↑ |
|---|---|---|---|---|---|
| Swe19centOCR-8B | 1.930 | 8.108 | 92.180 | 92.046 | 92.113 |
| Swe19centOCR-2B | 2.707 | 10.695 | 89.232 | 89.039 | 89.135 |
| Qwen3-VL-8B-Instruct | 3.198 | 12.453 | 88.388 | 87.349 | 87.865 |
| Qwen3-VL-2B-Instruct | 5.989 | 18.304 | 83.142 | 80.935 | 82.024 |
| FineReader 11.1.16 | 6.123 | 19.888 | 80.213 | 81.886 | 81.041 |
| Tesseract 5.5.1 | 7.089 | 21.882 | 78.195 | 80.161 | 79.166 |
| Florence-2-large (FT) | 14.354 | 36.338 | 64.322 | 67.207 | 65.733 |
| Florence-2-large (St) | 17.266 | 43.996 | 56.583 | 58.794 | 57.667 |

terms of CER. Since Florence-2-large remains substantially worse than the traditional OCR engines even after fine-tuning, it will not be included in the following analyses.

### 4.1.1 Evaluation of Performance Across Time

The error rates across time are shown in Figure 4.1 and Figure 4.2. We notice the similar topology of the graphs of different models. The segments that were harder for one model were typically harder for all of them and vice versa. This is true both for the VLM-based systems and the traditional systems. This suggests that segment level difficulty is dominated by intrinsic properties of the segments, rather than any weaknesses of a particular model.

We suspected that the models' performances would be poorer on the earlier newspapers because of elderly spelling and overall worse condition of the material. However, as seen in Figure 4.1 and Figure 4.2, this is not always the case. The performance of the models is at its worst during the 1820s, but then get much better on the following decades. The rest of the decades have similar error rates with a few notable under-performances, namely the years around 1870, and the early and late 1890s.

### 4.1.2 Evaluation of the Repetition Trimmer

Table 4.2 shows the difference in CER when using the VLMs alone, labeled as "wo.T.", and when combined with the repetition trimmer, labeled as "w.T.". Even without the repetition trimmer, the Swe19centOCR-8B, Swe19centOCR-2B and Qwen3-VL-8B-Instruct outperform the traditional OCR engines. The error reductions achieved by the repetition trimmer are quite substantial though, so it is hard to justify using a VLM without the repetition trimmer given how robust and light-driven it is.

An interesting thing to notice is that the fine-tuning greatly reduced the Swe19centOCR-2B's tendency to get stuck repeating text. This is not surprising, since repetitions make up the majority (55.369 %) of the total errors from the stock Qwen3-VL-

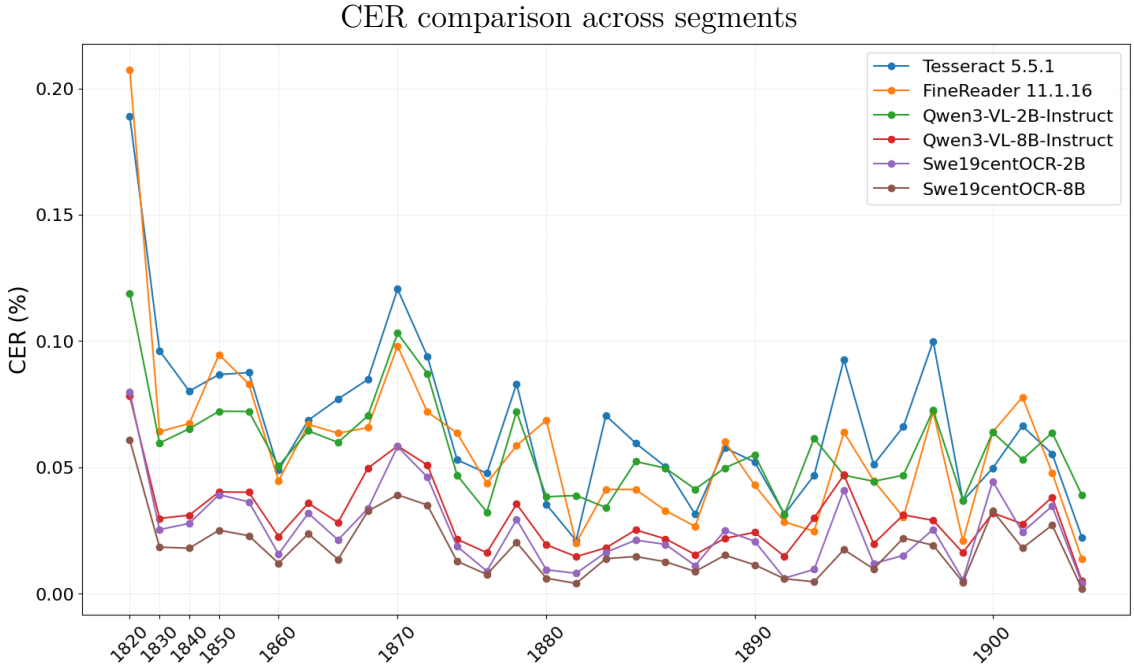



Figure 4.1: CER achieved by VLM based OCR systems (VLM + repetition trimmer) across the segments and the traditional OCR engines. Each point in the graph represents an average over 80 segments.

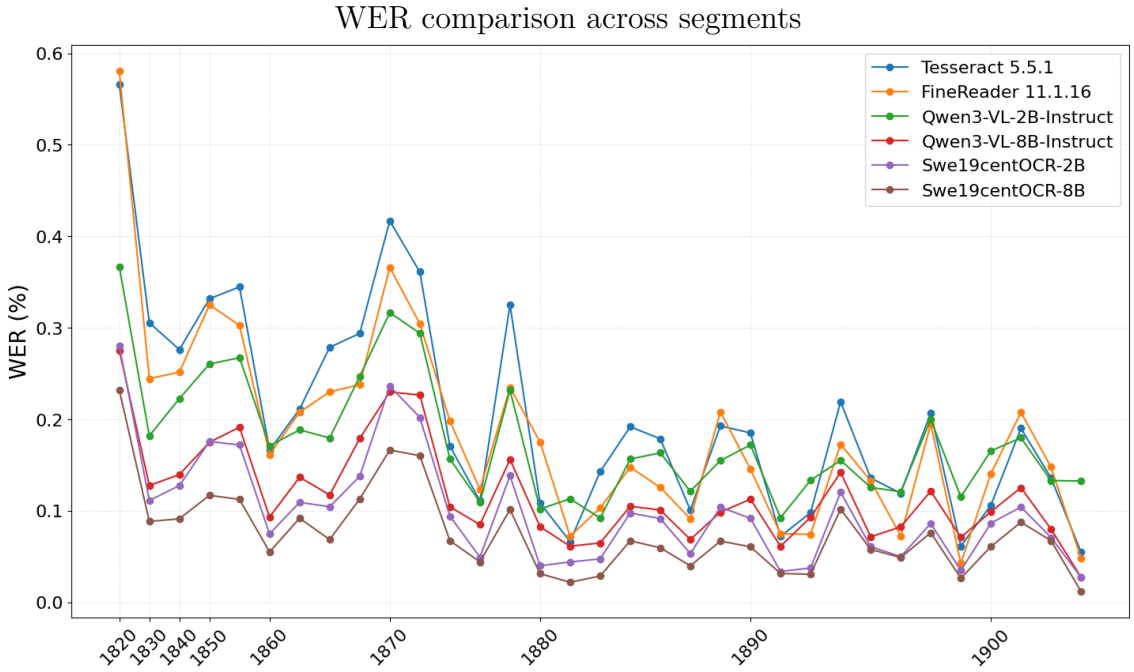



Figure 4.2: WER achieved by VLM based OCR systems (VLM + repetition trimmer) across the segments and the traditional OCR engines. Each point in the graph represents an average over 80 segments.

Table 4.2: Comparison between the CER achieved by the VLMs themselves without trimmer, and the whole VLM + Repetition Trimmer systems, as well as the absolute ("Abs.") and proportional ("Prop.") reduction in CER. The last column indicates how many of the 2583 segments in the test set that were affected by the repetition trimmer.

| | CER (%) | | Reduction | | |
|---|---|---|---|---|---|
| **Model** | **wo.T.** | **w.T.** | **Abs.** | **Prop. (%)** | **N. Seg.** |
| Swe19centOCR-8B | 3.565 | 1.930 | 1.635 | 45.863 | 4 |
| Swe19centOCR-2B | 4.649 | 2.707 | 1.942 | 41.772 | 4 |
| Qwen3-VL-8B-Instruct | 4.539 | 3.198 | 1.341 | 29.544 | 4 |
| Qwen3-VL-2B-Instruct | 13.419 | 5.989 | 7.430 | 55.369 | 23 |

2B-Instruct. Reducing this tendency is likely one of the most readily attainable improvements during the fine-tuning of that model.

Qwen3-VL-8B-Instruct already had a decently low tendency to get stuck repeating text before fine-tuning, and the tendency seems to have been mostly unaffected by the fine-tuning. The improved performance of Swe19centOCR-8B did not come from a reduced tendency to get stuck repeating text, but from other factors.

### 4.1.3 Evaluation of the Impact of Case

Table 4.3 shows the difference between the case-sensitive and the case-insensitive CER of the VLM based OCR systems and the traditional OCR engines. Errors caused by the case being wrong are on the order of magnitude 1% of the errors, with FineReader 11.1.16 standing out by having a noticeably higher proportion than the other systems. Either way, we can conclude that case-related errors are few, and there is little to be gained by addressing them specifically.

Table 4.3: Comparison between the case-sensitive ("C.S.") and the case-insensitive ("C.Ins.") CER achieved by the VLM based OCR systems (VLM + Repetion Trimmer) and the traditional OCR engines, as well as the absolute ("Abs.") and proportional ("Prop.") difference between them.

| | CER (%) | | Difference | |
|---|---|---|---|---|
| **Model** | **C.S.** | **C.Ins.** | **Abs.** | **Prop. (%)** |
| Swe19centOCR-8B | 1.930 | 1.909 | 0.021 | 1.088 |
| Swe19centOCR-2B | 2.707 | 2.679 | 0.028 | 1.034 |
| Qwen3-VL-8B-Instruct | 3.198 | 3.158 | 0.040 | 1.251 |
| Qwen3-VL-2B-Instruct | 5.989 | 5.943 | 0.046 | 0.768 |
| FineReader 11.1.16 | 6.123 | 5.984 | 0.139 | 2.270 |
| Tesseract 5.5.1 | 7.089 | 6.988 | 0.101 | 1.425 |

### 4.1.4 Evaluation by Typeface

As described in Section 2.6, there are two categories of typefaces used in our dataset, Blackletter and Antiqua. All of the newspapers in our dataset have been manually classified as either "Blackletter", "Antiqua" or "Both", as described in Section 3.1. The whole classification can be seen in Figure 3.2. The performance according to

typeface of our VLM based OCR systems, FineReader 11.1.16 and Tesseract 5.5.1 are presented in Table 4.4. It is clear that all four models, struggle more with Blackletter texts. This can be due to a multitude of factors. These are some of the possible ones:

- Blackletter typefaces are not commonly used today, likely meaning that they only constituted a small fraction of the models' pre-training material.
- The condition of the newspapers. As can be seen in Figure 3.2, Blackletter was more commonly used in the earlier parts of our dataset. It would therefore be reasonable to assume that the Blackletter material in our dataset is in a worse condition than the Antiqua material on average. It was however found that the Swe19centOCR-8B sometimes performs poorly on Blackletter text even when the text condition is good, as can be seen in Table 4.7.
- Spelling. Since Blackletter typefaces were more common earlier, the spelling in the Blackletter material differs more from the spelling of today on average than the Antiqua material.
- Some pairs of characters in Blackletter typefaces have a similar appearance, potentially making it intrinsically more difficult to read, as discussed in Section 2.6.1.

### 4.1.5 Character-wise Error Analysis

Table 4.5 shows the 30 most common character errors made by Swe19centOCR-8B and Swe19centOCR-2B, in combination with the repetition trimmer, as well as FineReader 11.1.16 and Tesseract 5.5.1. These are the computed substitutions, insertions and deletions in the Levenshtein distance for the CER (see Section 2.7).

The first thing we notice is that a lot of the top errors are common between all four models. Insertions and deletions of "." and " " made the top 30 for all models. Confusions between "." and "," are also common. Writing "-" instead of "=" is also done frequently by most models. This is because "-" often looks like "=" in Blackletter typefaces and is transcribed as "=" in the ground truth.

When it comes to letters, we see that "f" and "s" are confused many times by all systems because of the visual similarity between "f" and "f", long-s, in Blackletter typefaces. All systems also struggle to distinguish between "a", "å" and "ä" (for obvious reasons), and between "f", "k" and "t" (again because of their visual similarity in Blackletter typefaces). See Section 2.6.1 for a visual comparison of the letters.

Regarding quirks of the VLMs, both the 8B and the 2B models have a tendency to both miss many "s" and "t", and insert many "s" and "t" where there should not be any. Some of the insertions of "s" come from the models transcribing "ß" as "ss", whereas the ground truth generally does not do that substitution. Neither of the traditional engines do these errors regularly. Swe19centOCR-8B and FineReader 11.1.16 share the quirk of sometimes transcribing "w" as "v", whereas Swe19centOCR-2B and Tesseract 5.5.1 rarely do that.

Table 4.4: Performance of the systems based of typeface. The categories are separated per newspaper. "Both" refers to newspapers with multiple segments of both Blackletter and Antiqua. "Total" refers to the performance of the model on the whole test set. The models are arranged in order of performance, with the best performing model according to the "Total" CER listed first.

| **Model** | **Typeface** | **CER (%)** | **WER (%)** |
|---|---|---|---|
| Swe19centOCR-8B | Blackletter | 3.332 | 14.726 |
| | Antiqua | 1.292 | 5.010 |
| | Both | 1.847 | 8.158 |
| | Total | 1.930 | 8.108 |
| Swe19centOCR-2B | Blackletter | 4.646 | 19.373 |
| | Antiqua | 1.802 | 6.355 |
| | Both | 2.673 | 11.735 |
| | Total | 2.707 | 10.695 |
| Qwen3-VL-8B-Instruct | Blackletter | 4.955 | 20.836 |
| | Antiqua | 2.381 | 8.375 |
| | Both | 3.157 | 13.052 |
| | Total | 3.198 | 12.453 |
| Qwen3-VL-2B-Instruct | Blackletter | 8.562 | 28.820 |
| | Antiqua | 4.691 | 13.145 |
| | Both | 6.278 | 19.210 |
| | Total | 5.989 | 18.304 |
| FineReader 11.1.16 | Blackletter | 10.254 | 35.274 |
| | Antiqua | 4.127 | 12.231 |
| | Both | 6.280 | 21.598 |
| | Total | 6.123 | 19.888 |
| Tesseract 5.5.1 | Blackletter | 11.242 | 40.060 |
| | Antiqua | 5.033 | 12.475 |
| | Both | 7.417 | 25.175 |
| | Total | 7.089 | 21.882 |

The substitution "…" → "." making it quite high up on the list for the Swe19centOCR-2B is another quirk that we would like to mention. The majority (123 out of 215) of these errors actually come from a single segment, which can be seen in Figure 4.3 and Table 4.8.

A feature of the traditional engines is that they tend to end each transcription with an empty line. The VLMs do not do this, and neither is it done in the ground truth. If one were to use the traditional engines in combination with a simple function that deletes empty lines from their outputs, their CERs would be improved by approximately half a percentile.

Looking more broadly at the most common errors of FineReader 11.1.16, we notice that its top 30 list is dominated by substitutions between letters that look similar, especially in Blackletter typefaces, such as "å"/"ä", "f"/"s", "c"/"e", "l"/"t" and so on. We can also see its tendency to transcribe "s" as "S" high up in the list, which is the reason for its higher proportion of case-errors that we saw in Table 4.3.

Table 4.5: Comparison between the 30 most common character-wise errors (including the number of times it was done throughout the test set), for the Swe19centOCR-8B + repetition trimmer system, the Swe19centOCR-2B + repetition trimmer system, FineReader 11.1.16 and Tesseract 5.5.1. The character before the → is the character in the ground truth, and the character after the → is the character predicted by the OCR system. The number after ":" is how many times the error was made throughout the test set. For example, '"." → "": 334' means that the model wrote nothing when it should have written a "." 334 times. "\n" represents a line-break and is also considered to be a (single) character.

| **Swe19centOCR-8B** | | | **Swe19centOCR-2B** | | | **FineReader 11.1.16** | | | **Tesseract 5.5.1** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| "." | → | "": 334 | "." | → | "": 679 | "" | → | "\n": 2525 | "s" | → | "f": 2965 |
| " " | → | "": 310 | " " | → | "": 543 | "å" | → | "ä": 1525 | "" | → | "\n": 2787 |
| "s" | → | "f": 255 | "=" | → | "-": 429 | "." | → | "": 1006 | "s" | → | "f": 1225 |
| "" | → | " ": 211 | "s" | → | "f": 300 | "" | → | " ": 912 | "" | → | " ": 1041 |
| "ä" | → | "å": 200 | "k" | → | "t": 272 | "=" | → | "-": 901 | "å" | → | "ä": 978 |
| "f" | → | "s": 196 | "k" | → | "f": 261 | "s" | → | "f": 619 | " " | → | "": 952 |
| "" | → | "s": 188 | "s" | → | "": 243 | "e" | → | "c": 601 | "k" | → | "f": 684 |
| "" | → | ".": 182 | "ä" | → | "å": 237 | "" | → | "i": 479 | "=" | → | "-": 649 |
| "å" | → | "ä": 171 | "f" | → | "s": 237 | "t" | → | "l": 436 | "." | → | "": 629 |
| "s" | → | "": 152 | "å" | → | "ä": 215 | "s" | → | "S": 420 | "c" | → | "": 450 |
| "=" | → | "-": 150 | "." | → | "…": 215 | " " | → | "": 399 | "." | → | ",": 295 |
| "w" | → | "v": 132 | "" | → | "s": 210 | "f" | → | "s": 374 | "s" | → | "": 288 |
| "," | → | ".": 114 | "," | → | ".": 198 | "d" | → | "b": 316 | "l" | → | "": 282 |
| "k" | → | "f": 113 | "" | → | " ": 192 | "n" | → | "u": 288 | "" | → | ".": 280 |
| "k" | → | "t": 89 | "," | → | "": 174 | "" | → | "t": 273 | "," | → | "": 246 |
| "c" | → | "": 76 | "t" | → | "": 151 | "" | → | "l": 263 | "" | → | "-": 236 |
| "l" | → | "": 74 | "l" | → | "": 129 | "k" | → | "t": 255 | "r" | → | "": 224 |
| "t" | → | "": 69 | "t" | → | "k": 125 | "u" | → | "n": 246 | "e" | → | "": 221 |
| "n" | → | "": 64 | "" | → | ".": 119 | "å" | → | "a": 203 | "a" | → | "": 208 |
| "ß" | → | "s": 63 | "n" | → | "": 108 | "h" | → | "b": 180 | "t" | → | "": 205 |
| "," | → | "„": 62 | "f" | → | "": 89 | "," | → | ".": 179 | "" | → | ",": 194 |
| "" | → | "t": 61 | "e" | → | "": 87 | "m" | → | "n": 177 | "n" | → | "": 193 |
| "e" | → | "": 59 | "r" | → | "": 86 | "" | → | ".": 173 | "." | → | "\n": 189 |
| "" | → | "l": 57 | "i" | → | "": 81 | "" | → | "f": 171 | "h" | → | "b": 189 |
| "" | → | "n": 57 | "c" | → | "": 81 | "l" | → | "i": 163 | "o" | → | "": 180 |
| "" | → | "1": 55 | "-" | → | "=": 75 | "k" | → | "f": 155 | "i" | → | "": 150 |
| "n" | → | "u": 53 | "" | → | "t": 74 | "w" | → | "v": 150 | "k" | → | "d": 144 |
| "t" | → | "k": 51 | "a" | → | "ä": 70 | "k" | → | "l": 144 | "ä" | → | "å": 141 |
| "r" | → | "": 51 | "ß" | → | "s": 68 | "t" | → | "i": 141 | "1" | → | "": 141 |
| "." | → | ",": 49 | "ä" | → | "a": 67 | "n" | → | "i": 139 | "" | → | "E": 139 |

A feature of Tesseract 5.5.1 in particular is that it transcribes "f", long-s, instead of substituting it for regular "s" like the other three systems and the ground truth. If one were to use Tesseract 5.5.1 in combination with a simple function that substitutes all of the long-s for regular "s", its CER would improve by 0.645 percentiles, putting it just 0.321 percentiles behind FineReader 11.1.16.

By studying the most common errors of Tesseract 5.5.1, one notices that most of them are deletions, which becomes especially noticeable in comparison with FineReader 11.1.16. A major contributor to this is that Tesseract 5.5.1 often does not transcribe anything at all when the segment only consists of a single row, which is also discussed in Section 4.2.3. An almost constant failure to recognize “c” before “k” and “h” in Blackletter typefaces is another quirk unique to Tesseract. Tesseract also struggles more than the other systems with “,”. It regularly both misses “,” where there should be one, and writes “,” where there should not be any.

### 4.1.6 Assessment of Fine-Tuning Convergence

Table 4.6 shows the performance of our fine-tuned models together with the repetition trimmer on the different subsets of the dataset. As can be seen in the table, the models do not perform any better on the subset they were fine-tuned on. In fact, they actually perform worse on the training set than on the validation and test sets. This is probably because of a slightly higher proportion of hard segments, e.g. complicated tables and text in very poor condition, were randomly allocated to the training set during partitioning. Either way, the fact that neither model performed any better on the segments it had actually seen during fine-tuning than unseen ones suggests that the LoRA adapter size could have been a bottle neck that made the fine-tuning plateau.

Table 4.6: Comparison between the CER and WER achieved by the Swe19centOCR-8B and Swe19centOCR-2B together with the repetition trimmer on the different subsets of the dataset.

| Model | Subset | CER (%) | WER (%) |
|---|---|---|---|
| Swe19centOCR-8B | Test set | 1.930 | 8.108 |
| | Validation set | 2.145 | 8.095 |
| | Training set | 2.253 | 8.594 |
| Swe19centOCR-2B | Test set | 2.707 | 10.695 |
| | Validation set | 2.729 | 10.868 |
| | Training set | 3.142 | 11.377 |

## 4.2 Qualitative Evaluation

A qualitative analysis of the results was made by one of the authors of this thesis in order to get a better understanding of when models perform poorly. To find which types of text the models struggle with the most, we identified the 30 segments in which the Swe19centOCR-8B had the worst performance in terms of number of character errors, and studied them. When finding the top 30 segments with the worst performance, the number of characters in the output is taken into account. The purpose of this is to find the segments that affect the total CER the most, i.e. a longer segment with a high CER affects the total error more than a shorter segment with the same CER. The division of types of texts can be seen in Table 4.7. This table shows that there is a wide spread in the types of texts that get high errors. In the table, quality is measured as either “Good” or “Bad”. A text is considered to be

of good quality if it does not exhibit warping, buckling, smudging, ink bleed-through, or stains. "Text" in the context of type refers to a text that is not a list or a table, however, it can still have a slight variation in layout.

Table 4.7: A manual classification of the 30 segments with the worst performance using the Swe19centOCR-8B. The segments have been assessed subjectively.

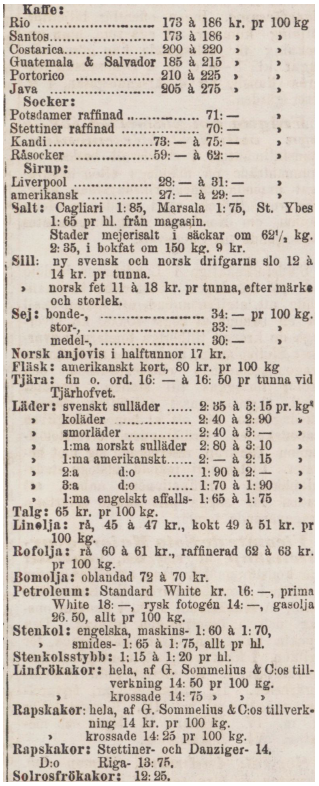
| Type | Typeface | Quality | N. Seg. |
|---|---|---|---|
| Text | Blackletter | Bad | 10 |
| List | Antiqua | Good | 6 |
| Text | Blackletter | Good | 6 |
| Text | Antiqua | Good | 3 |
| Table | Antiqua | Good | 2 |
| Sideways table | Antiqua | Good | 2 |
| List | Blackletter | Good | 1 |

What is interesting to see in Table 4.7 is the number of segments that are tables or lists. Out of the 30 segments with the worst results, 11 of them are tables or lists. This may not sound like a lot, but out of the 2,583 segments in the test set, only 35 of them are lists, as discussed in Section 3.1.3. This makes almost a third of all tables and lists part of the 30 most difficult segments to transcribe. It is therefore not unreasonable to conclude that the models studied in this thesis struggle the most with transcribing advanced layouts, such as tables and lists. Apart from tables and list, we found that the models struggle with many different types of texts, such as single row segments, images, paragraphs with an unusual layout, and so on.

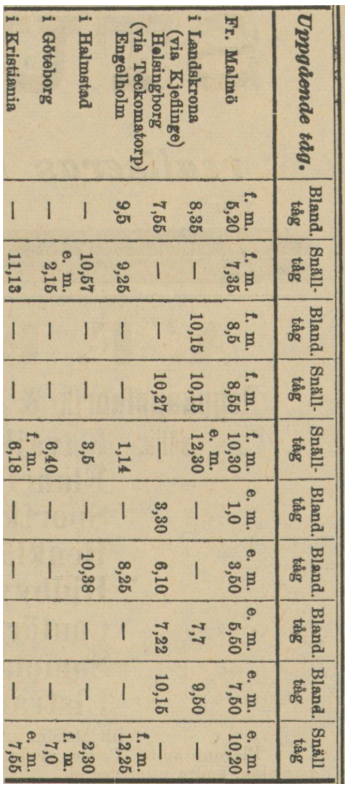
### 4.2.1 Exemplification of Errors Made in Tables and Lists

Two of the segments in the test set that contributed the most to the error rates are shown in Figure 4.3. The first five rows of the ground truth and the outputs of Swe19centOCR-8B, Swe19centOCR-2B, FineReader 11.1.16 and Tesseract 5.5.1 are displayed in Tables 4.8 and 4.9. The complete ground truth and outputs from the models are found in Appendix A.1. The tables show that even if a model is able to transcribe the text, it is not always obvious in which order the text should be transcribed. If the text is transcribed in a different order than that of the ground truth, the error rates will be high.

An example of an error that can occur is found in Table 4.8, in the output of the Swe19centOCR-2B. Here, the ground truth uses individual points, ".", to transcribe the points in the image. However, Swe19centOCR-2B does instead use the character "…". For a reader, this difference is negligible, but in terms of CER and WER, the result is affected. In Table 4.8 and 4.9, errors are marked in red. This is not however the only aspect that affects CER and WER. As described in Section 2.7, CER and WER count substitutions, deletions, and insertions. Missing or added line breaks, letters and blank spaces all increase error rate.



**Kaffe:**
Rio ..... 173 à 186 kr. pr 100 kg
Santos..... 173 à 186 » »
Costarica..... 200 à 220 » »
Guatemala & Salvador 185 à 215 » »
Portorico ..... 210 à 225 » »
Java ..... 205 à 275 » »
**Socker:**
Potsdamer raffinad ..... 71:— »
Stettiner raffinad ..... 70:— »
Kandi.....73:— à 75:— »
Råsocker .....59:— à 62:— »
**Sirup:**
Liverpool ..... 28:— à 31:— »
amerikansk ..... 27:— à 29:— »
**Salt:** Cagliari 1:35, Marsala 1:75, St. Ybes 1:65 pr hl. från magasin.
Stader mejerisalt i säckar om 62½ kg. 2:35, i bokfat om 150 kg. 9 kr.
**Sill:** ny svensk och norsk drifgarns slo 12 à 14 kr. pr tunna.
» norsk fet 11 à 18 kr. pr tunna, efter märke och storlek.
**Sej:** bonde-, ..... 34:— pr 100 kg.
stor-, ..... 33:— »
medel-, ..... 30:— »
**Norsk anjovis** i halftunnor 17 kr.
**Fläsk:** amerikanskt kort, 80 kr. pr 100 kg
**Tjära:** fin o. ord. 16:— à 16:50 pr tunna vid Tjärhofvet.
**Läder:** svenskt sulläder ..... 2:35 à 3:15 pr. kg
» koläder ..... 2:40 à 2:90 »
» smorläder ..... 2:40 à 3:— »
» 1:ma norskt sulläder 2:80 à 3:10 »
» 1:ma amerikanskt..... 2:— à 2:15 »
» 2:a d:o ..... 1:90 à 2:— »
» 3:a d:o ..... 1:70 à 1:90 »
» 1:ma engelskt affalls- 1:65 à 1:75 »
**Talg:** 65 kr. pr 100 kg.
**Linolja:** rå, 45 à 47 kr., kokt 49 à 51 kr. pr 100 kg.
**Rofolja:** rå 60 à 61 kr., raffinerad 62 à 63 kr. pr 100 kg.
**Bomolja:** oblandad 72 à 70 kr.
**Petroleum:** Standard White kr. 16:—, prima White 18:—, rysk fotogén 14:—, gasolja 26.50, allt pr 100 kg.
**Stenkol:** engelska, maskins- 1:60 à 1:70,
» smides- 1:65 à 1:75, allt pr hl.
**Stenkolsstybb:** 1:15 à 1:20 pr hl.
**Linfrökakor:** hela, af G. Sommelius & C:os tillverkning 14:50 pr 100 kg.
» krossade 14:75 » » »
**Rapskakor:** hela, af G. Sommelius & C:os tillverkning 14 kr. pr 100 kg.
» krossade 14:25 pr 100 kg.
**Rapskakor:** Stettiner- och Danziger- 14.
D:o Riga- 13:75.
**Solrosfrökakor:** 12:25.



| *Uppgående tåg.* | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Bland. tåg | | Snäll- tåg | | Bland. tåg | | Snäll- tåg | | Snäll- tåg | | Bland. tåg | | Bland. tåg | | Bland. tåg | | Bland. tåg | | Snäll tåg | |
| Fr. Malmö | f. m. 5,20 | | f. m. 7,35 | | f. m. 8,5 | | f. m. 8,55 | | f. m. 10,30 | | e. m. 1,0 | | e. m. 3,50 | | e. m. 5,50 | | e. m. 7,50 | | e. m. 10,20 | |
| i Landskrona (via Kjeflinge) | 8,35 | | — | | 10,15 | | 10,15 | | e. m. 12,30 | | — | | — | | 7,7 | | 9,50 | | — | |
| Helsingborg (via Teckomatorp) | 7,55 | | — | | — | | — | | — | | 8,30 | | 6,10 | | 7,22 | | 10,15 | | f. m. 12,25 | |
| Engelholm | 9,5 | | 9,25 | | — | | — | | 1,14 | | — | | 8,25 | | — | | — | | | |
| i Halmstad | — | | — | 10,57 | — | | — | | 3,5 | | — | | 10,38 | | — | | — | | 2,30 | |
| i Göteborg | — | | — | e. m. 2,15 | — | | — | | f. m. 6,40 | | — | | — | | — | | — | | f. m. 7,0 | |
| i Kristiania | — | | 11,13 | | — | | — | | f. m. 6,18 | | — | | — | | — | | — | | e. m. 7,55 | |

Figure 4.3: Newspaper clippings from *Nya Dagligt Allehanda*, 17th of April 1894 [47] to the left, and *Sydsvenska Dagbladet*, 22nd of June 1902 [48] to the right. Similar tables are found in multiple newspapers in our dataset. The right segment is shown in its original form, where it appears sideways in the newspaper.

By studying the outputs in Table 4.8, a lot of different errors can be found. The Swe19centOCR-8B and Swe19centOCR-2B perform relatively good in terms of transcribing the list to match the actual image and is easy for the reader to understand. However, because the outputs do not align with the ground truth, the error rates remain high. FineReader 11.1.16 and Tesseract 5.5.1 fail to find the dots in the image and have trouble identifying characters correctly, enlarging their error. Tesseract stands out even further as the model hallucinates characters that are not even in the image. It is hard to say why Tesseract does this. One reason could be because of the black vertical line to the very left of the image but it could also be due to the model trying to find letters and words in the dots.

The outputs found in Table 4.9 suffer from even higher error rates, with WER reaching above 100%. This was somewhat expected, as transcribing vertical segments is more difficult. Since the models are asked to transcribe the text as it is, a problem arises. If a human were to transcribe the text, they would reasonably have tried to write it according to a table from the top row to the bottom. A model, on the other hand, could easily transcribe from left to right, even if the words themselves are vertical. In the left image in Figure 4.3, the space between the items and the prices are separated by dots, which provides a clear division.

For the right image however, the separation made between objects are spaces and lines, which cannot be transcribed using just letters. In this specific case, the right

Table 4.8: Ground truth and generated outputs based of newspaper clippings from *Nya Dagligt Allehanda*, 17th of April 1894 [47]. Only the first five rows of ground truth and the outputs are shown. Errors in the outputs are marked in red. Complete ground truth and outputs are found in Appendix A.1.

| | |
|---|---|
| **Ground truth** | Kaffe: |
| | Rio ..... 173 à 186 kr. pr 100 kg |
| | Santos ..... 173 à 186 » » |
| | Costarica ..... 200 à 220 » » |
| | Guatemala & Salvador 185 à 215 » » |
| **Swe19centOCR-8B** | Kaffe: |
| CER 7.516 % | Rio ..... 173 à 186 kr. pr 100 kg |
| WER 39.855 % | Santos..... 173 à 186 , |
| | Costarica..... 200 à 220 , |
| | Guatemala & Salvador 185 à 215 , |
| **Swe19centOCR-2B** | Kaffe: |
| CER 22.595 % | Rio ..... 173 à 186 kr. pr 100 kg |
| WER 39.614 % | Santos..... 173 à 186 |
| | Costarica..... 200 à 220 |
| | Guatemala & Salvador 185 à 215 |
| **FineReader 11.1.16** | |
| CER 22.786 % | Kaffe: |
| WER 31.401 % | : Rio 173 å 186 hr. pr 100 kg |
| | Santos 173 å 186 » » |
| | Costarica 200 å 220 > » |
| **Tesseract 5.5.1** | Kaffe: |
| CER 30.732 % | |
| WER 52.174 % | HEBIO” 22.2... ess ecases 170 8 286 KE; DERE |
| | , Santos... 5. 2 F8- 2.1865 » |
| | i Costarica... GIS POE UED Ma 200 a 220 > > |

Table 4.9: Ground truth and generated outputs based of newspaper clippings from *Sydsvenska Dagbladet* , 22nd of June 1902 [48]. Only the first five rows of ground truth and the outputs are shown. Errors in the outputs are marked in red. Complete ground truth and outputs are found in Appendix A.1.

| | |
|---|---|
| **Ground truth** | Uppgående tåg. Bland. Snäll- Bland. Snäll- Snäll- Bland. Bland. Bland. Bland. Snäll |
| | tåg tåg tåg tåg tåg tåg tåg tåg tåg tåg |
| | f. m. f. m. f. m. f. m. f. m. e. m. e. m. e. m. e. m. e. m. |
| | Fr. Malmö 5,20 7,35 8,5 8,55 10,30 1,0 3,50 5,50 7,50 10,20 |
| | e. m. |
| **Swe19centOCR-8B** | Uppgående tåg. |
| CER 82.154 % | Bland. Snäll- |
| WER 117.143 % | tåg tåg |
| | Bland. Snäll- |
| | tåg tåg |
| **Swe19centOCR-2B** | Uppgående tag. |
| CER 91.801 % | Bland. |
| WER 96.429 % | Small- |
| | Bland. |
| | Small- |
| **FineReader 11.1.16** | ~ ~ ~ ^ g |
| CER 80.707 % | 111 ispi $ I |
| WER 158.571 % | 111 pifi I \| |
| | 6 rgI\|B * |
| | S” A g; |
| **Tesseract 5.5.1** | Bland.\| Bland.\| Bland.\| Bland.\| Snäll |
| CER 83.119 % | |
| WER 129.286 % | Em Kjeflinge) |
| | |
| | Helsingborg |

image has a ground truth that is transcribed as each row in the table to be its own line, not taking the cells into account. The Swe19centOCR-8B started from the top of the table, but chose to output the text in a somewhat more cell-based layout. The Swe19centOCR-2B started with the first row in the table, but instead chose to separate each cell with a line-break. Swe19centOCR-2B did also edit "Snäll" to "Small", which could indicate that the model prefers an English spelling.

FineReader 11.1.16 could not handle the transcription of the vertical table in Figure 4.3 at all. The reason for this is probably that the model did not understand that the table was vertical, and tried to find characters in rotated text and layout. Tesseract 5.5.1 however understood that the table was rotated, yet it missed a lot of words and introduced "|" to illustrate the cells.

Overall, by doing a qualitative analysis of the performance of the different models for the segments found in Figure 4.3, Tesseract 5.5.1 had the worst performance in Table 4.8 and FineReader 11.1.16 had the worst performance in Table 4.9. However, in the latter this is not true according to CER of the different models. Even if the output of FineReader is complete nonsense, it still achieved a CER of 80.707%, which was the lowest out of all of the models. Nevertheless, FineReader had the worst performance in terms of WER.

### 4.2.2 Exemplification of Errors Made in Shorter Paragraphs

Examples of segments with significantly lower error rates are found in Figure 4.4. These segments are very short paragraphs with roughly the same layout and quality. The upper image is a segment written in Blackletter and the lower image is a segment written in Antiqua. The outputs of Swe19centOCR-8B, Swe19centOCR-2B, FineReader 11.1.16, Tesseract 5.5.1, and the ground truth are found in Table 4.10 and 4.11.

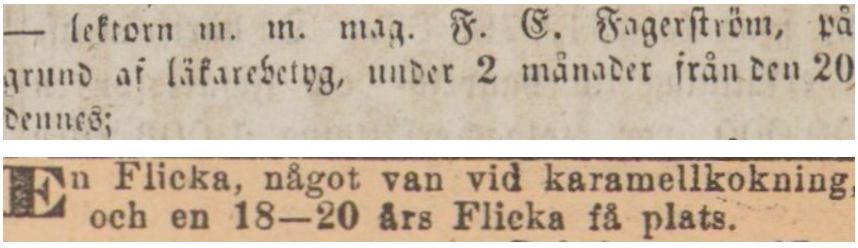



Figure 4.4: Newspaper clippings from *Falköpings Tidning*, 23rd of June 1865 [49], and *Stockholmstidningen*, 17th of May 1897 [50]. The first image is a segment written in Blackletter and the second image is a segment written in Antiqua.

In shorter paragraphs, a substitution of a character causes a large impact on the CER and WER. This is clear in, for example Swe19centOCR-8B in Table 4.10, where one substitution and one deletion cause a WER of 10.526%. However, when compared to the whole page with multiple segments and characters, this is a very small error in total.

Table 4.10: Ground truth and generated outputs based of newspaper clippings from *Falköpings Tidning*, 23rd of June 1865 [49]. Errors in the outputs are marked in red.

| | |
|---|---|
| **Ground truth** | — lektorn m. m. mag. F. E. Fagerström, på grund af läkarebetyg, under 2 månader från den 20 dennes; |
| **Swe19centOCR-8B**<br>CER 2.020 %<br>WER 10.526 % | — lektorn m. m. mag. F. G. Fagerström, på grund af läkarebetyg, under 2 månader från den 20 denes; |
| **Swe19centOCR-2B**<br>CER 3.030 %<br>WER 15.790 % | — lektorn m. m. mag. F. C. Tagerström, på grund af läfarebetyg, under 2 månader från den 20 dennes; |
| **FineReader 11.1.16**<br>CER 6.061 %<br>WER 21.053 % | — lektorn m. in. mag. F. C. Fagerstiöm, på grund af låkarebetyg, under 2 månader från den 20 dennes; |
| **Tesseract 5.5.1**<br>CER 8.081 %<br>WER 36.842 % | — lefktorn m. m. mag. F E. Fagerftröm, på zrund af läkarebetyg, under 2 månader från.den 20 deunes; |

Table 4.11: Ground truth and generated outputs based of newspaper clippings from *Stockholmstidningen*, 17th of May 1897 [50]. Errors in the outputs are marked in red.

| | |
|---|---|
| **Ground truth** | En Flicka, något van vid karamellkokning, och en 18-20 års Flicka få plats. |
| **Swe19centOCR-8B**<br>CER 1.333 %<br>WER 7.692 % | En Flicka, något van vid karamellkokning, och en 18—20 års Flicka få plats. |
| **Swe19centOCR-2B**<br>CER 2.667 %<br>WER 15.385 % | En Flicka, något van vid karamellkokning och en 18—20 års Flicka få plats. |
| **FineReader 11.1.16**<br>CER 13.333 %<br>WER 30.769 % | Wjtn Flicka, något van vid karamellkokning *44 och en 18—20 års Flicka få plats. |
| **Tesseract 5.5.1**<br>CER 10.667 %<br>WER 30.769 % | 1:50 Flicka, nägot van vid karamellkokning, och en 18—20 års Flieka få plats. |

The overall performances of the models on these two segments are good, with only a few errors. As described in Section 2.6.1, letters in Blackletter can sometimes be confused with each other. In Table 4.10, it is shown that the letter "E" is confused with "G" and "C". It is interesting to see how the models can get a character wrong, but in the same segment also get it right. This happens for both Swe19centOCR-2B and Tesseract 5.5.1, where they confuse the letter "f" with "k", yet do it right in another word. Swe19centOCR-2B also mistakes the letter "F" for being "T", although the model had already predicted the letter correctly just before.

An interesting thing to notice in Table 4.11 is the error completed by FineReader 11.1.16 and Tesseract 5.5.1 due to the enlarged "E". Both of our fine-tuned VLMs handle this layout of text without problems. This error made by Tesseract could be because of the model's architecture, described in Section 2.2.1. Tesseract starts of with finding the textline of the text and forms blobs based of it. In the case for the lower image in Figure 4.4, the enlarged "E" is not at the same textline as the rest of the text, which could make the model unable to recognize it as a letter. This error is a great example where VLMs are better to use for OCR, since they are able to approach the image as a whole instead of having to break it down line by line and character by character.

### 4.2.3 Exemplification of Errors Made in Single Row Segments

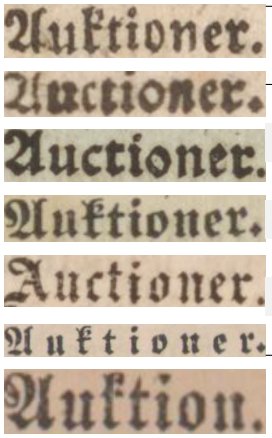
To compare differences in similar single row segments for which the models produce different outputs, Figure 4.5 and Figure 4.6 is shown. The chosen segments are examples of recurring words found in the test set, namely "förloradt" and "auktioner". They are commonly used as headings, which is why they are frequently appearing and for most of the time in single row segments. They vary in spelling, usage of uppercase and lowercase letters and quality of the segment.

GT: (F|f)örloradt.

| Segment | Swe19cent OCR-8B | Swe19cent OCR-2B | FineReader 11.1.16 | Tesseract 5.5.1 |
|---|---|---|---|---|
| 1 | Sölforade. | Sokoln. | Fsrloradr. | Sörloradt. |
| 2 | Förlorad | Sökecäde | Forlorade O wivvnvfr | Sorlorgor |
| 3 | Koloradt. | Sorloradt. | orloradt | Forloradt. |
| 4 | Förloradt. | Förloradt. | r örloradf. | Hörloradt. |
| 5 | förlokat. | Förkoradt. | JMoraDt. | |

Figure 4.5: Examples of similar segments from the test set. The ground truth is displayed above the segments while the models different outputs are shown to the right. Errors are marked with red. The segments are newspaper clippings, listed from top to bottom, from *Götheborgs Allehanda*, 26th of May 1819 [51], *Götheborgs Tidningar*, 10th of December 1822 [52], *Hvad Nytt*, 21st of December 1852 [53], *Upsala*, 7th of May 1867 [54] and *Härnösandposten*, 9th of September 1876 [55].

GT:
Au(k|c)tion(er).



| Segment | Swe19cent OCR-8B | Swe19cent OCR-2B | FineReader 11.1.16 | Tesseract 5.5.1 |
|---|---|---|---|---|
| 1 | Auktioner. | Auktioner. | Auktioner. | Auktioner. |
| 2 | Auctioner. | Auctioner. | Auclioner. | |
| 3 | Auctioner. | Auctioneer. | Aucrioner. | Auctioner. |
| 4 | Auktioner. | Aufftoner. | Auktioner. | Auktioner. |
| 5 | Auctioner. | Auctioneer. | Äiictioner. | Auctioner. |
| 6 | Auktioner. | Auctioneer. | A u k t i o n e r. | Auftioner. |
| 7 | Vuktion. | Auction. | Auktion. | Auktion. |

Figure 4.6: Examples of similar segments from the test set. The ground truth is displayed above the segments while the models different outputs are shown to the right. Errors are marked with red. The segments are newspaper clippings, listed from top to bottom, from *Götheborgska Nyheter*, 27th of October 1821 [56], *Götheborgs Tidningar* 10th of December 1822 [52], *Post- och Inrikes Tidningar*, 16th of June 1825 [57], *Jönköpingsbladet*, 3rd of May 1845 [58], *Nerikes Allehanda*, 18th of March 1846 [59], *Blekingsposten*, 25th of October 1859 [60] and *Kalmar*, 15th of June 1872 [61].

An interesting point to notice in Figure 4.5 is the amount of errors made by all of the models. Although none of the segments are in particularly poor condition, the models still produce many errors. Some of the errors are reasonable for a human, such as the letter "f" in segments 1 and 2 resembling an "S". At the same time, there are many problems that cannot really be explained, such as the Swe19centOCR-2B making a lot of errors in both segments 1 and 2, and FineReader 11.1.16 hallucinating a lot of extra letters for segment 2. Although segments 1 and 2 are almost identical in appearance, the outputs of the same models differ significantly. As Table 4.4 showed, all of the models perform better on texts written in Antiqua, which explains why the models achieve the best performance in segment 4. Segment 5 is also interesting to observe. It is of great quality and has clearly presented letters. However, the models struggle, including FineReader introducing inaccurate letters and Tesseract failing to produce an output at all.

As for Figure 4.6, all of the models achieve significantly better performance. Here, most of the letters are correctly transcribed with only a few mistakes. The errors made by the Swe19centOCR-2B opens up the question of whether the model incorporate knowledge of English grammar, or if it simply is a mistake when trying to transcribe the letters. It does introduce a "c" instead of a "k" two times out of seven and does also introduce an additional "e" three times out of seven. It is like the model is intentionally shaping the word to be of English form as "Auctioneer". Other errors are again found in Tesseract 5.5.1., as it fails to transcribe segment 2. It is also interesting to see the output from the Swe19centOCR-8B in segment 7, where a clear "A" is mistaken for a "V". We cannot conclude what the cause of this is, however, it shows the uncertainty of even the best performing models.

# 5

# Conclusion

## 5.1 Summary of the Findings

As described in Section 4.1, it is clear that small to medium sized VLM based OCR systems outperform traditional OCR engines such as FineReader 11.1.16 and Tesseract 5.5.1 for OCR of 19th century Swedish newspaper material. Even without fine-tuning, the stock Qwen3-VL-2B-Instruct and Qwen3-VL-8B-Instruct outperform traditional OCR engines when combined with our simple repetition trimmer, and our Swe19centOCR-8B does 68.5% fewer character errors than FineReader 11.1.16.

Although quantitative measures are the most straight forward way to compare the performance of different OCR systems, it is also of great importance to do qualitative analysis of the models to find out how errors are made and why they might be appearing. The transcription with the lowest error rate is not necessarily the easiest to read, although the correlation between error rate and ease of reading is in general high. The order of words, layout and choice of characters are highly important.

Returning to the research questions of the thesis stated in Section 1.2,

- Are OCR systems based on small to medium sized open-source VLMs competitive with traditional OCR engines for OCR of 19th century Swedish newspaper material?
- To what extent can the performance of VLM based OCR systems be improved by fine-tuning the VLM using a small dataset?

we can conclude that the best small to medium sized open source VLMs outperform traditional OCR engines on 19th century Swedish newspaper material even without any fine-tuning, and that with fine-tuning and the use of a simple repetition trimmer, the gap gets even larger.

## 5.2 Future Work

VLMs have improved fast during the past couple of years, and it seems likely that they are going to continue to do so in the near future. This means that it would likely be beneficial to do a similar project with the next generation of VLMs once they are released. Even with the current generation of VLMs, it would be interesting to explore fine-tuning of larger LoRA-adapters, especially since our fine-tuned

models did not perform any better on the training set than on the test set, which suggests that the LoRA-adapters were smaller than ideal, as discussed in Section 4.1.6. Another direction of future work would be to build and deploy VLM-based OCR engines for actual digitization efforts so that readers get to take advantage of OCR systems that are able to achieve lower error rates.

# Bibliography


[1] Kungl. Biblioteket, *Aftonbladet*, p. 4, Apr. 1831, ISSN: 1103-9000. Libris-ID: 4112678. Accessed: Aug. 11, 2025. [Online]. Available: https://tidningar.kb.se/hrtd9tqjfccjxv2q/part/1/page/4.

[2] R. Smith, "An overview of the tesseract OCR engine," in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, vol. 2, 2007, pp. 629–633. DOI: 10.1109/ICDAR.2007.4376991.

[3] *Abbyy FineReader PDF - User's Guide*, Abbyy (2023). Accessed: Nov. 10, 2025. [Online]. Available: https://help.abbyy.com/assets/en-us/finereader/16/Users_Guide.pdf.

[4] Kungl. Biblioteket, *Stockholms Dagblad*, p. 2, Nov. 1826, Libris-ID: 2811213. Accessed: Nov. 3, 2025. [Online]. Available: https://tidningar.kb.se/h1t7l9jt1g47sd2/part/1/page/2.

[5] "Grepect GmbH | Strukturierte Texterfassung, Digitalisierung, XML," *Texterfassung*, Accessed: Jan. 20, 2026. [Online]. Available: https://www.texterfassung.eu.

[6] C. Morand, A.-L. Ligozat, and A. Névéol, *How Green Can AI Be? A Study of Trends in Machine Learning Environmental Impacts*, Dec. 2024. DOI: 10.48550/arXiv.2412.17376. Accessed: Nov. 6, 2025.

[7] *Google Colaboratory*, Google (2025). Accessed: Nov. 6, 2025. [Online]. Available: https://colab.research.google.com/.

[8] D. Dannélls, L. Björk, O. Dirdal, and T. Johansson, "A Two-OCR Engine Method for Digitized Swedish Newspapers," English, Linköping Electronic Conference Proceedings 180, Tech. Rep., 2021, pp. 65–74. Accessed: Aug. 6, 2025. [Online]. Available: https://ecp.ep.liu.se/index.php/clarin/article/view/8/8.

[9] V. Löfgren, "New tools for old news," English, Department of Computer Science, Engineering, Chalmers University of Technology, and University of Gothenburg, Gothenburg, Sweden, Tech. Rep., 2023. Accessed: Aug. 6, 2025. [Online]. Available: https://odr.chalmers.se/items/27030b99-c9be-42f1-b566-dc103746a21c.

[10] L. Xue et al., "Byt5: Towards a Token-Free Future with Pre-trained Byte-to-Byte Models," *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 291–306, Mar. 2022, ISSN: 2307-387X. DOI: 10.1162/tacl_a_00461. Accessed: Nov. 24, 2025.

[11] A. Jasonarson, S. Steingrímsson, E. Freyr Sigurðsson, Á. Davíð Magnússon, and F. Ágúst Ingimundarson, "Generating Errors: OCR Post-Processing for

Icelandic," English, The Árni Magnússon Institute for Icelandic Studies, Iceland, Tech. Rep., 2023, pp. 286–291. Accessed: Aug. 6, 2025. [Online]. Available: https://aclanthology.org/2023.nodalida-1.29.pdf.

[12] *Kofax OmniPage Ultimate - User's Guide*, Kofax (2019). Accessed: Nov. 10, 2025. [Online]. Available: https://docshield.tungstenautomation.com/OPU/en_US/19.2.0-x8jtnwx31a/print/KofaxOmniPageUltimateUserGuide_EN.pdf.

[13] T. Breuel, "The OCRopus open source OCR system," vol. 6815, Jan. 2008, p. 68 150. DOI: 10.1117/12.783598.

[14] S. Bai et al., *Qwen2.5-vl technical report*, 2025. arXiv: 2502.13923 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2502.13923.

[15] A. Vaswani et al., "Attention is all you need," vol. 30, 2017. Accessed: Nov. 24, 2025. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[16] W. Kwon et al., "Efficient Memory Management for Large Language Model Serving with *PagedAttention*," ser. SOSP '23, Koblenz, Germany: Association for Computing Machinery, 2023, pp. 611–626, ISBN: 9798400702297. DOI: 10.1145/3600006.3613165. Accessed: Nov. 24, 2025.

[17] H. Chen et al., *Chatgpt's one-year anniversary: Are open-source large language models catching up?* 2024. arXiv: 2311.16989 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2311.16989.

[18] OpenAI et al., *Gpt-4 technical report*, 2024. arXiv: 2303.08774 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2303.08774.

[19] J. Wang, Y. Wang, L. Cheng, and Z. Zhong, "FakeSV-VLM: Taming VLM for detecting fake short-video news via progressive mixture-of-experts adapter," in *Findings of the Association for Computational Linguistics: EMNLP 2025*, C. Christodoulopoulos, T. Chakraborty, C. Rose, and V. Peng, Eds., Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 4782–4798, ISBN: 979-8-89176-335-7. DOI: 10.18653/v1/2025.findings-emnlp.257. [Online]. Available: https://aclanthology.org/2025.findings-emnlp.257/.

[20] D. Goswami, R. Subedi, and S. Chakraborty, "MediVLM: A vision language model for radiology report generation from medical images," in *Findings of the Association for Computational Linguistics: EMNLP 2025*, C. Christodoulopoulos, T. Chakraborty, C. Rose, and V. Peng, Eds., Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 10 287–10 304, ISBN: 979-8-89176-335-7. DOI: 10.18653/v1/2025.findings-emnlp.544. [Online]. Available: https://aclanthology.org/2025.findings-emnlp.544/.

[21] D. T. Tran, N. T. Pham, N. D. H. Nguyen, and B. Manavalan, "Mol2Lang-VLM: Vision- and text-guided generative pre-trained language models for advancing molecule captioning through multimodal fusion," in *Proceedings of the 1st Workshop on Language + Molecules (L+M 2024)*, C. Edwards, Q. Wang, M. Li, L. Zhao, T. Hope, and H. Ji, Eds., Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 97–102. DOI: 10.18653/v1/2024.langmol-1.12. [Online]. Available: https://aclanthology.org/2024.langmol-1.12/.

[22] C. Park, Y. Baek, J. Kim, Y.-J. Heo, D.-S. Chang, and J. Choo, "Evaluating visual and cultural interpretation: The k-viscuit benchmark with human-VLM collaboration," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, W. Che, J. Nabende, E. Shutova, and M. T. Pilehvar, Eds., Vienna, Austria: Association for Computational Linguistics, Jul. 2025, pp. 21 960–21 974, ISBN: 979-8-89176-251-0. DOI: 10.18653/v1/2025.acl-long.1066. [Online]. Available: https://aclanthology.org/2025.acl-long.1066/.

[23] M. Bauer, C. Seibold, J. Kleesiek, and A. Dada, "IKIM at MEDIQA-M3G 2024: Multilingual visual question-answering for dermatology through VLM fine-tuning and LLM translations," in *Proceedings of the 6th Clinical Natural Language Processing Workshop*, T. Naumann, A. Ben Abacha, S. Bethard, K. Roberts, and D. Bitterman, Eds., Mexico City, Mexico: Association for Computational Linguistics, Jun. 2024, pp. 439–447. DOI: 10.18653/v1/2024.clinicalnlp-1.44. [Online]. Available: https://aclanthology.org/2024.clinicalnlp-1.44/.

[24] T. Chakraborty, E. Caplan, and D. Goldwasser, "VIBE: Can a VLM read the room?" In *Findings of the Association for Computational Linguistics: EMNLP 2025*, C. Christodoulopoulos, T. Chakraborty, C. Rose, and V. Peng, Eds., Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 22 992–23 008, ISBN: 979-8-89176-335-7. DOI: 10.18653/v1/2025.findings-emnlp.1252. [Online]. Available: https://aclanthology.org/2025.findings-emnlp.1252/.

[25] J. Park et al., "R-VLM: Region-aware vision language model for precise GUI grounding," in *Findings of the Association for Computational Linguistics: ACL 2025*, W. Che, J. Nabende, E. Shutova, and M. T. Pilehvar, Eds., Vienna, Austria: Association for Computational Linguistics, Jul. 2025, pp. 9669–9685, ISBN: 979-8-89176-256-5. DOI: 10.18653/v1/2025.findings-acl.501. [Online]. Available: https://aclanthology.org/2025.findings-acl.501/.

[26] X. Bao, Z. Wang, J. Gu, and C.-R. Huang, "CalligraphicOCR for Chinese calligraphy recognition," in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, C. Christodoulopoulos, T. Chakraborty, C. Rose, and V. Peng, Eds., Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 4865–4877, ISBN: 979-8-89176-332-6. DOI: 10.18653/v1/2025.emnlp-main.245. [Online]. Available: https://aclanthology.org/2025.emnlp-main.245/.

[27] A. Kolavi, S. P, and V. Jain, "Nayana OCR: A scalable framework for document OCR in low-resource languages," in *Proceedings of the 1st Workshop on Language Models for Underserved Communities (LM4UC 2025)*, S. Truong et al., Eds., Albuquerque, New Mexico: Association for Computational Linguistics, May 2025, pp. 86–103, ISBN: 979-8-89176-242-8. DOI: 10.18653/v1/2025.lm4uc-1.11. [Online]. Available: https://aclanthology.org/2025.lm4uc-1.11/.

[28] F. Farsi, S. Shariati Motlagh, S. Bali, S. Sabouri, and S. Momtazi, "Persian in a court: Benchmarking VLMs in Persian multi-modal tasks," in *Proceedings of the First Workshop of Evaluation of Multi-Modal Generation*, W. E. Zhang et

al., Eds., Abu Dhabi, UAE: Association for Computational Linguistics, Jan. 2025, pp. 52–56. [Online]. Available: https://aclanthology.org/2025.evalmg-1.5/.

[29] S. Semnani, H. Zhang, X. He, M. Tekgurler, and M. Lam, "CHURRO: Making history readable with an open-weight large vision-language model for high-accuracy, low-cost historical text recognition," in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, C. Christodoulopoulos, T. Chakraborty, C. Rose, and V. Peng, Eds., Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 34 777–34 824, ISBN: 979-8-89176-332-6. DOI: 10.18653/v1/2025.emnlp-main.1763. [Online]. Available: https://aclanthology.org/2025.emnlp-main.1763/.

[30] S. Bai et al., *Qwen3-vl technical report*, 2025. arXiv: 2511.21631 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2511.21631.

[31] P. Wang et al., *Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution*, 2024. arXiv: 2409.12191 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2409.12191.

[32] A. Steiner et al., *Paligemma 2: A family of versatile vlms for transfer*, 2024. arXiv: 2412.03555 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2412.03555.

[33] G. Team et al., *Gemma 3 technical report*, 2025. arXiv: 2503.19786 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2503.19786.

[34] H. Liu, C. Li, Y. Li, and Y. J. Lee, *Improved baselines with visual instruction tuning*, 2024. arXiv: 2310.03744 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2310.03744.

[35] E. J. Hu et al., "LoRA: Low-rank adaptation of large language models.," *ICLR*, vol. 1, no. 2, p. 3, 2021. DOI: 10.48550/arXiv.2106.09685. Accessed: Oct. 30, 2025.

[36] Kungl. Biblioteket, *Hvad Nytt*, p. 4, Jun. 1889, Libris-ID: 2732042. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/s26ln5qhqqmss6gf/part/1/page/4.

[37] Kungl. Biblioteket, *Göteborgs Handels- och Sjöfartstidning*, p. 4, Nov. 1888, Libris-ID: 3678898, ISSN: 0345-4010. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/q82dzj82475g82f/part/1/page/4.

[38] Kungl. Biblioteket, *Dagligt Allehanda*, p. 4, Apr. 1829, Libris-ID: 2631189. Accessed: Jan. 6, 2026. [Online]. Available: https://tidningar.kb.se/sb4h57v4498424g/part/1/page/4.

[39] Kungl. Biblioteket, *Karlshamns Allehanda*, p. 2, Jul. 1873, Libris-ID: 2506033. Accessed: Jan. 6, 2026. [Online]. Available: https://tidningar.kb.se/q82gfxf24g614xr/part/1/page/2.

[40] Kungl. Biblioteket, *Wexjöbladet*, p. 2, Sep. 1835, Libris-ID: 2831177. Accessed: Jan. 6, 2026. [Online]. Available: https://tidningar.kb.se/n60fz84009f5f67/part/1/page/2.

[41] Kungl. Biblioteket, *Umebladet*, p. 4, Dec. 1861, Libris-ID: 2535033. Accessed: Jan. 6, 2026. [Online]. Available: https://tidningar.kb.se/q82g8qz25vmjnlk/part/1/page/4.

[42] Kungl. Biblioteket, *Hallandsposten*, p. 2, Feb. 1891, Libris-ID: 4112716, ISSN: 1103-9361. Accessed: Jan. 6, 2026. [Online]. Available: https://tidningar.kb.se/fpsbm16jcjfnk6sr/part/1/page/2.

[43] Kungl. Biblioteket, *Göteborgs Aftonblad*, p. 4, Oct. 1895, Libris-ID: 2647969. Accessed: Jan. 6, 2026. [Online]. Available: https://tidningar.kb.se/cwpz11cp5lqrd3j/part/1/page/4.

[44] "ABBYY FineReader 11 Boosts OCR Speed and Simplifies E-Book Creation," *ABBYY*, Aug. 2011. Accessed: Jan. 20, 2026. [Online]. Available: https://www.abbyy.com/company/news/abbyy-finereader-11-boosts-ocr-speed-and-simplifies-e-book-creation/.

[45] "Release Notes | Tesseract documentation," *Tesseract OCR Github*, Accessed: Jan. 20, 2026. [Online]. Available: https://tesseract-ocr.github.io/tessdoc/ReleaseNotes.

[46] W. Kwon et al., "vLLM | Vision Language," *Github*, 2023. [Online]. Available: https://github.com/vllm-project/vllm/blob/main/examples/offline_inference/vision_language.py.

[47] Kungl. Biblioteket, *Nya Dagligt Allehanda*, p. 4, Apr. 1894, Libris-ID: 2092229. Accessed: Jan. 6, 2026. [Online]. Available: https://tidningar.kb.se/k3w9n1pw1r2ll9r/part/1/page/4.

[48] Kungl. Biblioteket, *Sydsvenska Dagbladet*, p. 4, Jun. 1902, Libris-ID: 4112787, ISSN: 1104-0068. Accessed: Jan. 6, 2026. [Online]. Available: https://tidningar.kb.se/k3w9n1pw1r2ll9r/part/1/page/4.

[49] Kungl. Biblioteket, *Falköpings Tidning*, p. 4, Jun. 1865, Libris-ID: 4112699, ISSN: 1103-9191. Accessed: Jan. 6, 2026. [Online]. Available: https://tidningar.kb.se/tc5j61p50szc0xr/part/1/page/4.

[50] Kungl. Biblioteket, *Stockholmstidningen*, p. 4, May 1897, Libris-ID: 2812922. Accessed: Jan. 6, 2026. [Online]. Available: https://tidningar.kb.se/xg8qvwq83371t73/part/1/page/4.

[51] Kungl. Biblioteket, *Götheborgs Allehanda*, p. 4, May 1819, Libris-ID: 2625423. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/3mfvzptf300xnrg/part/1/page/4.

[52] Kungl. Biblioteket, *Götheborgs Tidningar*, pp. 2–4, Dec. 1822, Libris-ID: 2625488. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/dnr5cj7lbj2v3515/part/1/page/2.

[53] Kungl. Biblioteket, *Hvad Nytt*, p. 4, Dec. 1852, Libris-ID: 2732042. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/t3880x7fr0bmlfst/part/1/page/4.

[54] Kungl. Biblioteket, *Upsala*, p. 4, May 1867, Libris-ID: 2534858. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/k3w7ncbw0pw1bt8/part/1/page/4.

[55] Kungl. Biblioteket, *Härnösandposten*, p. 4, Sep. 1876, Libris-ID: 2794521. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/6qjwl0bj5k6j91d/part/1/page/4.

[56] Kungl. Biblioteket, *Götheborgska Nyheter*, p. 4, Oct. 1821, Libris-ID: 2627963. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/0jbr00gb38l22nm/part/1/page/4.

[57] Kungl. Biblioteket, *Post- och Inrikes Tidningar*, p. 2, Jun. 1825, Libris-ID: 8206782, ISSN: 1103-9892. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/4ngwchlg12lnp6c/part/1/page/2.

[58] Kungl. Biblioteket, *Jönköpingsbladet*, p. 4, May 1845, Libris-ID: 2746108. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/3mfsptkf5pvvt7x/part/1/page/4.

[59] Kungl. Biblioteket, *Nerikes Allehanda*, p. 2, Mar. 1846, Libris-ID: 8264853, ISSN: 1103-971X. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/8sl1b4hl3n1k17v/part/1/page/2.

[60] Kungl. Biblioteket, *Blekingsposten*, p. 4, Oct. 1859, Libris-ID: 4112687, ISSN: 1103-9094. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/tc5j6ms55drcf67/part/1/page/4.

[61] Kungl. Biblioteket, *Kalmar*, p. 4, Jun. 1872, Libris-ID: 2748542. Accessed: Jan. 5, 2026. [Online]. Available: https://tidningar.kb.se/bvn0kc7n4msf09r/part/1/page/4.

# A

# Appendix

## A.1 Complete Ground Truth and Model Outputs

This section will present the complete ground truth as well as the complete outputs generated by Swe19centOCR-8B, Swe19centOCR-2B, FineReader 11.1.16 and Tesseract 5.5.1. Note that all of the blank spaces, line breaks and empty rows are part of the generated outputs. The error rates generated by the newspaper clipping from *Nya Dagligt Allehanda* (1894) [47] are presented in Table A.1. Error rates generated by the newspaper clipping from *Sydsvenska Dagbladet* (1902) [48] are presented in Table A.2.

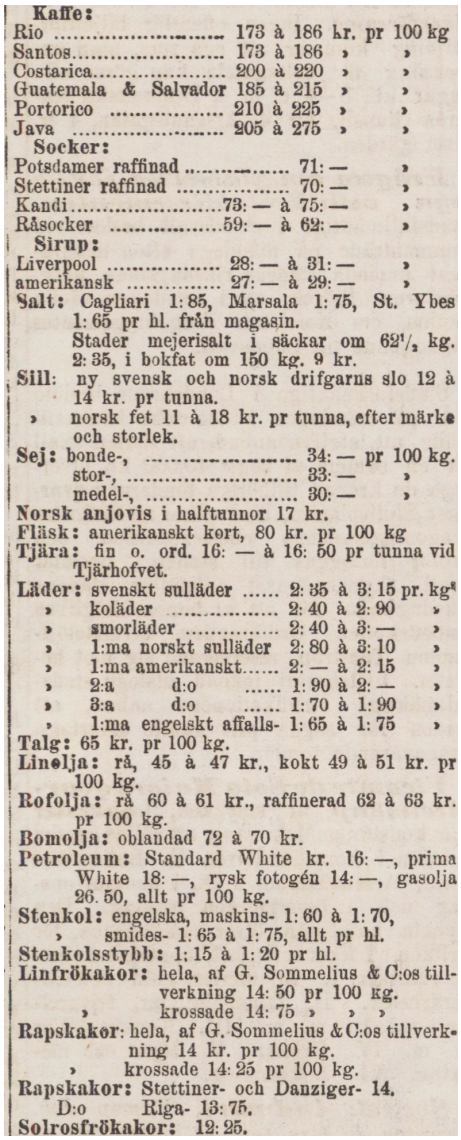
Table A.1: Error rates generated by newspaper clipping from *Nya Dagligt Allehanda*, 17th of April 1894 [47].

| **Model** | **CER (%)** | **WER (%)** |
|---|---|---|
| Swe19centOCR-8B | 7.516 | 39.855 |
| Swe19centOCR-2B | 22.595 | 39.614 |
| FineReader 11.1.16 | 22.786 | 31.401 |
| Tesseract 5.5.1 | 30.732 | 52.174 |

Table A.2: Error rates generated by newspaper clipping from *Sydsvenska Dagbladet*, 22nd of June 1902 [48].

| **Model** | **CER (%)** | **WER (%)** |
|---|---|---|
| Swe19centOCR-8B | 82.154 | 117.143 |
| Swe19centOCR-2B | 91.801 | 96.429 |
| FineReader 11.1.16 | 80.707 | 158.571 |
| Tesseract 5.5.1 | 83.119 | 129.286 |

# List from *Nya Dagligt Allehanda*, 1894



**Kaffe:**
Rio ..... 173 à 186 kr. pr 100 kg
Santos..... 173 à 186 » »
Costarica..... 200 à 220 » »
Guatemala & Salvador 185 à 215 » »
Portorico ..... 210 à 225 » »
Java ..... 205 à 275 » »

**Socker:**
Potsdamer raffinad ..... 71: — »
Stettiner raffinad ..... 70: — »
Kandi.....73: — à 75: — »
Råsocker .....59: — à 62: — »

**Sirup:**
Liverpool ..... 28: — à 31: — »
amerikansk ..... 27: — à 29: — »

**Salt:** Cagliari 1: 85, Marsala 1: 75, St. Ybes 1: 65 pr hl. från magasin.
Stader mejerisalt i säckar om 62½ kg. 2: 35, i bokfat om 150 kg. 9 kr.

**Sill:** ny svensk och norsk drifgarns slo 12 à 14 kr. pr tunna.
» norsk fet 11 à 18 kr. pr tunna, efter märke och storlek.

**Sej:** bonde-, ..... 34: — pr 100 kg.
stor-, ..... 33: — »
medel-, ..... 30: — »

**Norsk anjovis** i halftunnor 17 kr.

**Fläsk:** amerikanskt kort, 80 kr. pr 100 kg

**Tjära:** fin o. ord. 16: — à 16: 50 pr tunna vid Tjärhofvet.

**Läder:** svenskt sulläder ..... 2: 35 à 3: 15 pr. kg
» koläder ..... 2: 40 à 2: 90 »
» smorläder ..... 2: 40 à 3: — »
» 1:ma norskt sulläder 2: 80 à 3: 10 »
» 1:ma amerikanskt..... 2: — à 2: 15 »
» 2:a d:o ..... 1: 90 à 2: — »
» 3:a d:o ..... 1: 70 à 1: 90 »
» 1:ma engelskt affalls- 1: 65 à 1: 75 »

**Talg:** 65 kr. pr 100 kg.

**Linolja:** rå, 45 à 47 kr., kokt 49 à 51 kr. pr 100 kg.

**Rofolja:** rå 60 à 61 kr., raffinerad 62 à 63 kr. pr 100 kg.

**Bomolja:** oblandad 72 à 70 kr.

**Petroleum:** Standard White kr. 16: —, prima White 18: —, rysk fotogén 14: —, gasolja 26. 50, allt pr 100 kg.

**Stenkol:** engelska, maskins- 1: 60 à 1: 70,
» smides- 1: 65 à 1: 75, allt pr hl.

**Stenkolsstybb:** 1; 15 à 1: 20 pr hl.

**Linfrökakor:** hela, af G. Sommelius & C:os tillverkning 14: 50 pr 100 kg.
» krossade 14: 75 » » »

**Rapskakor:** hela, af G. Sommelius & C:os tillverkning 14 kr. pr 100 kg.
» krossade 14: 25 pr 100 kg.

**Rapskakor:** Stettiner- och Danziger- 14.
D:o Riga- 13: 75.

**Solrosfrökakor:** 12: 25.

Figure A.1: Newspaper clipping from *Nya Dagligt Allehanda*, 17th of April 1894 [47].

## Ground truth:

Kaffe:
Rio ..... 173 à 186 kr. pr 100 kg
Santos ..... 173 à 186 » »
Costarica ..... 200 à 220 » »
Guatemala & Salvador 185 à 215 » »
Portorico ..... 210 à 225 » »
Java ..... 205 à 275 » »
Socker:
Potsdamer raffinad ..... 71: — »
Stettiner raffinad ..... 70: — »
Kandi ..... 73: — à 75: — »

Råsocker ..... 59: — à 62: — »
Sirup:
Liverpool ..... 28: — à 31: — »
amerikansk ..... 27: — à 29: — »
Salt: Cagliari 1: 85, Marsala 1: 75, St. Ybes
1: 65 pr hl. från magasin.
Stader mejerisalt i säckar om 62½ kg.
2: 35, i bokfat om 150 kg. 9 kr.
Sill: ny svensk och norsk drifgarns slo 12 à
14 kr. pr tunna.
» norsk fet 11 à 18 kr. pr tunna, efter märke
och storlek.
Sej: bonde-, ..... 34: — pr 100 kg.
stor-, ..... 33: — »
medel-, ..... 30: — »
Norsk anjovis i halftunnor 17 kr.
Fläsk: amerikanskt kort, 80 kr. pr 100 kg
Tjära: fin o. ord. 16: — à 16: 50 pr tunna vid
Tjärhofvet.
Läder: svenskt sulläder ..... 2: 35 à 3: 15 pr. kg
» koläder ..... 2: 40 à 2: 90 »
» smorläder ..... 2: 40 à 3: — »
» 1:ma norskt sulläder 2: 80 à 3: 10 »
» 1:ma amerikanskt ..... 2: — à 2: 15 »
» 2:a d:o ..... 1: 90 à 2: — »
» 3:a d:o ..... 1: 70 à 1: 90 »
» 1:ma engelskt affalls- 1: 65 à 1: 75 »
Talg: 65 kr. pr 100 kg.
Linelja: rå, 45 à 47 kr., kokt 49 à 51 kr. pr
100 kg.
Rofolja: rå 60 à 61 kr., raffinerad 62 à 63 kr.
pr 100 kg.
Bomolja: oblandad 72 à 70 kr.
Petroleum: Standard White kr. 16: —, prima
White 18: —, rysk fotogén 14: —, gasolja
26. 50, allt pr 100 kg.
Stenkol: engelska, maskins- 1: 60 à 1: 70,
» smides- 1: 65 à 1: 75, allt pr hl.
Stenkolsstybb: 1: 15 à 1: 20 pr hl.
Linfrökakor: hela, af G. Sommelius & C:os till-
verkning 14: 50 pr 100 kg.
» krossade 14: 75 » » »
Rapskakor: hela, af G. Sommelius & C:os tillverk-
ning 14 kr. pr 100 kg.
» krossade 14: 25 pr 100 kg.
Rapskakor: Stettiner- och Danziger- 14.

D:o Riga- 13: 75.
Solrosfrökakor: 12: 25.

### Swe19centOCR-8B:

Kaffe:
Rio ..... 173 à 186 kr. pr 100 kg
Santos..... 173 à 186 ,
Costarica..... 200 à 220 ,
Guatemala & Salvador 185 à 215 ,
Portorico ..... 210 à 225 ,
Java ..... 205 à 275 ,
Socker:
Potsdamer raffinad ..... 71:— ,
Stettiner raffinad ..... 70:— ,
Kandi..... 73:— à 75:— ,
Råsocker ..... 59:— à 62:— ,
Sirup:
Liverpool ..... 28:— à 31:— ,
amerikansk ..... 27:— à 29:— ,
Salt: Cagliari 1:85, Marsala 1:75, St. Ybes
1:65 pr hl. från magasin.
Stader mejerisalt i säckar om 62½ kg.
2:35, i bokfat om 150 kg. 9 kr.
Sill: ny svensk och norsk drifgarns slo 12 à
14 kr. pr tunna.
, norsk fet 11 à 18 kr. pr tunna, efter märke
och storlek.
Sej: bonde-, ..... 34:— pr 100 kg.
stor-, ..... 33:— ,
medel-, ..... 30:— ,
Norsk anjovis i halftunnor 17 kr.
Fläsk: amerikanskt kort, 80 kr. pr 100 kg
Tjära: fin o. ord. 16:— à 16:50 pr tunna vid
Tjärhofvet.
Läder: svenskt sulläder ..... 2:35 à 3:15 pr. kg
, koläder ..... 2:40 à 2:90 ,
, smorläder ..... 2:40 à 3:— ,
, 1:ma norskt sulläder 2:80 à 3:10 ,
, 1:ma amerikanskt..... 2:— à 2:15 ,
, 2:a d:o ..... 1:90 à 2:— ,
, 3:a d:o ..... 1:70 à 1:90 ,
, 1:ma engelskt affalls- 1:65 à 1:75 ,
Talg: 65 kr. pr 100 kg.
Linölja: rå, 45 à 47 kr., kokt 49 à 51 kr. pr
100 kg.
Rofolja: rå 60 à 61 kr., raffinerad 62 à 63 kr.
pr 100 kg.

Bomolja: oblandad 72 à 70 kr.
Petroleum: Standard White kr. 16:—, prima
White 18:—, rysk fotogén 14:—, gasolja
26.50, allt pr 100 kg.
Stenkol: engelska, maskins- 1:60 à 1:70,
, smides- 1:65 à 1:75, allt pr hl.
Stenkolsstybb: 1:15 à 1:20 pr hl.
Linfrökor: hela, af G. Sommelius & C:os till-
verkning 14:50 pr 100 kg.
, krossade 14:75 ,
Rapskakor: hela, af G. Sommelius & C:os tillverk-
ning 14 kr. pr 100 kg.
, krossade 14:25 pr 100 kg.
Rapskakor: Stettiner- och Danziger- 14.
D:o Riga- 13:75.
Solrosfrökor: 12:25.

### Swe19centOCR-2B:

Kaffe:
Rio …… 173 à 186 kr. pr 100 kg
Santos…… 173 à 186
Costarica…… 200 à 220
Guatemala & Salvador 185 à 215
Portorico …… 210 à 225
Java …… 205 à 275
Socker:
Potsdamer raffinad …… 71:—
Stettiner raffinad …… 70:—
Kandi …… 73:— à 75:—
Råsocker …… 59:— à 62:—
Sirup:
Liverpool …… 28:— à 31:—
amerikansk …… 27:— à 29:—
Salt: Cagliari 1:85, Marsala 1:75, St. Ybes
1:65 pr hl. från magasin.
Stader mejerisalt i säckar om 621/2, kg.
2:35, i bokfat om 150 kg. 9 kr.
Sill: ny svensk och norsk drifgarns slo 12 à
14 kr. pr tunna.
> norsk fet 11 à 18 kr. pr tunna, efter märke
och storlek.
Sej: bonde-, …… 34:— pr 100 kg.
stor-, …… 33:—
medel-, …… 30:—
Norsk anjovis i halftunnor 17 kr.
Fläsk: amerikanskt kort, 80 kr. pr 100 kg
Tjära: fin o. ord. 16:— à 16: 50 pr tunna vid

Tjärhofvet.
Läder: svenskt sulläder …… 2:35 à 3:15 pr. kg
> koläder …… 2:40 à 2:90
> smorläder …… 2:40 à 3:—
> 1:ma norskt sulläder 2:80 à 3:10
> 1:ma amerikanskt…… 2:— à 2:15
> 2:a d:o …… 1:90 à 2:—
> 3:a d:o …… 1:70 à 1:90
> 1:ma engelskt affalls- 1:65 à 1:75
Talg: 65 kr. pr 100 kg.
Linölja: rå, 45 à 47 kr., kokt 49 à 51 kr. pr
100 kg.
Rofolja: rå 60 à 61 kr., raffinerad 62 à 63 kr.
pr 100 kg.
Bomolja: oblandad 72 à 70 kr.
Petroleum: Standard White kr. 16:—, prima
White 18:—, rysk fotogén 14:—, gasolja
26.50, allt pr 100 kg.
Stenkol: engelska, maskins- 1:60 à 1:70,
> smides- 1:65 à 1:75, allt pr hl.
Stenkolsstybb: 1:15 à 1:20 pr hl.
Linfrökkor: hela, af G. Sommelius & C:os till-
verkning 14:50 pr 100 kg.
> krossade 14:75 > > >
Rapskakor: hela, af G. Sommelius & C:os tillverk-
ning 14 kr. pr 100 kg.
> krossade 14:25 pr 100 kg.
Rapskakor: Stettiner- och Danziger- 14.
D:o Riga- 13:75.
Solrosfrökkor: 12:25.

### FineReader 11.1.16:

Kaffe:
: Rio 173 å 186 hr. pr 100 kg
Santos 173 å 186 » »
Costarica 200 å 220 > »
j Guatemala & Salvador 185 ä 215 > »
; Portorico 210 å 225 » »
; Java 205 å 275 » »
j Socker:
I Potsdamer raffinad 71: — »
Stettiner raffinad 70: — »
Kandi 73:— å 75:— >
Råsocker 59: — å 62: — »
i Sirup:
Liverpool 28: — å 31: — »
amerikansk 27: — å 29: — »

Salt: Cagliari 1:85, Marsala 1:76, St. Ybes
1: 65 pr hl. från magasin.
Stader mejerisalt i säckar om 62’/, kg.
2: 35, i bokfat om 150 kg. 9 kr.
( SiU: ny svensk och norsk drifgarns slo 12 å
14 kr. pr tunna.
» norsk fet 11 ä 18 kr. pr tunna, efter märk*
och storlek.
Sej: bonde-, 34: — pr 100 kg.
stor-, 33:— »
medel-, 30:— »
1 Norsk anjovis i halftunnor 17 kr.
Fläsk: amerikanskt kort, 80 kr. pr 100 kg
Tjära: fin o. ord. 16: — å 16: 60 pr tunna vid
Tjärhofvet.
Läder: svenskt sulläder 2:35 å 3:15 pr. kg1
» koläder 2: 40 å 2: 90 »
» smorläder 2: 40 å 3: — »
» l:ma norskt sulläder 2: 80 ä 3:10 »
» l:ma amerikanskt 2: — å 2:15 >
» 2:a d:0 1:90 ä 2: — »
> 3:a d:0 1:70 ä 1:90 »
» l:ma engelskt affalls- 1:65 ä 1: 75 »
Talg: 65 kr. pr 100 kg.
Linelja: rå, 45 ä 47 kr., kokt 49 å 51 kr. pr
100 kg.
i Rofolja: rå 60 å 61 kr., raffinerad 62 å 63 kr.
pr 100 kg.
i Bomolja: oblandad 72 å 70 kr.
Petroleum: Standard White kr. 16: —, prima
White 18:—, rysk fotogén 14:—, gasolja
26. 50, allt pr 100 kg.
; Stenkol: engelska, maskins- 1:60 ä 1:70,
1 > smides- 1: 65 å 1: 75, allt pr hl.
] Stenkolsstybb: 1:15 å 1:20 pr hl.
Linfrökakor: hela, af G. Sommelius k C:os till-
verkning 14: 50 pr 100 Kg.
» krossade 14:75 » » »
Rapskakor: hela, af G. Sommelius &0:os tillverk-
ning 14 kr. pr 100 kg.
> krossade 14: 25 pr 100 kg.
Rapskakor: Stettiner- och Danziger- 14.
D:o Riga- 13:75.
Solrosfrökakor: 12:25.

### Tesseract 5.5.1:

Kaffe:

HEBIO” 22.2... ess ecases 170 8 286 KE; DERE
, Santos... 5. 2 F8- 2.1865 »
i Costarica... GIS POE UED Ma 200 a 220 > >
Guatemala & Salvador 185 4a 215 > >
| Portorico ENIS GER 210 å 225 > »
FURVG: 2.4 2555200 HUD BR ID >
Socker:
| Potsdamer raffinad .....—..... 71: — »
Stettiner raffinad .....- 70: — »
ABE ASG KISA nte 73: =- å 75: – »
Rägocker 2 59: — a 62: — »
j Sirup:
EIVerpoOl sossesssansokonsne AD: = 4 31:— »
ameri ansk byn br BARS 27: — 4 29: – »

Salt: Cagliari 1:85, Marsala 1: 75, St. Ybes
1: 65 pr bl. från magasin.
Stader mejerisalt i säckar om 62’/, kg.

EA 2: 35, i bokfat om 150 kg. 9 k

iS ny svensk och norsk „Ebe Slo 12 a
14 kr. pr tunna.
Save fet 11 a 18 kr. pr tunna, efter märke
och storlek.

5 ram örten een on 33: — pr 100 kg.

Norsk a i halftunnor 17 kr.

Fläsk: amerikanskt kort, 80 kr. pr 100 kg
; Tjära: fin o. ord. 16: — ä 16: 50 pr tunna vid
Tjärhofvet.

Läder: (Svend er KA 2: 35
» KOSER 4.12.25 8
SMmorläder 002004040... 2:4
1:ma norskt Seiler 2
1:ma amerikanskt..... 2:
2:a MOE es 1
3:a LS SSA T:
1:ma engelskt affalls- 1:65 & 1: 75
Talg: 65 kr. pr 100 kg.
Fanes: - Ps å 47 kr., kokt 49 å 51 kr. pr

Rofoljaz, Ei 4 å 61 kr., raffinerad 62 å 68 kr.

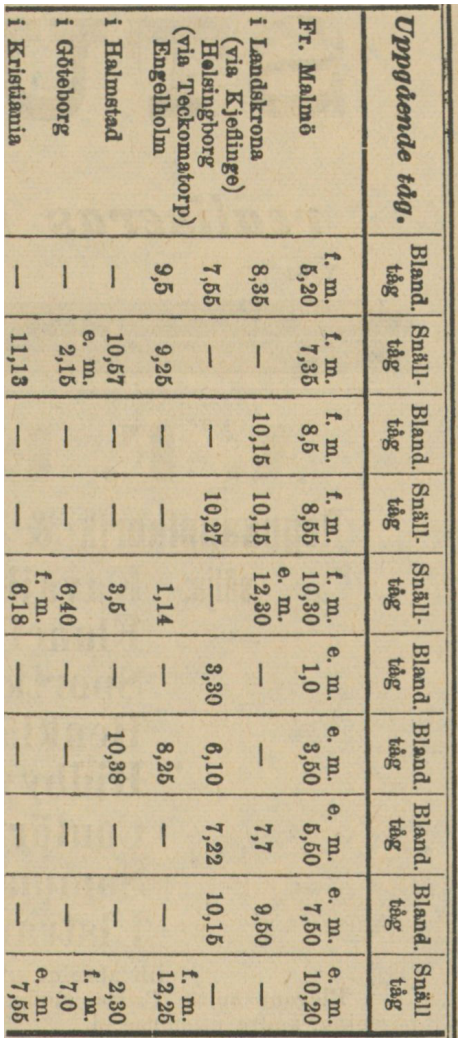
| Bomolja: ”obingäbe 72 4 70 kr.
| Nta Standard White kr. 16:—, prima

£0r för för gor för 10“ pr

vu vuv

White 18:—, rysk fotogen 14: —, ’gasolja
26. 50, allt pr 100 kg.
en be Mana maskins- 1:60 & 1: 70,
> smides- 1: 65 å 1:75, allt pr hl.
| Stenkolsstybb: 1: 15 a 1:20 pr hl.
Linfrökakor: hela, af G. Sommelius & C:os till-
verkning 14: 50 pr 100 kg.
krossade 14:75 > » >
Rapskakor: hela, af G.-Sommelius & C:0os tillverk«
ning 14 kr. pr 100 kg.
> krossade 14: 25 pr 100 kg.
Rapskakor: Stettiner- och Danziger- 14,
D:o Riga- 13: 75,
Solrosfrökakor: 18:25,

## Rotated table from *Sydsvenska Dagbladet*, 1902



| *Uppgående tåg.* | Bland. tåg | Snäll-tåg | Bland. tåg | Snäll-tåg | Snäll-tåg | Bland. tåg | Bland. tåg | Bland. tåg | Bland. tåg | Snäll tåg |
|---|---|---|---|---|---|---|---|---|---|---|
| | f. m. 5,20 | f. m. 7,35 | f. m. 8,5 | f. m. 8,55 | f. m. 10,30 / e. m. 12,30 | e. m. 1,0 | e. m. 3,50 | e. m. 5,50 | e. m. 7,50 | e. m. 10,20 |
| Fr. Malmö | | | | | | | | | | |
| i Landskrona (via Kjeflinge) | 8,35 | — | 10,15 | 10,15 | — | — | — | 7,7 | 9,50 | — |
| Helsingborg (via Teckomatorp) | 7,55 | — | — | 10,27 | — | 3,30 | 6,10 | 7,22 | 10,15 | — |
| Engelholm | 9,5 | 9,25 | — | — | 1,14 | — | 8,25 | — | — | f. m. 12,25 |
| i Halmstad | — | 10,57 / e. m. 2,15 | — | — | 3,5 | — | 10,38 | — | — | 2,30 / f. m. 7,0 / e. m. 7,55 |
| i Göteborg | — | — | — | — | 6,40 / f. m. 6,18 | — | — | — | — | — |
| i Kristiania | — | 11,13 | — | — | — | — | — | — | — | — |

Figure A.2: Newspaper clipping from *Sydsvenska Dagbladet*, 22nd of June 1902 [48].

### Ground truth:

Uppgående tåg. Bland. Snäll- Bland. Snäll- Snäll- Bland. Bland. Bland. Bland. Snäll
tåg tåg tåg tåg tåg tåg tåg tåg tåg tåg tåg
f. m. f. m. f. m. f. m. f. m. e. m. e. m. e. m. e. m. e. m.
Fr. Malmö 5,20 7,35 8,5 8,55 10,30 1,0 3,50 5,50 7,50 10,20
e. m.
i Landskrona 8,35 — 10,15 10,15 12,30 — — 7,7 9,50 —
(via Kjeflinge)
Helsingborg 7,55 — — 10,27 — 3,30 6,10 7,22 10,15 —
(via Teckomatorp) f. m.
Engelholm 9,5 9,25 — — 1,14 — 8,25 — — 12,25
i Halmstad — 10,57 — — 3,5 — 10,38 — — 2,30
e. m. f. m.
i Göteborg — 2,15 — — 6,40 — — — — 7,0

f. m. e. m.
i Kristiania — 11,13 — — 6,18 — — — — 7,55

**Swe19centOCR-8B:**

Uppgående tåg.
Bland. Snäll-
tåg tåg
Bland. Snäll-
tåg tåg
Bland. Snäll-
tåg tåg
Bland. Snäll-
tåg tåg
Bland. Snäll-
tåg tåg
Bland. Snäll-
tåg tåg
Fr. Malmö
f. m. f. m. f. m. f. m. f. m. f. m.
5,20 7,35 8,5 8,55 10,30 1,0
7,35 8,5 8,55 10,30 1,0 3,50
e. m. e. m. e. m. e. m. e. m. e. m.
1,0 3,50 5,50 7,50 10,20
i Landskrona
(via Kjellinge)
8,35 — 10,15 10,15 12,30 — 7,7 9,50 —
7,55 — 10,27 — 3,30 6,10 7,22 10,15 —
i Halmstad
— 10,57 — — 1,14 — 8,25 — —
9,5 9,25 — — 10,38 — — — —
i Göteborg
— 10,57 — — 1,14 — 8,25 — —
e. m. e. m. e. m. e. m. e. m. e. m.
2,15 — — — — — — — —
i Kristiania
— 11,13 — — 6,40 — — — —
f. m. f. m. f. m. f. m. f. m. f. m.
6,18 — — — — — — — —
e. m. e. m. e. m. e. m. e. m. e. m.
7,55 — — — — — — — —
i Göteborg
— 10,57 — — 1,14 — 8,25 — —
e. m. e. m. e. m. e. m. e. m. e. m.
2,15 — — — — — — — —
i Kristiania
— 11,13 — — 6,40 — — — —
f. m. f. m. f. m. f. m. f. m. f. m.

6,1

### Swe19centOCR-2B:

Uppgående tag.
Bland.
Small-
Bland.
Small-
Bland.
Bland.
Small-

### FineReader 11.1.16:

~ ~ ~ ^ g
111 ispi $ I
111 pifi I |
6 rgI|B *
S” A g;
►3 ^
»i» ta
I I I r* 0° 5* * £ 8
1 1 1 ©T Ot OO tO R (JQ P
03 03 O . p*
h* ® h* . 2?
H w o « I | Si ge
M Hg 03 M ’ ’ CO H CT3 ^
O» 03 - -JT 03 03. .
►-» M* -.g
I l I I l -2 -SB jfl
03 • PJ
I-* *-* ef-
I I I | O O 0° £ gj
1 to 03 g w {—
-4 03 03 - *7*
M» H* ® I—» M,
® M ® « r i “ $• g,
h* g 03 I-» 1 to p CO g
00 • o ^ o*o*
1 I I i £ i Zb Jfl
o pJ
I I ® | «
’ i co to i—* 1 ©3 B ^ o
00 03 o or p*
l l l I S tl Jfl
to o * CL
1111-22 J?!
03 O O • 3L

®m» ►—» M» CQ
• 4’ M M* | I O * & g.
ox g o B co tö g 1 \’ ”to g ora c
03 P r q ox r or ⌒

### Tesseract 5.5.1:

Bland.| Bland.| Bland.| Bland.| Snäll

Em Kjeflinge)

Helsingborg
(via Teckomatorp)