ICS 661: Final Project Report

https://github.com/DigitizeTextPLM/digitalize_text

Joel Nicolow, Amanda Nitta, Jan Mark Schittenhelm
2604-1114 2622-8976 2939-7662

1 Introduction

As society becomes more technologically centric, documents are transferring to be completely digitized. This leads for previous materials in file cabinets to be transcribed and compatible with digital storage, for viewing, sharing and editing. This has become more evident as the practices of hard-copy documents that were created using either the type-writer, or handwritten material are becoming out of style; especially with the limited usage of cursive in modern times. The goal of our work is to help improve digitizing information that was previously only hard-copy.

The task of transcribing and digitizing the contents of these hard-copy files is tedious and time-consuming. When considering photo-copying these texts, the images taken of documents may be of low-resolution, fading text, or different orientations. Specifically, in the case of handwritten text, it is often considered to be illegible between different individuals when interpreting materials. Furthermore, it is increasingly important to transcribe all documents to allow documents to be widely viewable and accessible. Optical Character Recognition (OCR) technologies have been developed in an attempt to parse text from photographs of documents. However, OCR has shortcomings that include not correctly recognizing and failing to transcribe text.

Our paper is structured with reference to handwritten data followed by the typed text dataset. The implementation and methodology sections are incorporated together (see section 4) for coherency followed by results, limitations, and conclusion.

As mentioned in the upcoming Related Work, the incorporation of deep learning with OCR have previously been used to transcribe and ensure accuracy of materials. With the emergence of Natural Language Processing (NLP) it allows for approaches that are more specific to correcting text and language output of the OCR. Specifically, with the incorporation of techniques such as fine-tuning and prompt-engineering large-language models with an emphasis on Pre-Trained language models (PLM). It allows for the digitization of texts to be more accurate and efficient. As a result, people can focus on finding and contextualizing the documents rather than transcribing. In our approaches, we use three open source datasets - IAM Database, Bentham Dataset, and a subset of ICDAR2015 to demonstrate the potential of NLP incorporation in transcribing handwritten hard copy documents.

2 Related Work

Previously, there were several efforts to transcribe previously handwritten to digital documents. These efforts are to digitize records of the past such as meteorological data in a standardized format using object character recognition (OCR) along with the usage of deep learning for fine-tuning (e.g, text recognition and layout detection) [8]. The usage of OCR has some difficulties in the recognition of older documents such as the digitizing of the Korean historical archives during the Lee dynasty; there are ancient characters that are no longer used in contemporary texts [7], requiring special expertise by domain experts to create a specific handwriting OCR system. These efforts have been ongoing for several years with the usage of text pre-processing, text detection, and deep learning [6]. In [6], the authors also note that on average a person can process 2,122 documents per month while an artificial intelligence (AI) model can process 108,000 documents per month. As more documents are desired to be digitized for record-keeping purposes, the usage of NLP techniques are important to further process a comprehensive understanding of individual handwriting.

Recent research highlights the success of using large language models (LLM) to improve OCR systems, credited to its ability to contextually

grasp the meaning of a sentence. [9]. There has also been an emphasis that recognition of handwritten data poses a challenge due to the inconsistency of the written characters and the lack of ground-truth training data. LLMs have also been used to post-process OCR output, a method similar to our initial approach, and achieved up to 60% error reduction on newspaper data [1]. This approach is based on the fact that correcting a sub-optimal OCR output has similarities to the masked token problem.

3 Data

3.1 Handwritten Datasets

To replicate the challenges of hard-copy documents, there were two handwritten datasets and one typed text dataset. The two handwritten datasets were used to incorporate cursive and print handwriting. All datasets were converted to have images as input and raw text as output (without positional information).

3.1.1 IAM Database

The IAM database curated by the University of Bern [4] includes 1,539 documents of scanned documents written by 657 writers; predominately written in a print format.

3.1.2 Bentham Dataset

The Bentham dataset is a collection of 433 documents written by renowned English philosopher and reformer Jeremy Bentham (1748-1832) hosted and maintained by University College London [10]. The dataset consists of 433 documents written in the hand of Bentham himself or his amanuensis and correspondents. All documents are written in cursive. Community engagement has been used to transcribe the dataset.

Table 1 displays descriptive statistics for the documents that the OCR tool was able to successfully parse some textual data. The size of the printed hand-written dataset is larger since as shown the sequences are smaller than the cursive dataset

3.2 SmartDoc Dataset

The International Conference on Document Analysis and Recognition (ICDAR) hosts challenges with data made publicly available. During the 2015 conference, a challenge titled SmartDoc was hosted [2]. The input for this challenge is an image of a hard-copy typed text document and the output is the text in the document. A subset of this dataset (3600 documents and ground truth texts) was downloaded.

4 Methodology

4.1 Data Pre-Processing

All the text was extracted using the Google Tesseract Open Source OCR tool [5]. However, the extraction process was slightly different for each dataset due to some formatting issues.

4.1.1 Hand-written Datasets

For the IAM database, when observing the whole page the ground truth was located on the same page with some cosmetic issues (e.g. Lines separating parts of the paper, different sizing of the images), which caused difficulties in only extracting the written text. Therefore, for the IAM database the handwritten lines were used for extracting the text and further concatenated to form a single string for each document/form. While the Bentham dataset text was parsed as a whole image through Tesseract.

Upon extracting the OCR text from the entries, the datasets were combined and randomized with a training, validation, and test split of 70%, 20%, and 10% respectively.

4.1.2 Typed Text Dataset

The SmartDoc dataset is comprised of high-resolution images to replicate real-world applications where image resolution is not ideal for OCR. Data was extracted from the smart documents at three resolutions; original resolution, $\frac{1}{4}$, and $\frac{1}{6}$ resolution.

4.2 Performance Metrics

There are several metrics that are used for NLP processes, however, we chose to only use cosine similarity as it compares the meaning of the output text and ground truth only. Other popular metrics in the NLP field have several limitations or are not directly relevant to the purpose of our study. For instance, BLEU score is a metric used when comparing reference text to machine-translated text; but it does not consider the meaning of the words which plays a role in our evaluation of whether cleaning of the OCR

Table 1: Character Statistics of Documents Compatible with OCR

Dataset	Modality	# Docs	Min	Max	Median
IAM	Printed Handwriting	1206	58	903	373.5
Betham	Cursive Handwriting	429	133	3452	1156
IAM & Bentham	Combined Handwritten	1635	58	3452	405
SmartDoc	Typed Text	3600	1253	5242	3557

output accurately finds the proper word. If it were BLEU score, it would be checking the accuracy of direct transcription which was not our primary goal. Similarly, ROGUE score uses word sequences as the main measure of similarity where although attempting to transcribe the information not all words next to each other are the same, which would of led to a misrepresentation of our results. Due to the reasons described and time constraints for the project, we had focused on cosine similarity as it was most aligned with our goals, but also lacks as a limitation of our work.

Thus to compare the performance of OCR to ground truth text cosine similarity was computed between embeddings of the predicted and ground truth text. BERT Base Uncased (110 million parameters) from Hugging Face's Transformers library [14] originally proposed by Devin et. al. [3] was used for text embeddings. This model was set to take sequences of length 512 tokens; sequences longer than 512 tokens were broken up then embeddings were max pooled to get an embedding for the entire sequence. Cosine similarity was computed using equation scikit-learn's `cosine_similarity` function [12].

Cosine similarity was used to create the loss function *1 - cosine similarity*. Due to numerical precision issues during training a cosine similarity of slightly greater than one occurred. Owing to this there is a conditional to control for a negative loss in Listing 1. While standard cross-entropy loss was first used Cosine similarity loss was later chosen given its proximity to the evaluation metric.

```
def cosine_sim_loss(cos_simil):
      if 1 - cos_simil < 0:
          return 0
      else
          return 1 - cos_simil
```

Listing 1: Cosine Similarity Loss function

4.3 Post Processing OCR Output

There were mistakes in the outputs of OCR across the different datasets. To reduce these errors pre-trained LLMs were used to process and correct mistakes in the OCR output. For the handwritten datasets, a pre-trained GPT2 [13] was fine-tuned, explained in section 4.3.1. For the typed text dataset Chat GPT3.5 turbo [11] was used to improve OCR output (see section 4.4).

4.3.1 LLM Fine-tuning

To improve the accuracy of the OCR text from the handwritten data, the open source PLM, GPT2 by OpenAI located on HuggingFace [13], was fine-tuned on the OCR as the input and ground truth data as the label. The GPT2 architecture includes 11 layers that were trained on Reddit outbound links. To fine-tune the model, there were several parameters that were optimized by hand to ensure the model was within the memory of the GPU and developed with the best evaluation metric.

Hardware for Fine-Tuning The hardware used was the high-performance computer cluster known as Koa. To train the model, the GPU requested was NV-H100, which has 80GB of GPU memory. This size allowed for the model and data to live on the same GPU. Other specifications requested were number of nodes (1), number of

tasks per node (1), number of cores per task (1), GB of RAM (50), and number of GPUs requested (1). To evaluate the test data on the model, the GPU requested was the NV-V100-SXM2 that has 32 GB of GPU memory.

Fine-tune the model The parameters in Table 2 were optimized to see the GPT-2 model that could continue to train within memory. Other parameters that remained constant to ensure that gradients would not disappear or vanish are as follow: number of epochs trained for (100), learning rate (5e-6), weight decay (0.001), fp16 (false), gradient accumulation steps (4), max grad norm (1.0). Early stopping was also implemented with a threshold of 0.01 of no decrease in the loss and patience of 3 based on the validation set.

Table 2: Optimized Parameters - Handwritten

Hyper Parameter	**Search Space**	**Best**
unfrozen layers	[1, 2]	1
batch size	[1, 2, 3]	3
max length	[500, 600, 700, 800, 1000]	700

During fine-tuning we used cosine similarity as the performance metric (see section 4.2); the loss function was the cosine similarity loss as shown in Listing 1. Note, the embeddings used for cosine similarity calculation were not from the model being fine-tuned they were from a BERT model (see section 4.2).

4.4 LLM Prompt Engineering

For the SmartDoc dataset, OCR was computed at three image resolutions (see section 4.1.2). Chat GPT3.5 turbo was then used to correct the OCR output. Four prompts (two zero shot, a one shot, and a few shot) were written and compared on a validation set comprised of 50% of the downloaded data (1800 documents). The remaining 50% of the dataset was used to evaluate the prompt which performed best on the validation set. All four prompts are shown below. The example prompts used for one and few shot were randomized and were not from the dataset that the model was being evaluated on. If the model was evaluated on the validation set the examples were from test set. Which examples were used for the one and few shot prompts were not analytically optimized.

prompt one: Dont add anything just clean grammar and fill missinmpting was not analytically optimized.g text. then return only the cleaned text: `raw OCR output`

prompt two: The following test is from ocr. There are some errors please fix them and return the text prompt with no additional correspondence: `raw OCR output`

prompt three: The following test is from ocr. There are some errors please fix them and return the text with no additional correspondence. Here is an example: raw: `raw OCR output example` cleaned: `cleaned OCR output example` now do that for this: `raw OCR output`

prompt four: The following test is from ocr. There are some errors please fix them and return the text with no additional correspondence. Here are some examples: raw: raw OCR output example cleaned: `cleaned OCR output example` raw: `raw OCR output example` cleaned: `cleaned OCR output example` raw: `raw OCR output example` cleaned: `cleaned OCR output example` now do that for this: `raw OCR output`

5 Experiment and Results

As mentioned in section 4.2 cosine similarity was used to evaluate the performance of the OCR combined with LLM post processing.

5.1 Handwritten Datasets

When comparing the OCR output to the ground truth as shown in Listing 2, based on human observations, it is hard to determine how the OCR got to its conclusions and looks like a random set of words. This could be attributed to OCR mainly because it was developed to parse printed or typed text data and not handwritten.

The results from the correcting of text on the fine-tuned GPT2 model are displayed in Table 3. It can be observed that there was a decrease in the performance of the cosine similarity of the fixed OCR compared to evaluation of the raw OCR.

Error Analysis When observing the fine-tuning of the pre-trained GPT2 model it could be seen that the loss for the training and validation was decreasing, indicating that the model was learning (Figure 1) the patterns in the OCR to match the ground truth data.

```
Ground Truth:
A MOVE to stop Mr. Gaitskell from
↪ nominating any more Labour life
↪ Peers is tobe made at a meeting
↪ of Labour M Ps tomorrow. Mr.
↪ Michael Foot has put downa
↪ resolution on the subject and he
↪ is to be backed by Mr. Will
↪ Griffiths, M P forManchester
↪ Exchange.

OCR:
Gufs, WP Lar Mancerasteys Creotonge |
↪ pve Aowu a vesotilior on Lhe
↪ DQ'AQ Ve Onm 2 wo to VA. backeR
↪ by Mr. Wiu MOL WON ug OA WOKR
↪ LoBowr Ute. § Faoro Lk MOVE to
↪ stoe Mr. Gots keQQ fron LO  Lo
↪ Qo_ AA MPs  douocrreno, My.
↪ Mickrac\  'took Hay
```

Listing 2: Example of OCR Text

However, despite the model learning, the output was not at all close to the expected ground truth of coherent words and phrases. When performing the evaluation of cosine similarity, the decoded labels and decoded predictions were used to ensure proper comparison. As shown in Listing 3, it can be observed that the tokenizer's low max length of only 700 tokens led to the meaning of the words to be lost. The low max length was due to memory issues in the GPU. If the max length was higher, the kernel would either die or print errors and terminate denoting that there was not enough memory to continue.

The OCR output contains random characters and it is nearly impossible to decipher any words since the OCR is made for printed typed data. This proposes a further challenge to solve, even for humans.

Table 3: Results for Handwritten Model Fine-Tuning

Dataset	Raw OCR	GPT Cleaned
Validation	**73.39%**	60.92%
Test	**73.83%**	57.12%

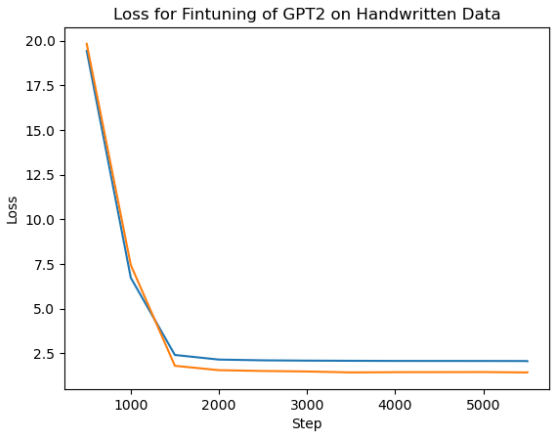

Figure 1: Loss Curve

To continue exploring challenging scenarios of transcipition. A dataset with images of text was modified to have various resolutions and then processed with Tesseract, to represent challenging situations. As described below in the limitations, fine-tuning models had led to hardware limitations, so we decided to explore another NLP technique - prompt engineering to see how it may provide assistance in correcting text.

5.2 Typed Text Dataset

As shown in Table 4 and in Figures 2 & 3 the performance of OCR decreases as image resolution decreases. Post-processing of OCR outputs can increase performance at lower resolutions.

Out of the four prompts compared in conjunction with Chat GPT3.5 turbo the few shot prompts performed best on the validation set for the three image resolutions as shown in Figure 2. This highest-performing prompt was then evaluated on the test set and compared with the raw OCR output in Figure 3.

Table 4: Cosine Similarity on Test Set

Dataset	Raw OCR	Post-proccessed OCR
full resolution	**95.2%**	94.3%
$\frac{1}{4}$ resolution	77.1 %	**85.6%**
$\frac{1}{6}$ resolution	64.9%	**76.8%**

```
Ground Truth Decoded:
 oo t
 the m^^
 a t t t'
 d^ a be ly
^^^^^^

 fo^^^ fo^ t^^^ fo^ fo t^^ o^^^ o
^^^^ fo^^^ fo^ fo^^^ fo^^^ fo fo^
↪  fo^^^^^ fo fo fo^^^^^^ vo fo^ vo
↪  fo fo^^^^ the^ a fo fo fo fo^^
↪  fo^^ fo the^^^ fo^ fo^^^ fo^^ fo
↪  fo^^ vo fo^ fo fo fo fo t vo^^ fo

Fine-Tuned OCR decoded:
 oo t
 the m^^
 a t t t'
 d^ a be ly
^^^^^^

 fo^^^ fo^ t^^^ fo^ fo t^^ o^^^ o
^^^^ fo^^^ fo^ fo^^^ fo^^^ fo fo^
↪  fo^^^^^ fo fo fo^^^^^^ vo fo^ vo
↪  fo fo^^^^ the^ a fo fo fo fo^^
↪  fo^^ fo the^^^ fo^ fo^^^ fo^^ fo
↪  fo^^ vo fo^ fo fo fo fo t vo^^ fo
```

Listing 3: Decoded Example

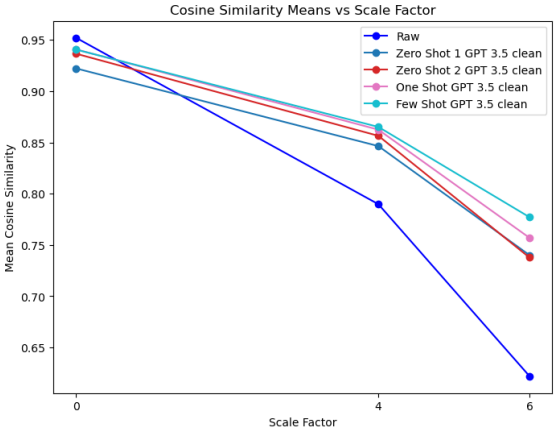

Figure 2: Prompt Performance Comparison on Validation Set

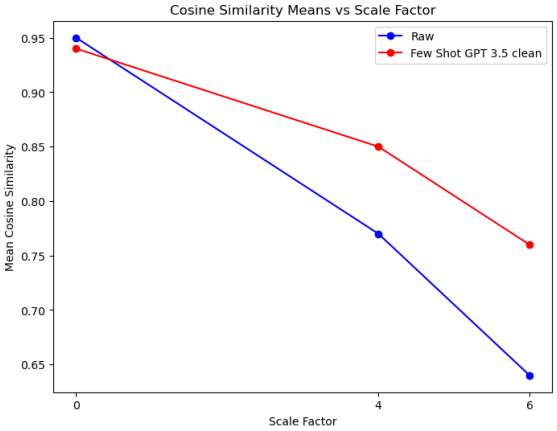

Figure 3: Prompt Performance Comparison on Test Set

6 Limitations

The application of NLP techniques relative to document rendering needs further exploration. There are several limitations that were encountered.

6.1 Fine-Tuning of PLM

For fine-tuning of the PLM challenges encountered were primarily attributed to hardware constraints. There was an attempt to fine-tune other models such as Llama3, however, due to the large size of the model and the tracking of gradients during training there was barely enough memory on one of the largest GPUs in the High-Performance Computing cluster, known more commonly as Koa, to train as shown in Listing 4. There was also an attempt to distribute the model across multiple GPUs but then there were constraints such as needing the data and the model on the same GPU. It was also attempted to distribute on all the GPUs through using frameworks such as accelerate, but there continued to be issues. Similarly, due to memory constraints, it only allowed the tokenizer to have a max length of 700 tokens before memory allocation or cache errors occurred. The size of each of the documents has a wide range, as shown in Table 1 and needs a substantial amount of tokens to learn the pattern correlations between the OCR and ground truth data. The effects of this limitation is directly displayed in Listing 3 as it shows the tokenizer was not able to save the proper coherent words or sequences leading for the model to not create the coherent words either. Further upon evaluation, it can be observed that this led to the model performing worse than the comparison of the raw OCR to the ground truth data. Furthermore, future work to improve generality of coher-

ent words and phrases from OCR is to increase the max number of tokens as well as train on a GPU with more memory.

```
GPU 0: NVIDIA H100 PCIe
  Memory Allocated: 1350.56 MiB
  Memory Cached: 71148.00 MiB
  Total Memory: 81116.69 MiB
```

Listing 4: Memory of GPU after pre-training GPT2

6.2 LLM Prompt Engineering

Only four prompts were evaluated when post-processing OCR output on the typed text dataset. In addition for the one and few shot prompts which examples were used in the prompt was not analytically optimized. As shown in Figures 2 & 3 post-processing OCR output via this approach increased performance for lower image resolution but slightly decreased performance on the high-resolution imagery.

7 Threats to Validity

Although the goal was to use the OCR tool Tesseract to extract the handwritten text, it is important to note that OCR was primarily developed to read printed text images rather than handwritten images. Thus this research demonstrates the challenges that OCR has in reading the handwritten data as well as the difficulties in transcribing it. It also demonstrates how tasks that are considered hard for humans also is hard for pre-trained language models.

8 Conclusion

As previously used methodologies to document information are beginning to pivot to digitization, innovative ways to digitize information are necessary which ensure all information is preserved for future use.

Our work suggests that the application of NLP techniques such as discussed in this document can increase the validity and performance of text extraction techniques for images. Although the use of Chat GPT3.5 to post-process OCR results on typed text documents increased performance for lower resolution imagery. OCR output for handwritten text did not represent the text and post-processing outputs with LLMs is not a solution. Further exploration is needed to ensure information in handwritten documents are digitized properly. While there are limitations to this work it provides an overview of how NLP techniques can be applied to these problems and assist in more effective and efficient transcription compared to manually.

9 Contributions

Joel Nicolow

- Developed script for cosine similarity.
- Located potential text datasets
- OCR for text datasets (computed at various resolutions)
- Few Shot prompting for OCR of the low resolution imagese
- Writing Presentation
- Writing final report

Amanda Nitta

- Scoping project idea
- Locating potential handwritten datasets
- Developing OCR Tesseract methods/ script to parse images for data and ground truth
- Developed script for loss of cosine similarity
- Parsed OCR handwritten data and randomized into train/validation/test
- Fine-Tuned pre-trained GPT2 on handwritten text data
- Writing Presentation
- Writing final report

Jan Mark Schittenhelm

- Conducted literature review of related works
- Fine-tuned GPT-2 model on handwritten data
- Writing Presentation
- Writing Final Report

References

[1] Bourne, J. Clocr-c: Context leveraging ocr correction with pre-trained language models. *arXiv preprint arXiv:2408.17428* (2024).

[2] Burie, J.-C., Chazalon, J., Coustaty, M., Eskenazi, S., Luqman, M. M., Mehri, M., Nayef, N., OGIER, J.-M., Prum, S., and Rusinol, M. Icdar2015 competition on smartphone document capture and ocr (smartdoc) - challenge 2 [data set]. *International Conference on Document Analysis and Recognition* (2015).

[3] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018. Available: `https://github.com/google-research/bert`.

[4] Fischer, A., and Bunke, H. Iam handwriting database, 1999.

[5] Google. Tesseract ocr. `https://github.com/tesseract-ocr/tesseract`, 2024.

[6] Gunaydin, E., Gencturk, B., Ergen, C., and KÖKLÜ, M. Digitization and archiving of company invoices using deep learning and text recognition-processing techniques. *Intelligent Methods In Engineering Sciences 2*, 4 (2023), 90–101.

[7] Kim, M.-S., Jang, M.-D., Choi, H.-I., Rhee, T.-H., Kim, J.-H., and Kwag, H.-K. Digitalizing scheme of handwritten hanja historical documents. In *First International Workshop on Document Image Analysis for Libraries, 2004. Proceedings.* (2004), IEEE, pp. 321–327.

[8] Lehenmeier, C., Burghardt, M., and Mischka, B. Layout detection and table recognition – recent challenges in digitizing historical documents and handwritten tabular data. *Digital Libraries for Open Knowledge (TPDL 2020) 12246* (2020), 229–242.

[9] Liu, Y., Li, Z., Yang, B., Li, C., Yin, X., Liu, C.-l., Jin, L., and Bai, X. On the hidden mystery of ocr in large multimodal models. *arXiv preprint arXiv:2305.07895* (2023).

[10] London, U. C. Bentham paper archieve, 2016.

[11] OpenAI. Chatgpt: Optimizing language models for dialogue, 2023. Model: GPT-3.5 Turbo, accessed via OpenAI API.

[12] Pedregosa, F., Varoququx, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Muller, A., Nothman, J., Louppe, G., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research 12*, Oct (2011), 2825–2830.

[13] Radford, A., Wu, J., Child, R., Lum, D., Amodel, D., and Sutskever, I. Language models are unsupervised multitask learners. *International Conference on Document Analysis and Recognition* (2015).

[14] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. Transformers: State-of-the-art natural language processing for pytorch and tensorflow. *arXiv preprint arXiv:1910.03771* (2019).