Integration of Deep Computer Vision Foundation Models for Document Analysis

Enhancing Optical Character Recognition Using an OCR-free Transformer Model

Department of Computer Science
OST – University of Applied Sciences
Campus Rapperswil-Jona

Autumn Term 2023

Author Anastasiia Graftceva
Advisor Prof. Dr. Marco Lehmann

Table of Content

Table of Figures

Abstract

This study explores the efficacy of a pre-trained transformer model from the open source Hugging Face Library applied in the domain of Optical Character Recognition (OCR), specifically to the task of extraction of dates from scanned documents.

Initially, OCR technology concentrated on recognizing patterns, using algorithms based on specific rules, to identify letters and numbers through their distinct shapes. Deep learning greatly improved accuracy and the ability to work with more nuanced text and complex layouts, which in combination with Large Language Models (LLMs) has made visual document understanding possible.

Approach: The conventional OCR approach follows two steps: First, one would OCR a scanned document with the help of an OCR engine like Tesseract, and then process the output using pattern matching and regular expressions, or, alternatively, a LLM trained for the specific field of application. A major limitation of OCR engines, however, lies in their generic nature, which often brings challenges in accuracy and efficiency.

The OCR-free or pseudo-OCR approach instead relies on a single encoder-decoder transformer model which integrates the aforementioned two steps, making it an end-to-end solution which can be adjusted and fine-tuned for a specific field of application.

For this project I selected the OCR-free Document Understanding Transformer model (Donut) which was initially pre-trained on an extensive and varied collection of documents. I then fine-tuned it on a targeted datasets of diverse sizes to find out model's ability to read, understand and extract dates from images. I evaluated the results based on accuracy and the model's adaptability to different document types and qualities, as well as different date formats.

Conclusion: The results of the study are encouraging, achieving an average accuracy of 75% on the somewhat limited training and test datasets meticulously assembled for fine-tuning. The OCR-free approach undoubtedly shows promise in performing atomic tasks on images such as extracting dates. However, its efficacy could be significantly enhanced by incorporating a wider variety of document types and date formats. Additionally, adapting it to manage scenarios with zero, one, or multiple dates in a single image is likely necessary. Data engineering has emerged as a crucial element, even in this proof-of-concept stage.

Management Summary

Overview

In this summary I present the findings of my SA-project focused on the application of an advanced, pre-trained deep learning model, Documents Understanding Transformer (Donut), for Visual Document Understanding (VDU). Distinctively, Donut is designed to perform tasks traditionally associated with Optical Character Recognition (OCR) but operates without the conventional mechanisms of OCR systems.

Objective of the study

The primary goal of this project is to assess Donut effectiveness in performing atomic, pattern recognition tasks, specifically identifying and extracting dates from images and document scans. The study aimed to demonstrate the capabilities of an OCR-free model in performing OCR tasks of high complexity. This task is particularly challenging due to the multitude of date formats and document layouts.

Key Findings

The following key findings were obtained during the study:

- Model's Proficiency in Date Recognition: The Donut model, despite being OCR-free, demonstrated a notable proficiency in recognizing dates within documents. This indicates its potential as a valuable tool in automated document processing and analysis.
- Challenges with Specific Date Formats and Numerals: Despite its overall effectiveness, the model encountered difficulties with certain date formats, document layouts and numeral representations of the dates. This limitation is primarily due to the nature of the training datasets, which might not encompass the diversity of date formats encountered in practical applications, including human errors and inconsistencies.
- Alignment with Original Hypothesis: The findings align with the study's hypothesis, indicating that while the Donut model is proficient in basic date recognition tasks, its performance can be further enhanced with a more comprehensive training approach including large corpora of relevant data.

Methodology

The project was conducted in a structured manner, starting with the acquisition of the pre-trained Donut base model[1]. Subsequently, I subjected the model to a fine-tuning process using targeted, more diverse datasets without adjustments of the model's architecture. These datasets consisted of images of receipts and applica-

[1] Donut: naver-clova-ix/donut-base at https://huggingface.co/naver-clova-ix/donut-base.

tion forms in four European languages, and contain dates in various formats along with annotations. These datasets were specifically re-engineered to enhance the model's ability to recognize and extract dates accurately.

Implications and Recommendations

- Significance of VDU in Document Processing: VDU plays an extremely important role in handling the large volumes of scanned and archived documents. The OCR-free nature of the Donut model adds a new dimension to this field, offering a more streamlined and efficient approach to document analysis compared to traditional OCR systems that involve two steps, an OCR engine followed by an extraction or understanding process.
- Potential of OCR-Free Transformer Models: The study demonstrates the potential of OCR-free transformer models like Donut in specific, atomic tasks like pattern recognition and extraction. This capability makes such models highly valuable for document analysis, especially in areas where traditional OCR systems may fall short.
- Future Research and Development Directions: future studies should prioritize fine-tuning the mentioned models on targeted diverse datasets, with a focus on a minimum of 1000 images per language and document type. This approach is expected to significantly improve the accuracy of predictions. Additionally, adjusting the model's architecture is crucial for handling more complex tasks. This includes the ability to identify multiple instances per page and accurately return their localization on the input image, potentially using techniques like bounding boxes. Such advancements will broaden the practical applications and efficiency of these models in various document processing tasks.

In conclusion, the Donut model represents a significant advancement in the field of VDU, showcasing the feasibility and effectiveness of OCR-free models in handling tasks reliant on conventional OCR systems, with potential to process and analyze documents. However, based on my study, I think that to fully realize the capabilities of models like Donut, further research and development are necessary, particularly in expanding and diversifying the training datasets and with possible adjustments of the models architecture as well as its performance evaluation.

Introduction

In the rapidly changing world of information technology and data science, Visual Document Understanding (VDU), a Deep Learning (DL) approach to the interpretation and analysis of the content of documents, has become a field of interest due to its ability to process and interpret large quantities of unstructured data, predominantly found in scanned or photographed documents. This ability plays an important role in transforming unstructured data into structured, analyzable formats.

VDU has the potential to enhance automation and efficiency in various industries. In sectors like finance, healthcare, and legal services, document handling is a necessity but it is laborious and error prone. VDU introduces operational improvements in document processing thereby enhancing productivity and accuracy, for instance with the streamlining of tasks through automation, and the ability to classify and query documents.

VDU stands at the forefront of recent developments in the fields of artificial intelligence (AI) and machine learning (ML), with the integration of state-of-the-art generative general-purpose architectures for processing and understanding natural languages. Hugging Face Transformers is an open-source library that offers a vast collection of pre-trained models for Natural Language Processing (NLP), providing tools for tasks like text classification, translation, summarization, and question answering. This library is widely used by AI community for its ability to integrate advanced VDU capabilities into various applications.

The relevance of VDU is evident across multiple applications, from digitizing historical records to optimizing business workflows and ensuring regulatory compliance. In the era of Big Data, the capability to accurately and swiftly process document-based information is invaluable.

Nevertheless, VDU faces challenges, particularly in processing documents with complex layouts, such as columns, tables, boxes, or embedded images and the document formats. Issues with document design variability, image quality, and language diversity make it a complex problem to solve mostly due to misinterpretation of the structure, leading to errors in text extraction.

Traditionally, VDU challenges have been addressed by combining Optical Character Recognition (OCR) outputs with visual encodings, but this approach has limitations. Tesseract and similar OCR engines, while effective for basic text recognition, often fall short in complex scenarios. They struggle with intricate document layouts, low-quality images, and diverse fonts. Additionally, they lack the contextual and semantic understanding crucial for modern VDU tasks.

In contrast, end-to-end pre-trained transformer models from Hugging Face (HF) Library[2], offer new options in the field. These models, unlike traditional OCR engines, stand out in accuracy and robustness, particularly with complex layouts and various text formats. They combine DL and NLP to recognize text as well as understand its context within the document. This ability to process unstructured data and adapt to a wide range of languages and fonts sets them apart from traditional methods.
Furthermore, transformers continuously learn and improve, adapting to new data, a feature absent in static traditional OCR engines. They also offer better integration with modern systems and are scalable, addressing the evolving needs of VDU tasks more effectively. While these advanced models require more computational resources, their efficiency in real-time processing and adaptability make them a more fitting choice for complex VDU applications.
However, a comprehensive exploration of integrating advanced transformers as complete substitutes for traditional OCR engines remains limited. Most current methodologies employ transformer models alongside OCR outputs, not as standalone solutions. This gap presents an opportunity to expand knowledge by examining the potential of transformers to process and understand document images without relying on conventional OCR engines at all. Trials with specific tasks, such as date identification, could assess the feasibility, efficiency, and accuracy of transformer models as comprehensive solutions for text extraction and document understanding.

Researchers from Naver CLOVA[3] AI team have developed an innovative end-to-end OCR-free VDU solution using an encoder-decoder transformer model architecture, now accessible through the HF library. Documents Understanding Transformer (Donut) encodes images, segmented into patches using a SWIN (Shifted Window) transformer, into token vectors. The token vectors are then decoded into structured sequences, further parseable into, for instance, JSON, using the BART (Bidirectional and Auto-Regressive Transformers) as a decoder, to generate output based on task prompts.

In my study I systematically explore the capabilities and limitations of Donut in the field of VDU in extracting dates, a basic but recurrent atomic task. In the Methods Chapter I explain the methodology used for the scope of the study:
1. Using the Pre-trained and fine-tuned Donut Model for Date Extraction
This section evaluates effectiveness of Donut, generally fine-tuned on diverse datasets for text parsing to extract only dates from documents. It critically analyzes

[2] Hugging Face open source library at https://huggingface.co/models?pipeline_tag=image-to-text&sort=trending

[3] Naver CLOVA on Hugging Face at https://huggingface.co/naver-clova-ix

the model's basic date identification capabilities and identifies key limitations, including diverse date format handling and extraction accuracy.
2. Building a Pipeline to Fine-Tune Donut-base Model
Addressing the first section's findings, this part describes the development of a pipeline for fine-tuning the Donut model using targeted datasets. It details the dataset modifications and optimizations made to better align the Donut-base model with the task of varied date format recognition and improved accuracy.
3. Fine-Tuning Donut-base on Various Datasets and Comparative Analysis
The final section presents an in-depth comparative analysis of results from fine-tuning Donut variants across various datasets. It aims to identify the most effective fine-tuning strategies and understand how different datasets impact the model's performance in date extraction tasks.

This study contributes to the broader understanding of applying advanced transformers in document understanding, highlighting the enhanced capabilities of the Donut model and opening paths for future research and development in VDU area of study.

Methods

Traditional OCR follows a multi-step process (Fig. 1). Initially, an image undergoes preparation, which includes modifying brightness and contrast, noise reduction, correcting any skewing, and segmenting it to pinpoint text-containing areas. After that the OCR system identifies these text-containing areas, recognizing the document's layout elements such as columns, paragraphs, headings, and specific text sections. Advanced OCRs are capable of discerning more complex structures like tables and lists.

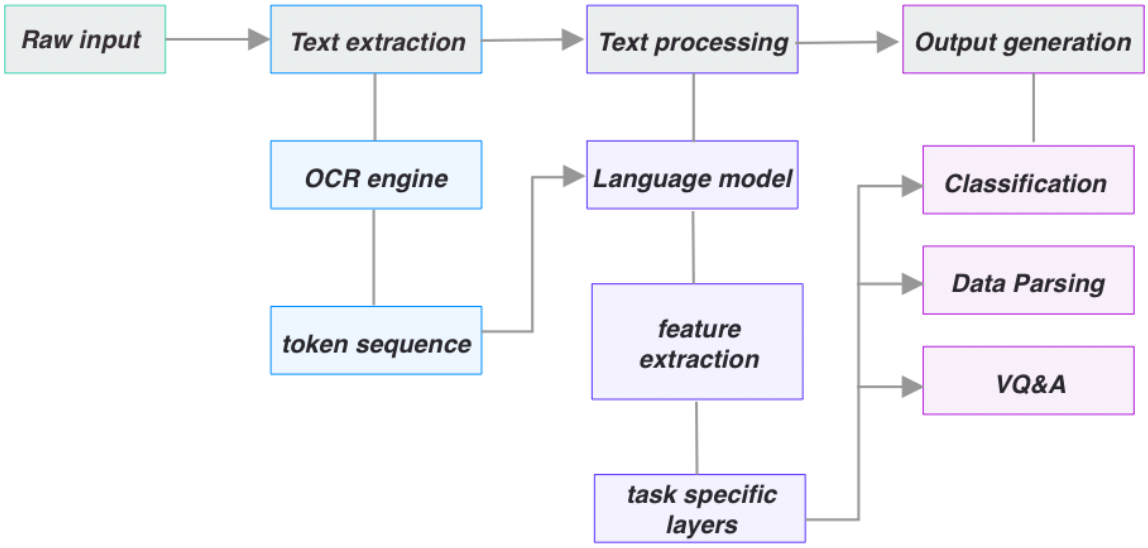

Figure 1 - Illustrated steps using traditional OCR solutions

In the essential phase of OCR is character recognition the system runs through each identified text segment (also called a patch), segregating it into lines, words, and eventually singular characters. Each character is then matched against a character image database. Through pattern recognition algorithms, the system concludes the identity of each character, a challenging task due to the diversity in fonts and handwriting styles.

In the post-recognition phase, OCR software undertakes post-processing to enhance accuracy. This involves rectifying typical errors, employing dictionaries for correct spelling, and applying linguistic rules applicable to grammar and syntax. The final stage involves converting recognized text into a preferred format, trying to preserve the original document's layout.

A more advanced approach integrates OCR with sophisticated language models, e.g., BERT (Bidirectional Encoder Representations from Transformers), combining

OCR's text extraction capability with the interpretative power of natural language processing.

The principle however is similar to traditional OCR - an OCR engine transforms text in images into machine-encoded text, but additionally handling diverse fonts and formats. The extracted text is then tokenized, or broken into smaller units for NLP processing. The tokenized text then is fed into a LLM, able to 'understand' word context and overall text meaning. If visual features are available and relevant, they can be integrated into the process. This can include information about the layout of the text, font style, size, location of certain data, and other visual cues. These features can be particularly helpful in understanding the structure and meaning of complex documents containing tables, graphs, or multi-column layouts.

This fusion of OCR and advanced language models allows for precise text extraction and a deeper, contextually aware interpretation and processing of the text.

However, implementing advanced language models in conjunction with OCR demands substantial computational resources and know-how. These requirements present difficulties in settings with constrained processing power, lack of expertise, or in applications that require rapid processing. Another drawback that comes to mind is that traditional OCR-based methods might exhibit limited adaptability when encountering documents of various formats, languages and styles as OCR engines are usually generic, not fine-tuned to the field of application of the subsequent processing.

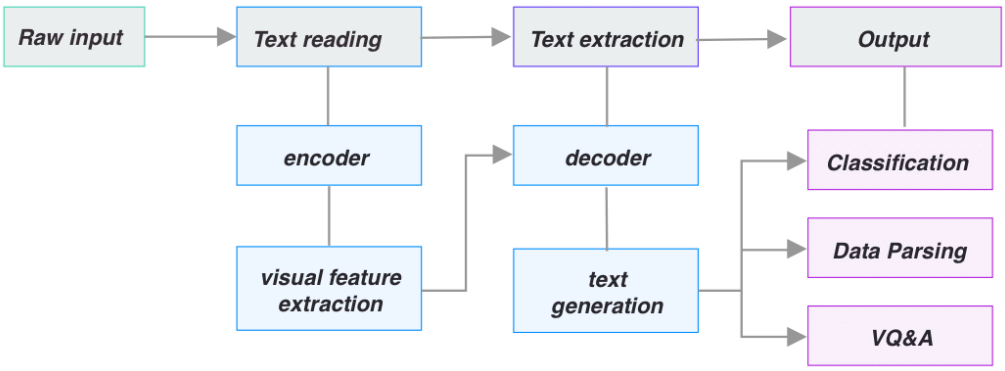

Figure 2 - Illustrated steps in an end-to-end OCR-free approach

In contrast, an OCR-free, end-to-end document recognition approach generally presents a more streamlined operation by eliminating separation of OCR and text processing and generation (Fig. 2). This method can learn directly from raw data to the final output, potentially reducing error propagation, a process by which uncertainties and errors in input data or measurements amplify and affect the accuracy of subsequent computations and results. Additionally, it might demonstrate better

generalization across various document types and styles, as it is trained on the complete task rather than individual parts.

Description of transfer learning approach

The transfer-learning approach in ML involves careful selection of a model that has been trained on a large, comprehensive dataset and its fine-tuning for a specific task that may have a smaller and more specialized dataset. This approach leverages the 'knowledge', i.e., features, weights, etc., that the model has gained during its initial training phase to enhance its performance on a different but related task.
In this project, I aimed to employ the transfer learning approach by fine-tuning Donut, the OCR-free pre-trained model I selected, for the task of pattern recognition and extraction from documents in image.
The Document Understanding Transformer was developed by the research team at Naver CLOVA AI[4] in South Korea approximately one year ago. During my project I explored the efficacy of this model, particularly in its ability to adapt and perform the specialized task of recognizing dates on scanned documents. By leveraging the transfer learning methodology.
I try to ascertain whether the pre-existing knowledge and capabilities of the Donut model can be effectively tailored to address specific requirements of the extraction of dates, thus providing insights into the adaptability of the model in handling specialized pattern recognition challenges.

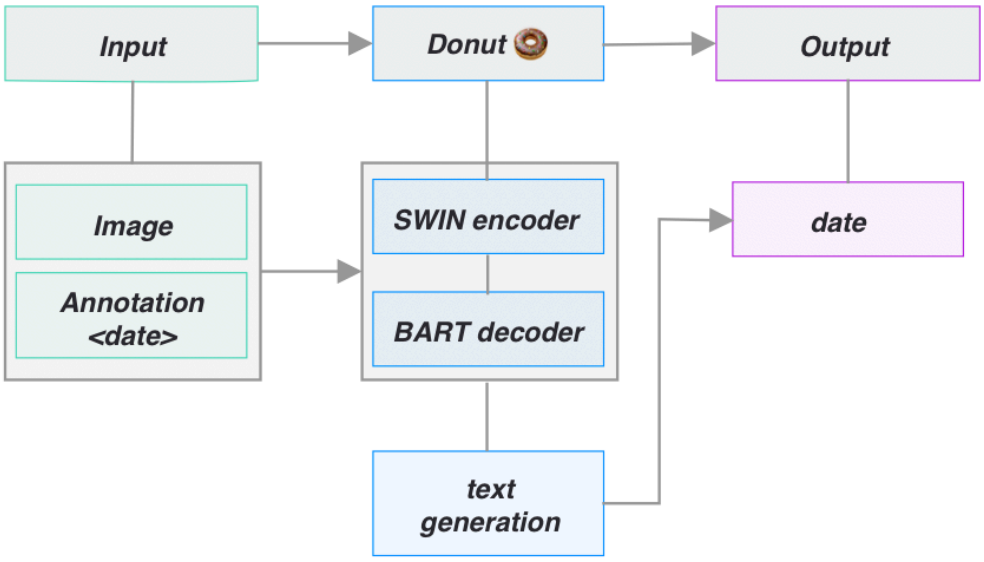

Figure 3 - Architecture of the Document Understanding Transformer (Donut) model

[4] OCR-free Document Understanding Transformer by Geewook Kim et al. https://arxiv.org/abs/2111.15664

Architecture of the Donut

Donut (Fig. 3) employs a visual encoder to extract features from document images, instead of relying on OCR modules. The visual encoder (SWIN) segments the input image into discrete patches which then undergo processing by SWIN Transformer blocks, which include a shifted window-based multi-head self-attention module and a two-layer multilayer perceptron[5]. Patch merging layers further refine the tokens at each stage. The final output from the SWIN Transformer block feeds into the textual decoder.

The textual decoder translates these features into subword tokens, creating a structured format. This model is built entirely on Transformer technology, facilitating straightforward end-to-end training. Using the output, the decoder produces a sequence of tokens, each represented as a one-hot vector. The BART architecture forms the basis of the decoder, initialized with weights from a pre-trained multi-lingual BART model.

Adopting the original Transformer's approach, Donut uses a teacher-forcing scheme during training, where ground truth, rather than previous model outputs, informs each step. The model generates token sequences from a given prompt. Task-specific new special tokens are incorporated into the prompts for various downstream applications. The produced token sequence is transformed into a structured format, with a preference for JSON.

Initial tests

In a pre-phase, I executed inference tests on Donut to evaluate its capability in identifying date patterns, “out of the box”. The Donut model is designed to handle three specific tasks:

1. Classification,
2. Text Extraction, and
3. Document Visual Question Answering (DocVQA),

given the corpora it was fine-tuned on. For the purpose of my study I tested Donut on data parsing and DocVQA, as document classification is not directly relevant.

The results revealed that the Donut model was indeed capable of recognizing dates in the text found on an image. However, the model faced challenges when dealing with dates in various formats and often confused the dates with other numeric patterns. As illustrated in figure 4, the model fails to predict dates correctly in three following attempts: in the first attempt it captures the reference number (“11-14”); in the second it fabricates a date (“frift June 2001”); and, in the third the model returns a time instead of a date (“11:39 a.m.”). But it also undoubtedly recognizes dates, as per the fourth (“january 11, 2005”), and numerous other attempts.

---

[5] OCR-free Document Understanding Transformer https://arxiv.org/pdf/2111.15664.pdf

Recognizing the model's inherent ability to identify dates, I determined that it was unnecessary to modify the architecture of the Donut model, unless I wanted to fine-tune the model to extract multiple dates and their bounding boxes.

First, over-engineered pipeline

As the initial tests with the Donut model proved promising and the HF ecosystem seemed highly adaptable, I embarked on building a pipeline that would prove to be overly complex. Indeed, I thought I could fairly simply extract not just a date but all dates on a document and also identify the matching bounding boxes.

I started with the creation of an explorative, tailor-made dataset specifically containing documents with dates in various formats. The dataset comprised 20 images to represent a diverse range of document layouts and formats, including documents with and without color to test the model's performance under different visual conditions; a variety of layouts, including letter type documents, plain texts, text structures in columns and tables, to challenge the model's adaptability to different structural formats.
Each image in the dataset was accompanied by corresponding annotations, formatted in JSON. The annotations were obtained with help of RectLabel[6], and prepared by careful examination of each image. The process involved:

- Selecting regions in the image that contained dates and possible non-date patterns. This process aimed to test the model's proficiency in differentiating genuine date patterns from similar-looking elements, such as reference numbers (e.g., “38/1908");
- Annotating each identified area with labels indicating whether it was a date or a non-date; and for dates:
- Specifying the text of the date, providing a reference for the expected output of the model;
- Documenting the format of the date to evaluate the model's ability to recognize and process various date formats; and
- Defining the bounding boxes.

My next step then consisted in preparing a HF dataset[7,8] underlying training and testing in the HF ecosystem. The heavy lifting is done in my Python `parse_-annotations_files` function which processes image and annotation files in parallel, and produces the underlying data for the dataset. As I was still experimenting a lot

---

[6] RectLabel is an offline image annotation tool for object detection and segmentation for MacOS (https://rectlabel.com/).

[7] https://huggingface.co/docs/datasets/index

[8] For guidance I used https://www.kaggle.com/code/nbroad/donut-train-benetech

```
 11
'<s_docvqa><s_question> date</s_question><s_answer> 11-14</s_answer></s>'
{'question': 'date', 'answer': '11-14'}
11  Press Enter to continue...

 12
('<s_docvqa><s_question> date formatted as dd-mm-yyyy</s_question><s_answer> '
 'frift june 2001</s_answer></s>')
{'question': 'date formatted as dd-mm-yyyy', 'answer': 'frift june 2001'}
12  Press Enter to continue...

 13
('<s_docvqa><s_question> all dates on image</s_question><s_answer> 11:39 '
 'a.m.</s_answer></s>')
{'question': 'all dates on image', 'answer': '11:39 a.m.'}
13  Press Enter to continue...
../dataset_dir/train/images/img_001.jpeg

 21
('<s_docvqa><s_question> date</s_question><s_answer> january 11, '
 '2005</s_answer></s>')
{'question': 'date', 'answer': 'january 11, 2005'}
```

Figure 4 - Initial, 'out of the box' testing of Donut on date extraction

at this stage with the training data, `parse_annotations_files` allows me to produce different ground truths, tokens and groupings of image and annotations, with or without bounding boxes.

The transformer is then setup with the tokens I introduced for the date extraction task, e.g., `<s_date>` and `</s_date>`[9] to denote the start and end of a date token, and the training and testing dataset is constructed by mapping `transform_and_tokenize` to all data items. In essence, image paths are casts to image objects and converted to their pixel values, and the Donut tokenizer generates ids.

With the data set at hand my next step was to do perform training using PyTorch Lightning[10] considering that the Donut model itself had been trained and tested in that environment. I ran into numerous challenges, possibly due to my lack of experience with PyTorch and PyTorch Lightning, from fundamental but difficult to address compatibility issues with how my dataset and elements thereof at the interface of

---

[9] Other start-end pairs of tokens I experimented with include: `<s_dates>` and `</s_dates>` to fine-tune on multiple dates (`<s_dates> <s_date> date1 </s_date> <s_date> date2 </s_date> </s_dates>`), `<s_bb>` and `</s_bb>` for bounding boxes (`<s_bb> x1, y1, x2, y2 </s_bb>`).

[10] A lightweight wrapper for PyTorch aimed at simplifying machine learning applications. https://pypi.org/project/pytorch-lightning

the HF and PyTorch environments, to my model training indefinitely which is indicative of an issue with the training loop or the data being fed into the model. Every time I tried to fix an issue, I ended up with another one in a different part of the pipeline.

Unwind and restart: Proof of concept pipeline

Given the obstacles I faced with my first pipeline, I made the difficult decision to unwind and restart with the simplest possible pipeline[11], to stay in the HF ecosystem as far as possible, and, with available time, to iteratively improve the pipeline and expand the training and test data sets. Importantly, instead of the custom PyTorch Lightning module, I explored the possibilities of HF sequence to sequence training (`Seq2SeqTrainer`[12]) for my own needs.

Going back to the pre-phase testing, I decided to build a proof of concept pipeline for me to understand the data engineering, fine-tuning, up to prediction and inference results, end-to-end, before embarking on the compilation of a more comprehensive dataset, longer training and testing cycles, and more detailed accuracy testing. To that effect, I reused the explorative, tailor-made dataset of 20 images and own annotations.

I then adapted my Python `parse_annotations_files` function to only handle the date element of my annotations. Similarly, the preparation of the HF dataset had to be streamlined to concentrate on the date element, and the tokens involved in fine-tuning reduced to `<s_date>` and `</s_date>`. All in all, the data preparation and engineering steps remained the same, and ultimately images are converted to their pixel values and the Donut tokenizer generates ids. The processed dataset then consist of three features:

- `labels`: the tokenized input ids,
- `pixel_values`, and
- `target_sequence`: the date to extract.

The next steps consist in:

- Setting up a repository for my own fine-tuned model, e.g., `Anagra/donut-base-ost-sa`. Indeed, HF provides an extensive repository for machine learning models. Each model in the HF Hub has a unique identifier, known as the repository ID. This ID is used to reference and access the specific model one wants to work with. After fine-tuning, the updated model can be saved back on HF Hub, possibly with a new `hf_repository_id` which is specified among arguments for the `Seq2Seq-Trainer`. The repository may be either closed for

[11] As a guidance I used https://www.philschmid.de/fine-tuning-donut

[12] https://huggingface.co/docs/transformers/v4.36.1/en/main_classes/trainer#transformers.Seq2Seq-Trainer

public access (with an assigned private repository tag) or used to share the knowledge with the community (public repository tag).

- Defining the training arguments of the sequence-to-sequence trainer with `Seq2SeqTrainingArguments`,
- Training the model on my dataset, and lastly,
- Saving the fine-tuned model on HF repository and the processed dataset, locally, for subsequent testing purposes.

Building the proper pipeline

With the proof of concept pipeline working, I started searching for more comprehensive datasets and to make better use of the HF ecosystem, in particular for the data engineering part.

I found two useful datasets with both images and annotations, SROIE and XFUND:

- The SROIE dataset is an extensive collection of 1000 whole scanned receipt images and corresponding annotations[13]. A smaller subset of this dataset, comprising 625 images, was obtained for further study. The choice of SROIE was driven by its public domain availability, its relevance and pre-existing annotations.
- XFUND, a multilingual (7 languages) form understanding benchmark dataset, with 150 training and 50 validation images per language, was chosen to bring in more linguistic diversity and additional formats. I focused on European languages, therefore only the German, Spanish, French, and Italian subsets were employed.

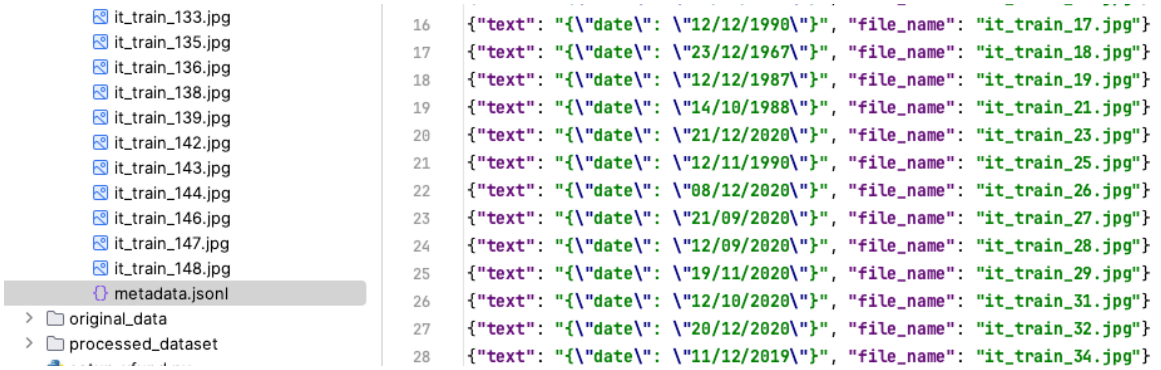

Figure 5 - Meta data for the XFUND dataset

The SROIE and XFUND material does understandably not follow the same organization. I therefore had to develop modules (`setup_sroie.py` and `setup_xfund.py`) to prepare the material in way that lends itself to efficient loading into a HF dataset. In essence the tasks consists in setting up a data folder, possibly split in train, test,

[13] https://paperswithcode.com/dataset/sroie

and validation, with all images together with a single JSONL[14] file (`metadata.jsonl`, Fig. 5) for the annotations and reference to the corresponding image files. With the SROIE data set the preprocessing is relatively straightforward as possible dates are tagged in the JSON annotations. With the XFUND data set the preprocessing is significantly more involved as possible dates are embedded in a multi-level nested JSON annotations, and often treated as filled-in field of forms of different nature and purpose.

The chosen preparation and organization of training and testing material makes the creation of a HF dataset as simple as a single instruction

```
load_dataset('imagefolder', data_dir=base_path),
```

which efficiently and consistently loads image and annotation pairs, and handles image casting from file to image in the same process.
For the dataset to be used in training and testing (fine-tuning), it requires additional processing, along the lines of what I implemented for the proof of concept pipeline.

[14] https://jsonlines.org/

Results

In this chapter, I present a comprehensive overview of the most significant findings from evaluating Donut's performance across various stages of its fine-tuning. I concentrate on those observations that hold relevance and potential applicability for future developments.

The datasets used in the study are the following:

1. HR Private Dataset (HRPD) - a collection of scanned private documents of 20 images in high resolution. Images include documents in various forms, written in English. Each image may or may not contain a date, or contain multiples dates.
2. SROIE - an open source collection of 625 images of receipts
3. XFUND - an open source collection of 706 images of application forms in 4 languages (DE, FR, IT, ES)
4. Random Samples Dataset (RSD) - a collection of 20 images, randomly obtained from Google Image Search. Includes letters, receipts, invoices in different languages.

These datasets were split into Training and Test subsets. I opted against the formation of Validation splits as my datasets are comparatively modest. For evaluation purposes HRPD and RSD were used.

Donut-base + HR Private Dataset (HRPD)

Fine-tuning of Donut-base on HRPD played a role mostly for me to set up the proof of concept HF pipeline, and adapting a dataset to a Donut compatible format. Therefore there were no serious expectations that this fine-tuning would bring some comprehensible results. After the training, the model was not able to find any dates and during inference it was returning a meaningless mix of alphanumeric and special characters and could not recognize any date on the given image.

Donut-base + SROIE

This dataset was split into Training (563 images) and Test (63 images) subsets.
The trained Donut-base + SROIE model showed performance on the Test split of an acceptable level with 74.60% accuracy (executed on 18.12.2023).

The model showed solid understanding of the dates in DD/MM/YYYY format (the format used for the receipts), but not without challenges. Inference on RSD showed calculated accuracy of 22.73% (executed on 18.12.2023).

Firstly, the model encountered difficulties in identifying non-numeric dates (e.g., “March 10, 1987”, Fig. 6) in images of archived documents. This difficulty can be attributed to the training of the model, which has not sufficiently covered archived

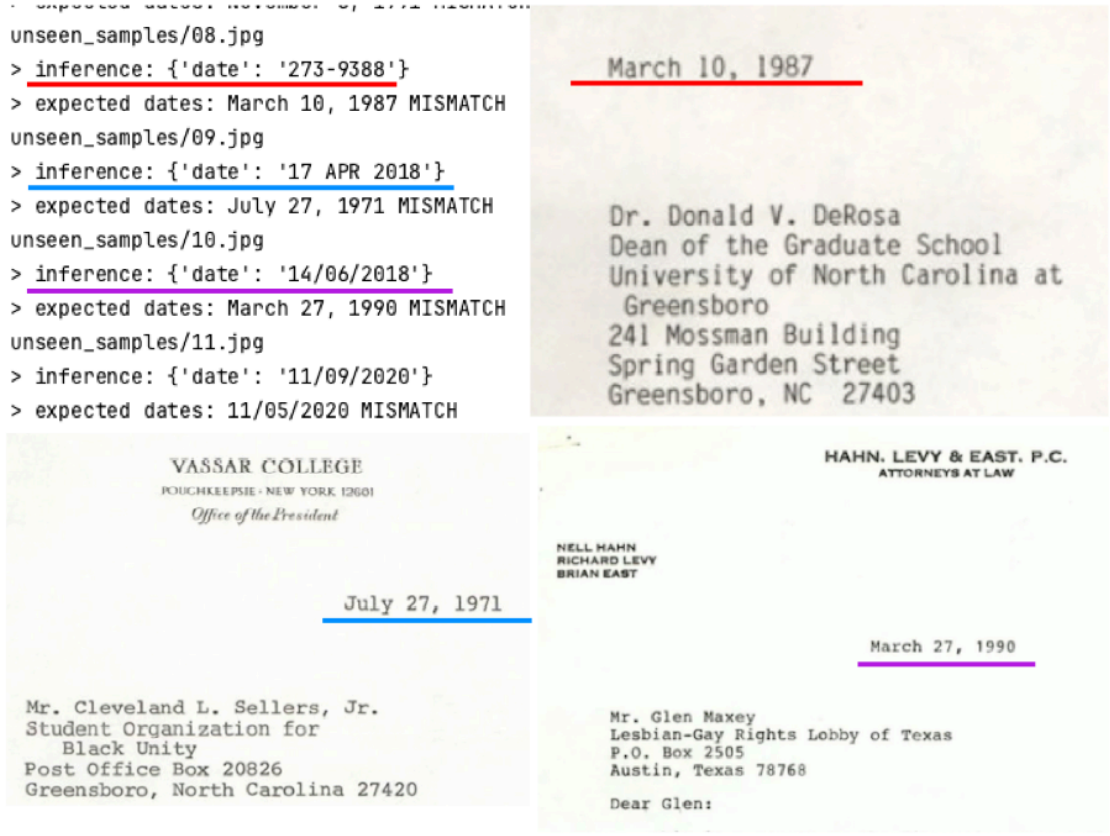

Figure 6 - Examples of improper date extraction involving non-numeric formats

documents characterized by the specific features of such images. I see two factors that contributed to the models low performance on this particular samples:

1. the presence of non-white backgrounds in these documents could have acted as a distracting factor, complicating the date recognition process.
2. the training data's possible bias towards certain fonts, styles, or layouts, typical of modern documents, might have limited the model's exposure to and familiarity with typewriter-style papers often found in archives. This lack of diversity in the training dataset, especially in terms of not including typewriter-specific characteristics, likely hindered the model's ability to accurately recognize and process dates within such contexts.

Secondly, documents in non-Latin languages, such as those using the Cyrillic script (here: Russian, Fig. 7), posed another challenge for the model. Despite being familiar with the date format used on these images, the model struggled to accurately read dates in these contexts. This underlines a critical observation: a model trained predominantly on Latin languages may fail to recognize numerical patterns in images with Cyrillic texts, even if these patterns are similar to those in the training data. This limitation is likely contextual in nature. The model, having learned to identify numerical patterns within the specific context of Latin alphabets, might not effectively

transfer this recognition to contexts where these patterns are surrounded by Cyrillic text. The contrast in linguistic environment can disrupt its ability to accurately isolate and interpret numerical data.

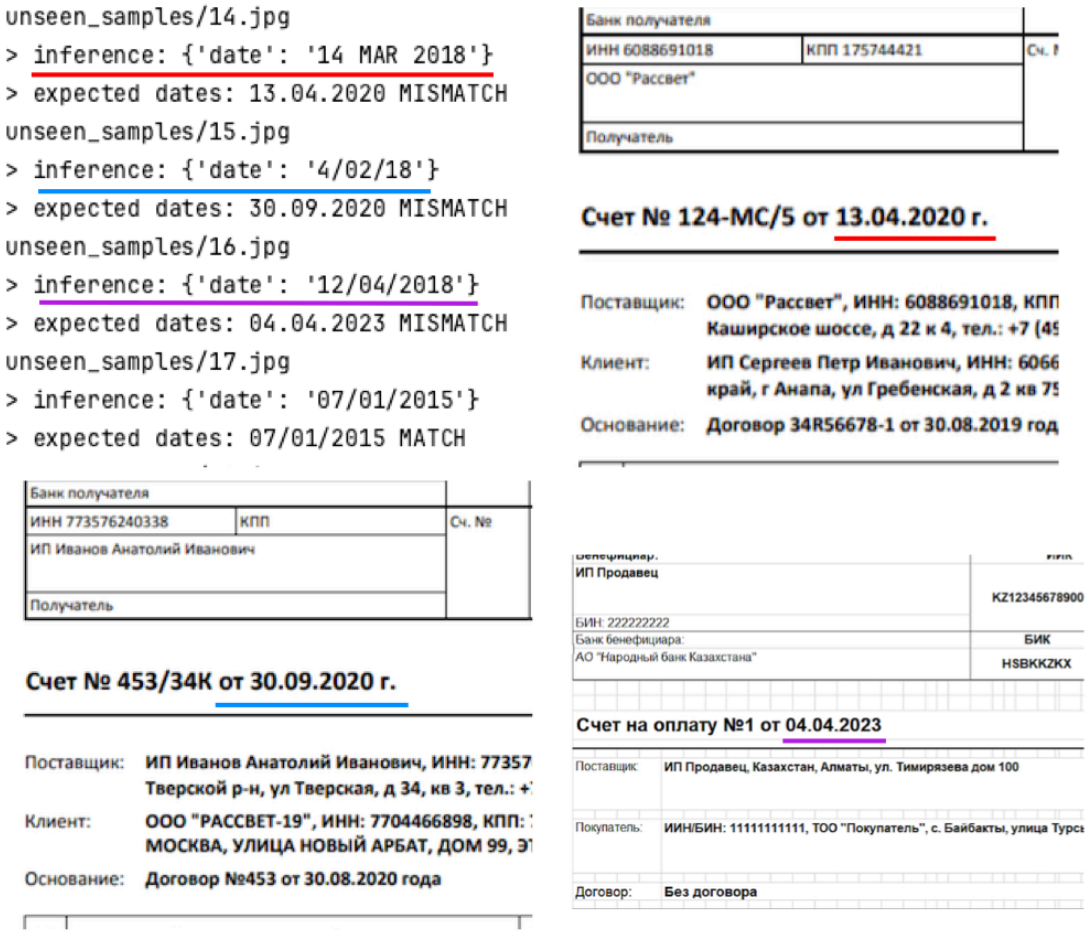

Figure 7 - Examples of improper date extraction involving non-latin scripts

Another interesting aspect I would like to highlight is the model's tendency to generate dates that appear valid but do not correspond to the actual content (I call this phenomenon “hallucinating dates”). This behavior can primarily be attributed to the nature of predictive machine learning models, which are typically conditioned to yield an output for every input, even if the input data is unclear or unfamiliar. These models do not inherently possess the ability to express uncertainty or refrain from making a prediction. Instead, they will attempt to find the closest match based on their training, which can lead to seemingly fabricated outputs. The model's performance is significantly shaped by the nature of its training data and the loss function employed. In scenarios when the model was trained on a dataset where every image had a date and the loss function penalized not predicting a date, the model would learn to always predict a date, even when the input is vague or uncertain.

Donut-base + XFUND

This dataset was split into Training (635 images) and Test (71 images) subsets.

Despite a slight increase in the dataset size, the model underperformed, achieving an accuracy of only 54.93% on the test split, as assessed on 19.12.2023. A likely contributing factor to this outcome is the contextual nature of the training and testing data. The dataset was evenly distributed across four languages, which meant that for each language, the model had only 150 images for training. This quantity is relatively insufficient for robust learning. Additionally, the variation in date formats across these languages should be considered. For instance, while the date August 9, 1997, would be represented as “09.08.1997” in German (DE), it takes the form “09/08/1997” in Italian (IT). In Spanish (ES), the date might appear as either “09/08/1997” or “09-08-1997”. Such differences in date notation can further complicate the model’s learning process, impacting its accuracy.

Donut-base + (SROIE + XFUND)

After merging the SROIE and XFUND datasets, I conducted training on this combined dataset, comprising 1166 images in the training split and 292 images in the test split. The test split yielded a calculated accuracy of 68.84%, which falls short of the performance achieved with Donut-base + SROIE alone.

Interestingly, in some cases the model accurately identified the correct date but represented it in a slightly mixed format (e.g., predicting “06-10.2019” vs actual “06-10-2019”, Fig. 8). In standard accuracy calculations, which have been used to evaluate Donut performance in this project, such predictions would be classified as errors, despite the model correctly identifying the date, significantly impacting model accuracy on unseen data.

Another example: the model correctly predicts the date as "12-07-2021", matching the target date. However, the evaluation flags a MISMATCH due to a subtle difference in the dash characters used. In the target, both dashes are short, whereas in the prediction, the first dash between "12" and "07" is a short dash, and the second between "07" and "2021" is a long, en-dash (Fig. 9).

Both cases illustrate the importance of precision

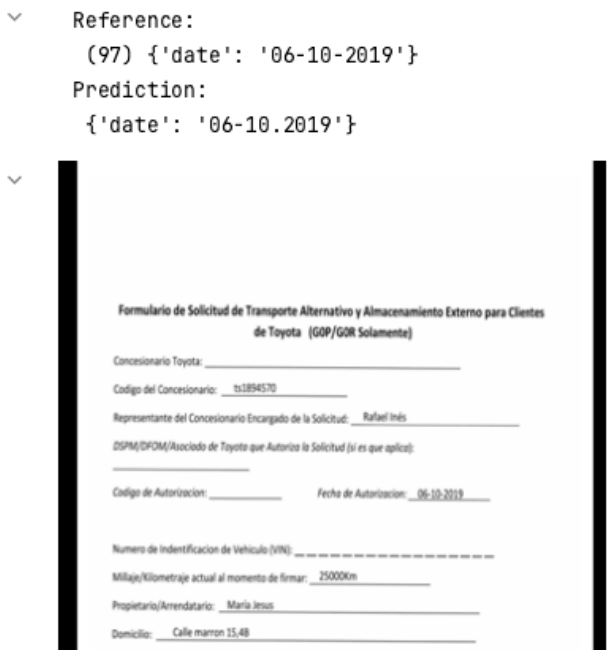

Figure 8 - Challenges with the multiplicity of date separators

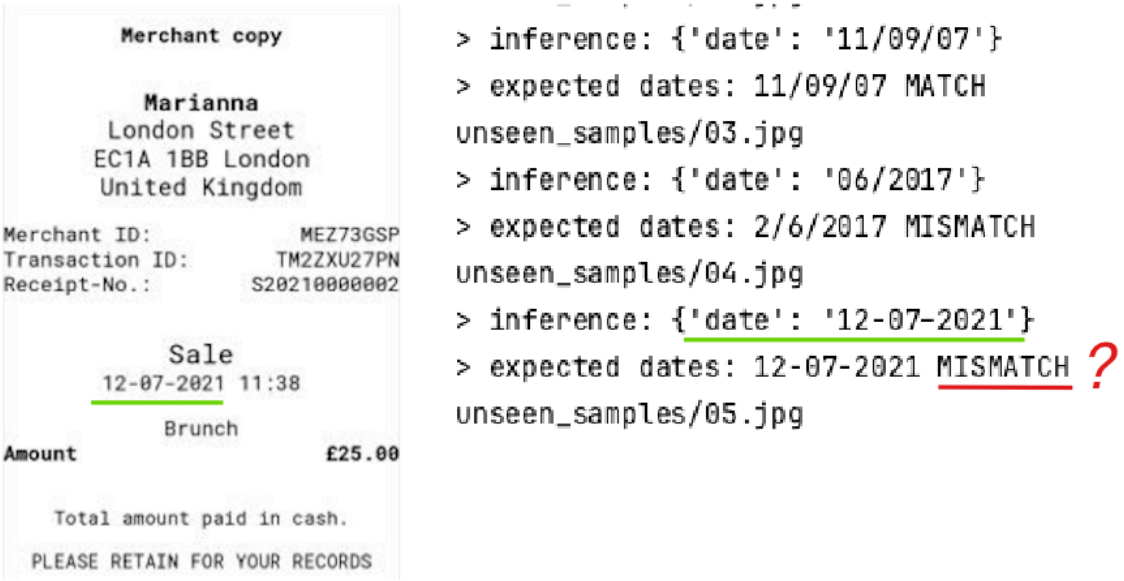

Figure 9 - Challenges with the date separators: dash v. en-dash v. em-dash

in character recognition and the complexities involved in evaluating model performance even in an atomic task like date extraction. While the model successfully captured the essential information (the correct date) in both cases, discrepancies in format and ancillary details can lead, in a strict sense, to an incorrect prediction. This suggests the need for a more refined metric or evaluation approach that can acknowledge and credit partial correctness, especially in applications where exact replication of text format is irrelevant.

Two additional factors which may impair the accuracy of Donut-base + (SROIE + XFUND) come to mind:

1. Varied and Complex Layouts in Application Forms: The combined dataset includes both receipts and application forms (in even parts), with the latter featuring more diverse and intricate layouts. Unlike receipts, application forms often contain text arranged in columns or boxes or both, and their formatting can vary significantly, including lined and unlined styles. This complexity in layout presents a more challenging task for the model to accurately interpret and extract information. The model performed well on less complex forms, but struggled in identifying a date correctly on “busy” layouts (Fig. 10)
2. Multilingual Context of Application Forms: as previously mentioned, the multilingual context poses a challenge for the model, potentially hindering its ability to understand and process the content effectively due to variations in language-specific contexts.

Based on these insights, I hypothesize that enlarging the XFUND dataset to to include approximately 600 images for each language, while retaining consistent lay-

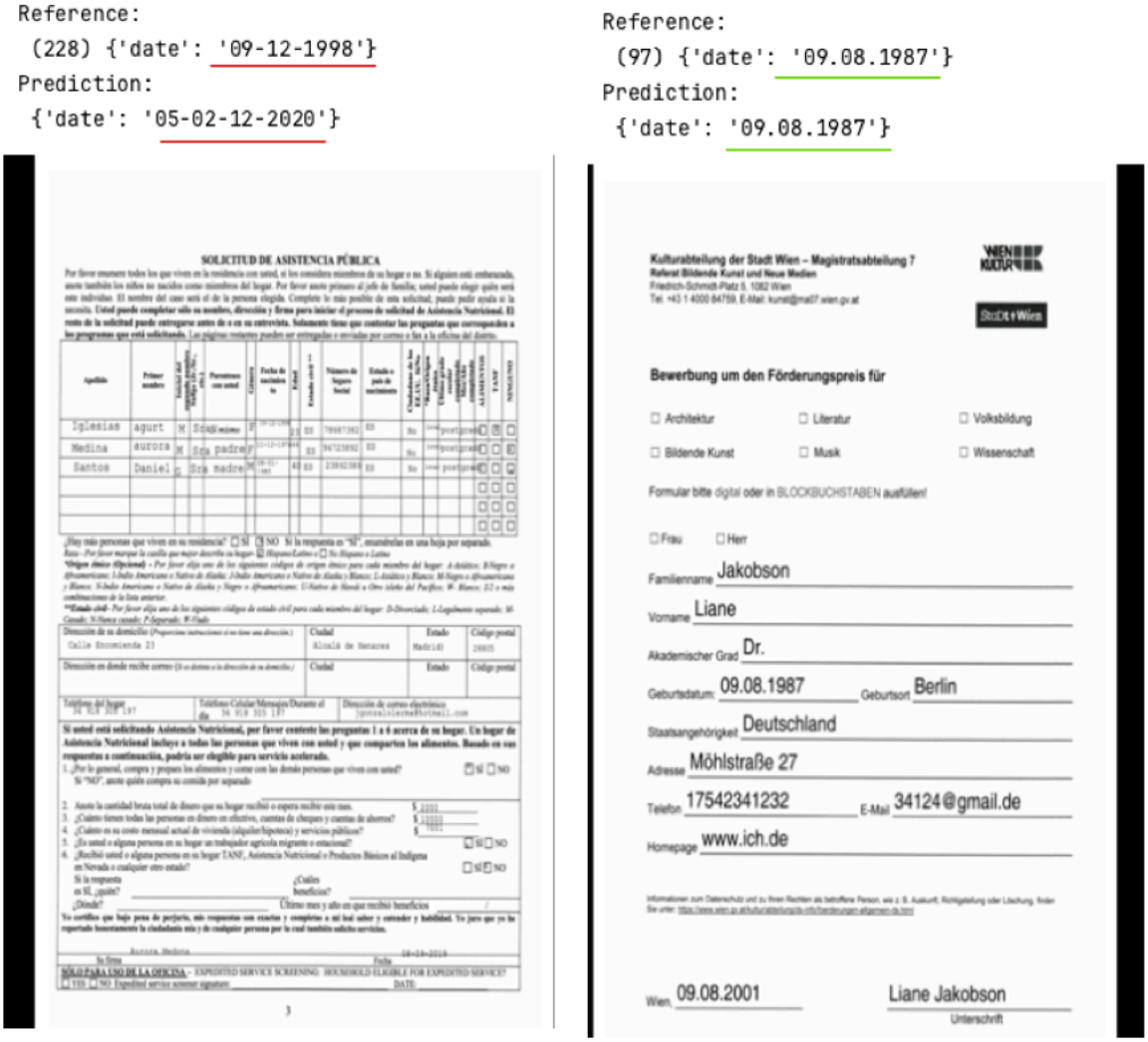

Figure 10 - Challenges with 'busy' forms

out features, particularly with application forms, incorporating typical errors or discrepancies and refining the methods for assessing model performance could boost the model's accuracy on the combined SROIE+XFUND dataset. By focusing on the overall correctness of the predicted date, rather than minor format details or character nuances, the evaluation process will more accurately reflect the model's date recognition capabilities. Expanding the dataset size and diversifying the range of examples per language should enrich the model's training, enhancing its proficiency in navigating complex layouts and multilingual content. This approach aims to provide a more holistic understanding of the model's ability to accurately identify dates.

Conclusion

The exploration of the Donut model's capabilities in date recognition within document images has yielded several insights, central for both the development and application of machine learning models in document parsing and analysis.

The HF library with the large number of pre-trained models it offers is an invaluable resource. These models, while robust in their current form, can be (and should be) fine-tuned to address specific tasks. This adaptability underscores the versatility and potential of pre-trained models in custom applications - from documents classification and digitalization, to image analysis and automated data entry in areas such as finance, healthcare, and legal services.

An essential finding is the influence of dataset size and variety on model performance. The study proved that a small dataset would definitely lead to underperformance, despite the pre-training factor. To enhance its ability to accurately identify and extract dates, regardless of their format, the model requires a larger dataset encompassing a broader spectrum of documents, beyond just receipts and application forms. This diversity is critical for the model to generalize effectively across various document types.
For applications involving documents in languages other than those in the training set, it is imperative to include such language samples in the training process. Given the contextual nature of the model, its ability to recognize patterns is heavily dependent on the context provided during training. Therefore, incorporating a wide range of languages ensures better adaptability and accuracy in diverse linguistic environments.
The complexity of a document's layout has a direct impact on the model's efficiency in recognizing date patterns. Training the model with documents featuring complex layouts is essential to improve its proficiency in these scenarios. This aspect must be carefully considered to enhance the model's overall effectiveness.
Another encouraging observation from this study is the feasibility of fine-tuning the Donut model on large datasets using standard home computing hardware. For instance, training on 700 images with a Mac M1 Max having 32Gb RAM was completed in approximately 1 hour and increasing the dataset to 1200 images resulted in completing the training within just 1.5 hours. This finding opens up possibilities for individual researchers and small teams to conduct significant model training without requiring extensive computational resources.

Finally, the method of validating the model's accuracy should align with the expected output. If the precision of the date format is not as crucial as the general recognition of the date itself, the accuracy metrics should reflect this priority. Tailoring the

validation approach to emphasize overall identification over format accuracy ensures that the model's performance is evaluated in a manner consistent with its intended application.

Recommendations for Future Research

Investigating potential architectural modifications of Donut and understanding model behavior in the context of targeted VDU tasks are crucial areas that warrant detailed exploration. Such modifications are vital for enhancing model performance, especially in ability of the model to locate the extracted pattern on the input image. This can involve two approaches:

- Layer Optimization. Adjusting the number and types of layers within the model can impact how it processes and interprets data. For example, adding convolutional layers and attention mechanisms can enhance a model's ability to recognize bounding boxes and predict the position of dates on images. CNNs excel in spatial feature extraction, while attention mechanisms focus the model on relevant areas, improving accuracy in localizing specific elements like dates.
- Customization for Date Recognition. Tailoring the model specifically for date recognition could involve training it on a dataset with a wide array of date formats, or even tweaking the model to focus on the specific areas of a document where dates are likely to appear. To achieve significant accuracy across various types and forms of documents as well as date formats, the dataset for the proper training must include at least a million of image samples and their annotations.

In conclusion, this study underscores the intricate balance between dataset diversity, model training, and validation methods in the development of effective machine learning models for document analysis. The insights gained provide a roadmap for future enhancements, ensuring that models like Donut can meet the evolving demands of document parsing and pattern recognition.

Glossary

AI. Artificial Intelligence, **“**a branch of computer science dealing with the simulation of intelligent behavior in computers” (https://www.merriam-webster.com/dictionary/artificial intelligence)

BART. Bidirectional and Auto-Regressive Transformers (https://huggingface.co/docs/transformers/model_doc/bart)

CORD. Consolidated Receipt Dataset for Post-OCR Parsing (https://github.com/clovaai/cord)

DL. Deep Learning, “a form of machine learning in which the computer network rapidly teaches itself to understand a concept without human intervention by performing a large number of iterative calculations on an extremely large dataset” (https://www.merriam-webster.com/dictionary/deep learning)

DocVQA. Document Visual Question Answering (https://www.docvqa.org/)

GPT. Generative Pre-trained Transformer (https://en.wikipedia.org/wiki/Generative_pre-trained_transformer)

JSON. JavaScript Object Notation, a lightweight data interchange format

JSONL. JSON Lines text format (https://jsonlines.org/)

LLM. Large Language Model, “a large-scale language model notable for its ability to achieve general-purpose language understanding and generation” (https://en.wikipedia.org/wiki/Large_language_model)

ML. Machine Learning, “a computational method that is a subfield of artificial intelligence and that enables a computer to learn to perform tasks by analyzing a large dataset without being explicitly programmed” (https://www.merriam-webster.com/dictionary/machine learning)

NLP. Natural Language Processing, “an interdisciplinary subfield of computer science and linguistics. It is primarily concerned with giving computers the ability to support and manipulate human language.”
(https://en.wikipedia.org/wiki/Natural_language_processing)

OCR. Optical character recognition, “electronic or mechanical conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document” (https://en.wikipedia.org/wiki/Optical_character_recognition)

RVL-CDIP. A dataset that “consists of scanned document images belonging to 16 classes such as letter, form, email, resume, memo, etc.” (https://paperswithcode.com/dataset/rvl-cdip)

SROIE. An open source dataset of scanned receipts (https://huggingface.co/datasets/darentang/sroie)

SWIN. Pre-trained Shifted Window transformer model (https://huggingface.co/docs/transformers/v4.19.0/model_doc/swin)

VDU. Visual Document Understanding, an area in computer vision and NLP that focuses on interpreting and extraction of visual information from images

XFUND. An open source dataset of scanned forms (https://github.com/doc-analysis/XFUND)

XLMRoBERTa. A Cross-Lingual Language based on Facebook’s RoBERTa (Robustly Optimized BERT pre-training Approach) model released in 2019 ((https://huggingface.co/docs/transformers/model_doc/xlm-roberta, and https://ai.meta.com/blog/roberta-an-optimized-method-for-pretraining-self-supervised-nlp-systems/).