UNIVERSITAT AUTÒNOMA DE BARCELONA

BACHELOR'S THESIS

Multi-Task Document Understanding using Layout+Text to Text Transformer

Author:
Artur Llabrés Brustenga

Supervisor:
Dr. Dimosthenis Karatzas

Final Degree Project
for the bachelor's degree in Computational Mathematics and Data Analysis

Document Analysis Group
Computer Vision Center

September 7, 2022

UNIVERSITAT AUTÒNOMA DE BARCELONA

Abstract

Computational Mathematics and Data Analysis

Multi-Task Document Understanding using Layout+Text to Text Transformer

by Artur Llabrés Brustenga

Document understanding tasks, like visual question answering, information extraction, and document classification, not only have a language component but also a visual one, as the document's layout holds useful information. In this report we take an existing document visual question answering (DocVQA) model, the LayoutT5, and turn it into a multi-task model by framing every task as a question over a document (which is represented by its layout and text). By treating every task in the same way we are able to simultaneously fine-tune the model using multiple datasets on various tasks, and achieve results close to the ones where fine-tuning is done on individual datasets, also we prove that the pretraining of the existing model is useful for all tasks, even those that would not seem related to the original pretraining, like classification.

Contents

Chapter 1

Introduction

1.1 Document Analysis

Today's society uses a large number of documents to function correctly, documents are used for many purposes and exist in many types and forms such as legal contracts, financial contracts, mortgages, diplomas, invoices, scientific papers, etc.

Documents convey information not only through their textual content, but through the way such information is organised, styled, combined with graphical elements, annotations, etc. But not only that, as documents can also contain images, graphs, charts, tables, handwriting, etc. and all of these must be understood together with the text that surrounds them in order to fully understand the document.

Therefore, automatic document understanding and analysis can not be tackled only as a natural language processing (NLP) task (treating the document as a long sequence of words) and neither only as a computer vision task; to fully understand document as a 2-dimensional written message, both language and vision are needed.

Both the natural language processing and computer vision fields have been using deep learning and neural networks since its resurgence in the 2010s, although they have been following slightly different paths.

For computer vision the star architecture has been the Convolutional Neural Network (CNN) which, although it was introduced in 1989 by Yan LeCun (LeCun et al., 1989) for handwritten digit classification, it would not get wide-spread use until training on GPUs allowed the use of larger models and larger datasets, as demonstrated by AlexNet (Krizhevsky, Sutskever, and Hinton, 2012) a CNN architecture which won the ImageNet classification competition in 2012.

Meanwhile the natural language processing field had mostly been using Recurrent Neural Networks (RNNs) and Long-Short Term Memory (LSTM) networks to process text. In 2014 a mechanism of attention was used in a RNN based encoder and decoder for text translation (Bahdanau, Cho, and Bengio, 2014), which allowed the model to focus on the most relevant parts of the text and therefore improve the translation.

In 2017 using self-attention the Transformer architecture was introduced (Vaswani et al., 2017), since then it has become the backbone of almost all natural language processing tasks, usually starting with a large pretrained model on an unsupervised task (Devlin et al., 2019) and then fine-tune on the desired task using transfer-learning (Howard and Ruder, 2018). This has made possible to train larger and

larger models as can be seen in Figure 1.1.

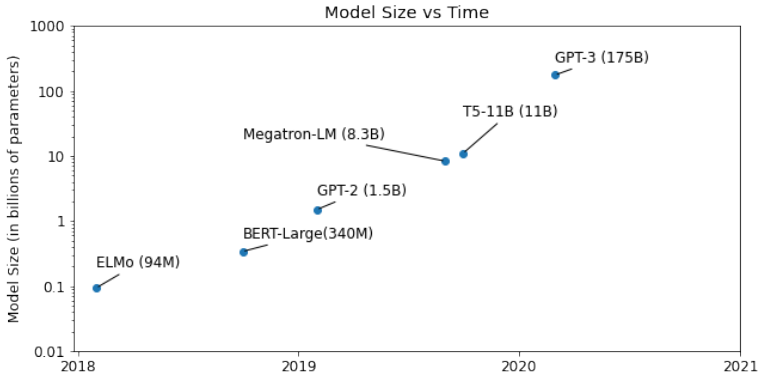


FIGURE 1.1: Number of parameters and date introduced for Transformer-based language models: ELMo (Peters et al., 2018), Bert (Devlin et al., 2019), GPT-2 (Radford et al., 2019), Megatron-LM (Shoeybi et al., 2019), T5 (Raffel et al., 2020), GPT-3 (Brown et al., 2020)

Parallel to the advances in language, Transformers have started to be used in computer vision tasks (Khan et al., 2022), like: image recognition (Dosovitskiy et al., 2020), object detection (Carion et al., 2020; Zhu et al., 2020) and image segmentation (Ye et al., 2019).

As mentioned above, only applying language to document understanding would not work well, as the model would be blind to all the information not encoded in the text. This is why the current approach is to combine the visual information or layout of the document with the text, which can be read with an Optical Character Recognition (OCR) system (Xu et al., 2020; Powalski et al., 2021) or tackle document understanding tasks end-to-end without explicitly performing OCR (Kim et al., 2021).

Document understanding comprises a number of possible tasks, like document classification (Harley, Ufkes, and Derpanis, 2015), information extraction (Graliński et al., 2020), Document Visual Question Answering (DocVQA) (Mathew et al., 2021), document summarization (Zhang, Wei, and Zhou, 2019). However, even if traditionally these have been treated as different tasks, they can be seen in the same way, as the input of the model is a document (for example layout + OCR text) and the output can be casted into text (for example in classification the output would be the name of the class in text). This has already been done for text-to-text with T5 (Raffel et al., 2020) where a single language model can be trained on multiple tasks like text summarization, translation, question-answering. As the input and output for all these tasks is text, then the model is told which is the desired task with a prompt inside the text of the input query, for example "translate: <*text*>" or "summarize: <*text*>".

Possibly something similar can be done for document understanding models where the input will not only be the text but the layout of the document and each task will be defined by the question input to the model, for example: "What type of document

is this? <*layout*>+<*text*>", "Which is the date of this document? <*layout*>+<*text*>".

This would mean that we would be able to use existing document visual question answer (DocVQA) models for other document tasks, which would be cheaper and faster as they leverage the existing pretraining of these models instead of starting from scratch.

1.2 State of the Art

The T5 model (Raffel et al., 2020) is a multi-task text-to-text transformer where the task at hand is indicated to the model using a different text prompt.

As most large language models, T5 starts with an unsupervised pretraining on the Colossal Clean Crawled Corpus, which is constructed from scrapping text from the web. Then it is fine-tuned in a great variety of downstream tasks simultaneously, ensuring that the same number of samples from each task is used.

However, it must be mentioned, that while the fine-tuning was done simultaneously on all task, the evaluations, which in many of the considered tasks surpassed the state-of-the-art, where done by selecting the checkpoint that produced the best results on each task.

The state of the art in document understanding tasks is being achieved by using transformers which take as input the text from the document and also its layout and in some cases visual features from the document.

LayoutLM (Xu et al., 2020) is a transformer model, based on Bert (Devlin et al., 2019) not only using the same architecture (a 12-layer Transformer with 768 hidden sizes, and 12 attention heads, containing a total of about 113M parameters), but also using the same pretrained weights on language.

The layout information of the document is treated by using 2-D positional embeddings which take bounding boxes with the coordinates of each word of the document, and then pass them through four position embedding layers with two embedding tables.

The model can also operate with extra visual information by using image embeddings which consists of the result of cropping the document around each word by using every bounding box and passing this cropped regions of interest through a Faster R-CNN model to produce the embeddings.

While the language weights of the model are already initialized by using the pretrained Bert, the layout and visual embeddings are not, therefore an unsupervised pretraining is done using masked language modeling on 11 million documents where the model must predict the next word, also called token, in the text, with the added difficulty that at random some tokens have been masked, by replacing them with [MASK].

This pretraining is done using 8 NVIDIA Tesla V100 32GB GPUs and it takes 80 hours for one epoch on the 11 million documents for the base model and 170 hours

for the Large model, which has 343 million parameters.

After pretraining the model is fine-tuned on the desired task.

The Tilt model (Powalski et al., 2021), similarly to LayoutLM uses a transformer model, but instead, is based on the T5 model (Raffel et al., 2020).

In this case the layout is provided by concatenating pairwise 1+2D distances to the output of the query and key, as can be seen in Figure 1.2.

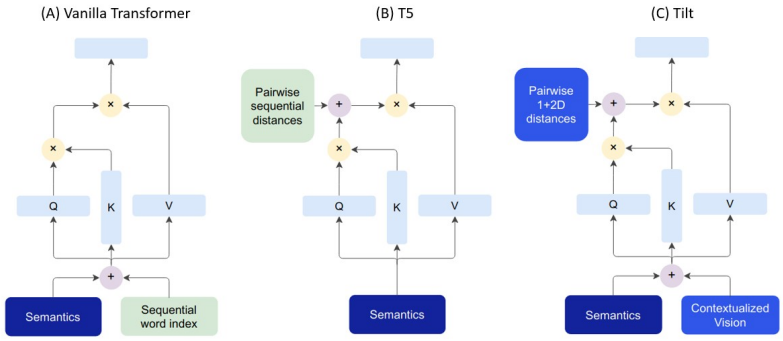


FIGURE 1.2: Comparison between the architecture of a Vanilla Transformer, T5 and Tilt model. Source Powalski et al., 2021.

For TILT visual features are provided by using a truncated U-Net Network instead of the Faster R-CNN used by LayoutLM.

In the same way as in LayoutLM, the language weights are initialized using the T5's pretrained weights, and an unsupervised masked language modeling pretraining is used on a corpus of over 1 million documents.

Afterwards it is pretrained on supervised document tasks by using a collection of multiple datasets like WikiTable, WikiOps, SQuAD, and infographicsVQA. Finally, it can be fine-tuned on the desired task.

Donut (Kim et al., 2021) instead of having the input of the model be the output of an Optical Character Recognition (OCR) program, like LayoutLM, TILT, and us will be doing, uses another approach which is to have an end-to-end model that receives as input an image of the document and a text question, they call this model **Do**cume**n**t **U**nderstanding **T**ransformer (Donut).

In the same way as the previously mentioned models, Donut, uses a general pretraining, which in this case it is based on a large dataset of synthetically generated documents, and then fine-tuned on specific tasks like document classification, document parsing, and document VQA.

Note, that unlike T5, none of the mentioned document understanding models are fine-tuned simultaneously in multiple tasks, after pretraining they are always fine-tuned to specific tasks separately.

1.3 Objectives

As mentioned in Section 1.1 we believe that a good way to train multi-task document understanding model is to follow a similar approach to T5, which consists on taking a general language model and frame every task as text-to-text, in our case we would use a DocVQA model and frame every task as a question over a document which consists of layout and text.

In Section 1.2 we have seen that document understanding models based on large language transformers exist and can be fine-tuned on multiple tasks, but this is always done separately and not simultaneous fine-tuning on multiple tasks at once is done.

Our goal would be to fine-tune an existing document understanding model on multiple tasks and datasets simultaneously in order to obtain a single model that can be used for all tasks.

To do so it will be necessary to adapt existing datasets as visual question answering ones, and employing a DocVQA model to solve the corresponding tasks., see Chapter 2.

In Chapter 3 we will evaluate the performance of the model individually for each dataset (Section 3.1) in order to compare it to the performance obtained when training simultaneously on all datasets (Section 3.2).

Finally in Chapter 4 the conclusions (Section 4.1) drawn form the obtained results will be presented as well a discussion of possible future approaches (Section 4.2).

Chapter 2

Setup: Model, Datasets, and Metrics

2.1 Model: LayoutT5

As mentioned in the previous chapter, the State-of-the-Art models for document understanding are based on Transformer language models like Bert or T5 with additional input that provides the layout and visual features of the document.

In our case we will be using a model called LayoutT5 which is an implementation of the Tilt model developed by Ruben Perez Tito from the Computer Vision Center (CVC).

Two versions of the LayoutT5 model will be used during this study: LayoutT5-small and LayoutT5-base, having as difference the size of the T5 model, T5-small or T5-base, they use as backbone.

The T5 architecture consists of an encoder and decoder comprised of blocks with each block comprising self-attention, optional encoder-decoder attention, and a feed-forward network.

In the case of the T5-small both the encoder and decoder consist of 6 blocks, with 8-headed attention and the feed-forward network having a 2,048-dimensional output, with all other sub-layers and embeddings have a dimensionality of 512. In total T5-small has approximately 60 million parameters.

The T5-base uses 12 blocks for the encoder and decoder, 12-headed attention, an output layer of 3072 dimensions, and all other sub-layers and embeddings have a dimensionality of 768. With approximately 220 million parameters.

Larger T5 models exists, T5-large, T5-3B and T5-11B with 770 million, 2.8 billion, and 11 billion parameters respectively. However they are not used in this study due to the large amount of resources required to train them.

The layout of the document is introduced to the model by concatenating the language token embeddings obtained from the T5 models to the spatial embeddings which are the result of a dropout and a layer normalization on the concatenation of the 2D positional embeddings of the coordinates of the bounding box: left, upper, right, lower, height, width.

Unlike LayoutLM and Tilt we do not input visual features that are the embeddings of a convolutional neural network, therefore the only inputs to the model are the text of the document word by word in the form of OCR tokens, and the layout information in the form of bounding boxes around each OCR token.

The model uses the pretrained weighs from T5 and also has had the following pretraining:

- LayoutT5-small: 10k iterations on a subset of the OCR-IDL dataset with a batch size of 16 and a learning rate of 2e-4. During approximately 8 hours on a GeForce GTX 1080 Ti.
- LayoutT5-base: Pretrained in two steps on two different dataset:
  - 27k iterations on a subset of 432,000k documents from OCR-IDL dataset with a batch size of 16. During aproximately 23 hours using 8 GPUs GeForce GTX 1080 Ti.
  - 24k iterations on DocVQA with a batch size of 8. During aproximately 24 hours using 4 GPUs GeForce GTX 1080 Ti.

Note that from now on when talking about pretraining we will refer to this document specific pretraining, not the language pretraining in T5 weights which we will always use to initialize the model, therefore by saying only fine-tuning we are fine-tuning over the T5 weights without using the document pretraining.

2.2 Datasets

2.2.1 SingleDocVQA

The Single Document Visual Question Answering (SingleDocVQA) (Mathew, Karatzas, and Jawahar, 2020) dataset is a collection of documents from the Industry Documents Library, with document types ranging from letters and emails to notes, memos and reports, containing a mix of printed, typewritten and handwritten content. The answers to the questions are taken verbatim from the documents, however there might be more than one valid answer per question.

We will not be training directly on the SingleDocVQA dataset, instead we will use the test set of this dataset as a benchmark to evaluate the loss of generality of our model after training on other datasets.

2.2.2 SROIE

The Scanned Receipts OCR and Information Extraction (SROIE) Dataset, was introduced at the 2019 Robust Reading Competition.

The dataset consists of 986 scanned receipts from a great variety of businesses, from convenience stores and book stores to restaurants and parking garages. A sample receipt can be seen in Figure 2.1.

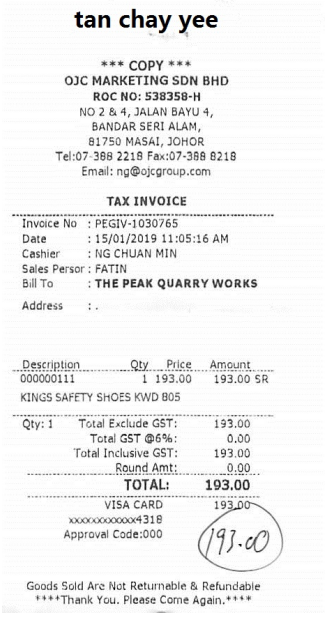


FIGURE 2.1: Sample of a SROIE receipt

For each receipt its OCR text with bounding boxes for each line on text is available in the dataset.

For the receipt shown in Figure 2.1 the beginning of the OCR text is:

```
    98,26,321,26,321,66,98,66,TAN CHAY YEE

138,95,279,95,279,120,138,120,*** COPY ***

80,119,329,119,329,140,80,140,OJC MARKETING SDN BHD

129,142,287,142,287,160,129,160,ROC NO: 538358-H

104,163,306,163,306,182,104,182,NO 2 & 4, JALAN BAYU 4,

123,185,286,185,286,205,123,205,BANDAR SERI ALAM,

116,205,292,205,292,223,116,223,81750 MASAI, JOHOR

69,226,339,226,339,248,69,248,TEL:07-388 2218 FAX:07-388 8218

110,249,300,249,300,272,110,272,EMAIL:NG@OJCGROUP.COM

    [...]
```

The eight numbers at the beginning of each line are the bounding box coordinates for the line, in pairs of two they represent the x and y pixel coordinates of each of the corners of the bounding box. After the coordinates we have the OCR text for the line.

For each receipt in the dataset we also have the company name and its address, the date of the receipt, and the total amount of the receipt. In our current example of Figure 2.1 it would be as follows:


```
"company": "OJC MARKETING SDN BHD",
"date": "15/01/2019",
"address": "NO 2 & 4, JALAN BAYU 4, BANDAR SERI ALAM, B1750 MASAI, JOHOR",
"total": "193.00"
```


Therefore the task we would want our model to tackle is information extraction from the receipt. The input would be the OCR text with the bounding boxes and the name of one of the four fields (company, date, address, total) as query and then the output of the model would be the text or value of that field.

To train and evaluate the model we will use the same split used in the competition, with 626 receipts for training and 360 for testing.

Since we have four possible fields to query the model for each receipt, we will create four samples from each receipt, one for each field.

Therefore in our dataset we will have each receipt four times, with a different query each time. Since the four fields are not always present in every receipt the final proportions will be the following:

- Train:
  - 626 images
    - Company: 626
    - Date: 626
    - Address: 625
    - Total: 626
  - 2503 total samples
- Test:
  - 360 images
    - Company: 346
    - Date: 346
    - Address: 346
    - Total: 346
  - 1384 total samples

2.2.3 RVL-CDIP

The Ryerson Vision Lab Complex Document Information Processing dataset, RVL-CDIP (Harley, Ufkes, and Derpanis, 2015), is a collection of one-page, grayscale, scanned or born-digital documents of different types, from letters and emails to presentations and invoices.

The source of these documents is the Legacy Tobacco Document Library (LTDL) which contains more than 14 million documents with over 80 million pages in total.

This dataset is used to train and evaluate document classification models.

In total it consists of 16 classes or types of documents: letter, form, email, handwritten, advertisement, scientific report, scientific publication, specification, file folder, news article, news article, budget, invoice, presentation, questionnaire, resume, and memo.

A sample of each can be seen in Figure 2.2.

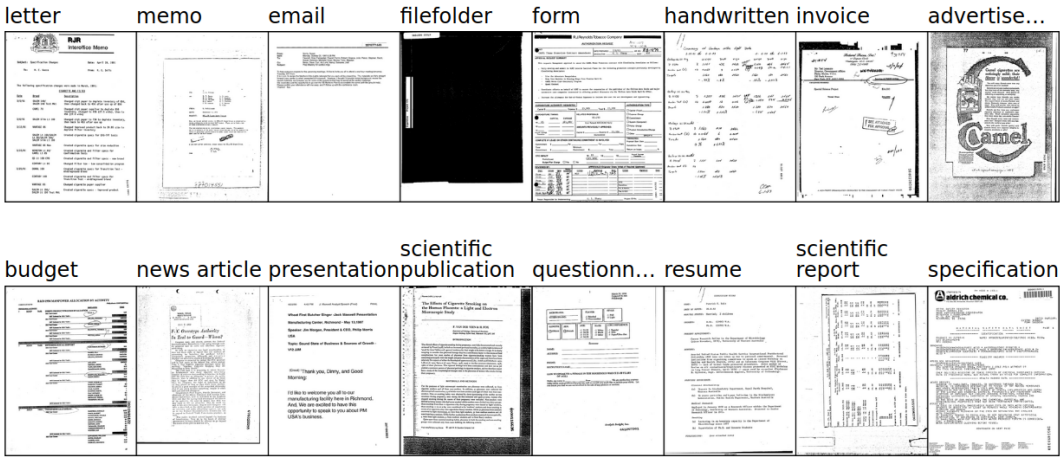


FIGURE 2.2: Sample images of each class in the RVL-CDIP dataset.

The complete dataset contains 400,000 images, 320,000 of them for training, 40,000 for validation, and 40,000 for testing.

In our case however we will not be using the entire dataset, the reason being that the LayoutT5 model we are using requires an OCR of the document with bounding boxes for each word of the text, doing so is expensive and time consuming.

However we had access to an OCR with bounding boxes for a subset of 4.6 million documents from the over 14 million in the Legacy Tobacco Document Library, from which the documents form the RVL-CDIP dataset are also extracted from.

Therefore, using the document metadata we were able to match the documents we have with OCR with the documents in RVL-CDIP.

In total we had about 31% of the documents from the RVL-CDIP:

- Train: 100,352/320,000 (31.36%)
- Validation: 12,563/40,000 (31.41%)
- Test: 12,557/40,000 (31.39%)

Since we have a model for Visual Question Answering (VQA) we will use natural language questions as prompt for the model when using this dataset.

2.2.4 RVL-DocVQA

This Custom DocVQA dataset is a collection of natural language questions regarding information that can be found or extracted from a document. This dataset is composed by the same documents available in the RVL-CDIP dataset. However in this case instead of classifying the documents the task consists of finding the title, the author and the date of the document.

Since it is based on RVL-CDIP, but instead of classification the task is Visual Question Answering, we call this dataset RVL-DocVQA.
Each question type can be ask in two different ways:

Title:

- What is the title of this document?
- What title does this document have?

Author:

- Who is the author of this document?
- Who wrote this document?

Date:

- What is the date of this document?
- What date does this document have?

All questions types are asked for each document if data is available (not all documents contain every field: Title, Author, Date), the version of the question used is chosen randomly between the two.

The answers for the questions are extracted from the document metadata, meaning that they might not appear verbatim in the document.

2.2.5 DocClassVQA

The DocClassVQA dataset is another document classification dataset, that uses natural language questions to ask about the type or class a document belongs to.

To generate this dataset we used the metadata for each document which contains a field with its type.

The dataset has the following size:

- Train: 155,000
- Validation: 15,000
- Test: 15,000

Since the classes are taken directly form the metadata and are not treated in any way there are 116 different classes, where the number of samples in each class differs widely with some classes with thousands of samples and others with less than ten.

Also the proportions between each class are not necessarily maintained the same between the train, validation and test sets, and some classes might not appear in one of the three sets.

2.3 Metrics

The metrics used to evaluate the performance of our models.

2.3.1 Accuracy

Accuracy is simply the ratio between the number of correct answers and the total number of answers, therefore 0 would mean that no answer was correct and 1 that all answers were correct.

An answer will be considered correct if it is exactly the same as the one provided in the dataset.

Accuracy will be displayed as a percentage, and will be therefore between 0 and 100.

2.3.2 Average Normalized Levenshtein Similarity (ANLS)

The ANLS metric (Biten et al., 2019), described in Equation 2.1, mesures the distance between the ground truth answers and the models answers using the Normalized Levenshtein distance and then takes the average over the batch size, it is less strict than accuracy which would require an exact match.

When the Normalized Levenshtein distance, which is a value between 0 and 1, is greater or equal to 0.5 it is considered to large and set to 0.

$$ANLS = \frac{1}{N}\sum_{i=0}^{N} s(a_i, o_{q_i}) \tag{2.1}$$

$$s(a_i, o_{q_i}) = \begin{cases} (1 - NL(a_i, o_{q_i})) \text{ if } NL(a_i, o_{q_i}) < 0.5 \\ 0 \text{ if } NL(a_i, o_{q_i}) \geq 0.5 \end{cases}$$

where a_i is the ground truth answer to the i^{th} question and o_{q_i} is the model's answer.

With $NL(a_i, o_{q_i})$ being the Normalized Levenshtein distance between a_i and o_{q_i}.

ANLS will be displayed as a percentage, and will be therefore between 0 and 100.

Chapter 3

Experiments

3.1 Single-Dataset Experiments

3.1.1 SingleDocVQA

The SingleDocVQA dataset has only been used as a benchmark and therefore no specific training or fine-tuning has been done on this dataset, therefore all results shown in this subsection are using a pretrained LayoutT5 as described in Chapter 2, Section 2.1.

In Table 3.1 the results on the SingleDocVQA test set are shown for both the LayoutT5-small and LayoutT5-base models and when using 300 and 1024 OCR tokens.

As expected the base model performs better than the small model, and more OCR tokens give better results.

TABLE 3.1: Accuracy, and inside parenthesis, ANLS, on the SingleDocVQA test set using the LayoutT5-small and LayoutT5-base models, when using 300 and 1024 OCR tokens respectively. Both metrics as a percentage (%).

Model	300 OCR tokens	1024 OCR tokens
LayoutT5-small	47.88 (55.85)	52.79 (60.72)
LayoutT5-base	54.21 (62.46)	60.06 (68.72)

3.1.2 SROIE

As mentioned in Chapter 2, Section 2.2.2 the OCR for this dataset has bounding boxes for each line of text instead of for each word.

Since the rest of datasets, including the ones used for pretraining, had bounding boxes per word, the first experiments performed where to analyze the effect of this different OCR.

Results using a LayoutT5-small model with 300 OCR tokens and fine-tuned on 5000 iterations with batch size of 30.

In Table 3.2, each row is an experiment using a different type of bounding boxes, the first one uses the bounding boxes directly from the dataset with a box for each line of text, the second one does not use bounding boxes it sets them to 0, the third

creates boxes for each word by dividing the length of the box for the line between each word using the number of letters per word as an estimation of its length, finally the last row uses the OCR and bounding boxes from Amazon's Textract API, which is the same OCR used for the other datasets and has bounding boxes for each word.

Note that the Textract OCR might make some mistakes when reading the documents and then the text we are feeding to the model might not contain the answer in the same way it is written in the ground truth, for example we saw it read a date as 28/04/2017 when it was 26/04/2017 (although frankly looking at the image I think more than one human would have made the same mistake), another difference we saw a lot between Textract and the original OCR is to not register some small spaces as a space and therefore joining to words together.

In the test set for SROIE we have found that the answer provided as ground truth and the text form Textract differ in 219 samples of the total 1384, therefore the maximum accuracy we could expect using this OCR is 84.17%.

The columns of Table 3.2 show the accuracy, and inside parenthesis, ANLS, for when only pretraining is used, only fine-tuning, and pretraining & fine-tuning.

TABLE 3.2: Accuracy, and inside parenthesis, ANLS, on the SROIE test set using the LayoutT5-small, with only pretraining, only fine-tuning, and pretraing & fine-tuning. Both metrics as a percentage (%).

Model	Only Pretraining	Only Fine-Tuning	Pretraing & Fine-Tuning
Original BBox	40.04 (53.87)	67.94 (81.10)	69.96 (82.50)
No BBox	40.04 (53.18)	68.30 (81.41)	70.11 (82.47)
Word Bbox	28.23 (53.10)	59.40 (79.69)	67.77 (83.02)
Textract OCR	38.51 (54.08)	64.22 (81.01)	66.63 (82.50)

In Table 3.2 we see that using the original bounding boxes or setting them to 0 does not seem to change the results that much, we see less than 1% improvement in accuracy and ANLS stays almost the same.

Trying to create bounding boxes for each word from the original OCR does not work and produces worst results, which is not entirely surprising as inside one line there could be large gaps between words, and by just dividing the total length of the line between the words this are not taken into account, therefore this boxes confuse the model rather than help it.

The results for Textract OCR are also not very surprising, accuracy is considerably worse, but ANLS is similar to the original OCR, this is most likely due to the mistakes in the text as has been mentioned above. Despite this, the rest of the experiments use the the Textract OCR.

The second round of experiments consists on comparing the LayoutT5-small model with the LayoutT5-base model in the same cases of only pretraining, only fine-tuning, and pretraining & fine-tuning.

The fine-tune for the base model use 1024 OCR tokens, a batch size of 2 and 10000 iterations.

As we would have expected, in the results shown on Table 3.3 we see that the base model achieves better results than the small model, despite using a much smaller batch size.

TABLE 3.3: Accuracy, and inside parenthesis, ANLS, on the SROIE test set using the LayoutT5-small and LayoutT5-base models, with only pretraining, only fine-tuning, and pretraing & fine-tuning. Both metrics as a percentage (%).

Model	Only Pretraining	Only Fine-Tuning	Pretraing & Fine-Tuning
LayoutT5-small	38.51 (54.08)	64.22 (81.01)	66.63 (82.50)
LayoutT5-base	45.85 (57.90)	70.23 (85.44)	72.69 (86.33)

3.1.3 RVL-CDIP

The experiments conducted on the RVL-CDIP dataset are the following:

- Only DocVQA Pretraining: Inference on the RVL-CDIP test set using the pretrained weights as described in Chapter 2, Section 2.1.
- Only Fine-tuning: Inference on the RVL-CDIP test set without DocVQA pretraining. Note that it does have the pretraining on language of the T5 model, but not the weights of the layout embedding layers.
- DocVQA Pretraining and Fine-tuning: Inference on the RVL-CDIP test set using the pretrained weights as described in Chapter 2, Section 2.1 and then fine-tuned on the RVL-CDIP training set.

For LayoutT5-small the fine-tuning consisted of 10000 iterations on the training set with a batch size of 50 and using 300 OCR tokens, and for LayoutT5-base the same number of iterations where used, but since it is a larger model and more OCR tokens are being used, the batch size was reduced to 2 and using 1024 OCR tokens.

This means that the small model will have trained on many more samples than the base model, as the total number of samples seen during training is *batch_size* times *n_iterations*.

Since the RVL-CDIP training set we have consists of 100352 samples it means that the small model will go through the entire training set close to five times, while the base model will not have seen every sample in the training set.

In Table 3.4 the results of the experiments mentioned above can be seen, as expected pretraining and fine-tuning gives the best results.

Since none of the pretraining included document classification, the model with only pretraining has very low accuracy and ANLS, the reason seams to be that the model expects to find the answer in the document and therefore answers with some text

from the document, like the title.

While only pretraining is not useful at all, it clearly does help when it is used before fine-tuning, as accuracy improves about 8 percent when comparing only fine-tuning with pretraining & fine-tuning.

TABLE 3.4: Accuracy, and inside parenthesis, ANLS, on the RVL-CDIP test set using the LayoutT5-small and LayoutT5-base models, with only pretraining, only fine-tuning, and pretraing & fine-tuning. Both metrics as a percentage (%).

Model	Only Pretraining	Only Fine-Tuning	Pretraing & Fine-Tuning
LayoutT5-small	1.15 (2.71)	82.87 (83.38)	91.48 (91.68)
LayoutT5-base	0.69 (1.40)	81.25 (81.70)	89.28 (89.62)

3.1.4 RVL-DocVQA

Same experiments as in RVL-CDIP, but only using the LayoutT5-small with a batch size of 5.

As expected the best results are obtained with pretraining and fine-tuning, as can be seen in Table 3.5.

The curious thing is that ANLS is always much higher than accuracy, this is not entirely surprising as has been mentioned in Chapter 2, Section 2.3 ANLS is less strict than accuracy as it does not require an exact match.

However, in this case ANLS is much larger than accuracy, which might be a result of the way the dataset was generated, the answers used in the dataset come from the metadata of the documents, and therefore the answer might not be written in the exact same way as in the document, for example the date provided as answer is always in the same format, while in the document other formats might be used.

ANLS being an edit distance is not penalizing the model as much for these errors that are in part the dataset's fault not the model's.

TABLE 3.5: Accuracy, and inside parenthesis, ANLS, on the RVL-DocVQA test set using the LayoutT5-small model, with only pretraining, only fine-tuning, and pretraing & fine-tuning. Both metrics as a percentage (%).

Model	Only Pretraining	Only Fine-Tuning	Pretraing & Fine-Tuning
LayoutT5-small	1.58 (8.38)	1.97 (33.77)	9.02 (51.09)

Note that this dataset will not be used in any of the other experiments.

3.1.5 DocClassVQA

Same experiments as in RVL-CDIP, using both LayoutT5-small and LayoutT5-base models, with the same batch size (50 and 2), and same number of iterations.

In the results shown in Table 3.6 we see a similar behaviour as in the RVL-CDIP results (Table 3.4), however both metrics are much lower in this case.

The reason for the worst performance on this dataset is clear, the RVL-CDIP dataset is much easier than the DocClassVQA dataset, the former only has 16 classes with a similar number of samples for each class, while the latter has 116 classes which are not balanced at all with some classes having thousands of samples and others less than ten.

Interestingly, we see that the LayoutT5-small model is obtaining a better performance than the base model, this is due to the larger batch size, 50 instead of 2.

TABLE 3.6: Accuracy, and inside parenthesis, ANLS, on the DocClassVQA test set using the LayoutT5-small and LayoutT5-base models, with only pretraining, only fine-tuning, and pretraing & fine-tuning. Both metrics as a percentage (%).

Model	Only Pretraining	Only Fine-Tuning	Pretraing & Fine-Tuning
LayoutT5-small	0.82 (0.79)	36.94 (37.50)	51.25 (52.02)
LayoutT5-base	0.57 (0.96)	35.80 (36.4)	45.83 (46.37)

3.2 Multi-Dataset Experiments

Until now we have seen results for a single dataset, but as stated in Chapter 1 our goal is to have a general model that can work on multiple tasks and datasets.

3.2.1 Fine-Tuning One After the Other

As its name suggests, this experiment consists on fine-tuning a model on each dataset one after the other, and then testing it on the test set of all datasets between each step.

The model used is the LayoutT5-small, using 300 OCR tokens and a batch size of 50 with 5000 iterations per training dataset. Results are shown in Figure 3.1 and in table form on Table 3.7.

In these results we see a clear case of catastrophic forgetting, which occurs when a model is fine-tuned on a new dataset and forgets the other datasets it has been previously fine-tuined on, meaning that the performance of the model on the previous dataset is close to 0.

In this case it happens between the information extraction datasets and the document classification datasets, as we see that both SROIE and SingleDocVQA have 0 or almost 0 accuracy after fine-tuning on RVL-CDIP and DocClassVQA.

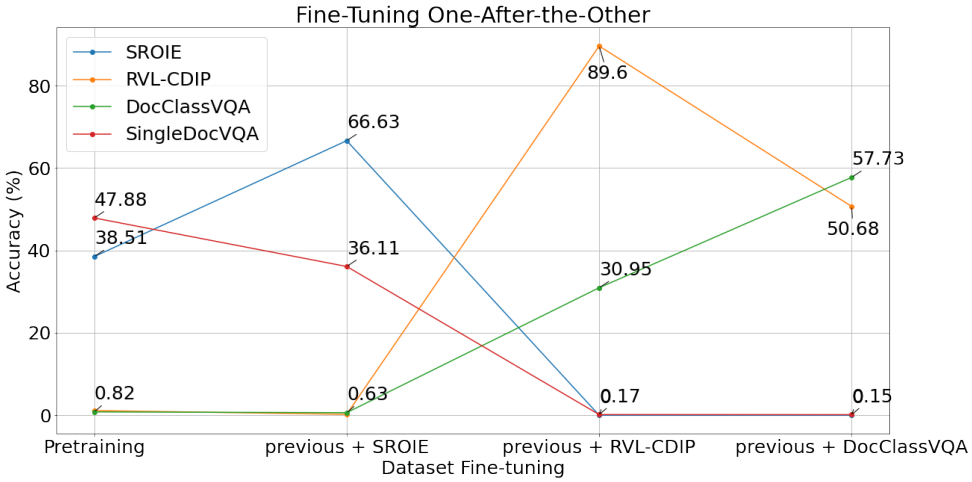


FIGURE 3.1: Results for the fine-tuning one after the other experiment. Each line is the accuracy of the model on the test set of the corresponding dataset. The x-axis notes which dataset has been used for fine-tuning at each step, the order used for this experiment was first pretraining, and then fine-tuning on SROIE, RVL-CDIP, and DocClassVQA.

TABLE 3.7: Accuracy, and inside parenthesis, ANLS, on each dataset (rows) after fine-tuning a LayoutT5-small model in the each dataset in order (columns): Pretraining, SROIE, RVL-CDIP, DocClassVQA. Both metrics as a percentage (%).

Fine-Tuning	Pretraining	SROIE	RVL-CDIP	DocClassVQA
SROIE	38.51 (54.08)	66.63 (82.50)	0 (0.04)	0 (0.05)
RVL-CDIP	1.15 (2.71)	0.23 (0.91)	89.6 (89.87)	50.68 (56.21)
DocClassVQA	0.82 (0.79)	0.63 (0.46)	30.95 (33.78)	57.73 (58.34)
SingleDocVQA	47.88 (55.85)	36.11 (44.45)	0.17 (0.24)	0.15 (0.19)

From these results it is also interesting to note the synergies between RVL-CDIP and DocClassVQA, while DocCalssVQA has many more classes than RVL-CDIP (116 vs 16) the former includes the classes of the latter, in fact the test set of DocClassVQA includes 5857 documents which belong to classes in RVL-CDIP, which represent a 36.83% of the test set, explaining the 30.95% accuracy of DocClassVQA after fine-tuning on RVL-CDIP.

3.2.2 Fine-tuning Simultaneously on all Datasets

The other way to fine-tune on multiple datasets is to fine-tune on all datasets at the same time, this way can mitigate the risk of catastrophic forgetting which we saw in the previous experiment (3.2.1).

Training simultaneously means that each sample will be chosen randomly from one of the datasets, however since each dataset has a different number of samples choosing randomly from the pool of all samples would mean that the model would be

training more on one dataset than the others.

This is why we will use a parameter that specifies the percentage of samples we want of each dataset in the training set of all samples.

By trying different combinations of this percentage we can find the right proportions of each dataset in order to get the best results on all datasets.

For this kind of experiments only LayoutT5-base will be used, with 1024 OCR tokens, a batch size of 2 and 10000 fine-tune iterations on top of the pretraining.

The results of these experiments can be found in Table 3.8.

The first column of Table 3.8 shows the percentage of samples of each dataset in the training set, using the same order as the following four columns, SROIE, RVL-CDIP, DocClassVQA, and SingleDocVQA.

Note that the percentage for SingleDocVQA is always 0 as it is only being used for testing and not training.

Columns 2 to 4 show the accuracy, and inside parenthesis, ANLS, of the model on the test set of each dataset.

The last column are the results when using a test set composed by samples of all datasets in the same proportions as noted by the percentages in the first column.

The table is divided in five sections:

The first one, where all percentages are 0, has results for when only pretraining is being used.

The second one, has results for when the percentage of one dataset is 100%, meaning that only that dataset is being used.

The third one, where all percentages are 33 (except for SingleDocVQA which is 0 as has been mentioned before), has results for when the proportion of each dataset is the same.

The last two sections have results for setting one dataset at 60 or 40 and keeping the other two at 20 or 30 respectively.

When only using pretraining (first row with all percentages at 0) we see the same results which have been already discussed individually for each dataset, SROIE and SingleDocVQA since they are similar to the pretraining get accuracies of 45% and 60% while the two classification datasets, RVL-CDIP and DocClassVQA, get accuracies of less than 1%.

When keeping all percentage the same, in row 5, we obtain an accuracy on SROIE close to the one obtained when only fine-tuning on SROIE.

TABLE 3.8: Results for Fine-tuning on all datasets simultaneously. The percentage of each dataset used is in the first columns and the accuracy and ANLS on the test set of each dataset is the rest, with the final one being the results on a test dataset combining the test datasets in the same proportions as the first columns. Both metrics as a percentage (%). When fine-tuning on a single dataset, the results on that dataset are marked in bold, and when fine-tuning using percentages, the best results on each dataset is also marked in bold.

Percentage of each dataset	SROIE	RVL-CDIP	DocClassVQA	SingleDocVQA	Combine
0 0 0 0	45.88 (57.90)	0.69 (0.14)	0.57 (0.96)	60.06 (68.72)	-
100 0 0 0	**72.69 (86.33)**	0.36 (1.05)	0.21 (0.46)	49.44 (58.27)	-
0 100 0 0	0 (0)	**89.38 (89.62)**	30.79 (33.69)	0.07 (0.12)	-
0 0 100 0	0 (0)	40.38 (46.69)	**45.72 (46.25)**	0.18 (0.25)	-
33 33 33 0	71.97 (86.41)	82.13 (82.44)	41.43 (43.77)	15.71 (17.91)	65.25 (71.10)
60 20 20 0	72.04 (86.25)	81.01 (81.29)	37.25 (39.87)	21.70 (25.34)	67.10 (76.03)
20 60 20 0	71.10 (85.84)	**86.89 (87.21)**	34.53 (37.42)	9.83 (11.14)	73.63 (77.55)
20 20 60 0	71.32 (85.72)	65.66 (73.74)	**42.53 (44.67)**	9.98 (11.31)	53.70 (59.23)
40 30 30 0	**72.25 (86.38)**	80.48 (80.74)	40.03 (42.71)	16.05 (18.78)	64.94 (71.66)
30 40 30 0	71.89 (86.18)	83.25 (83.54)	41.52 (44.05)	**22.33 (26.23)**	67.09 (72.01)
30 30 40 0	71.60 (86.05)	78.28 (78.72)	42.35 (44.71)	17.20 (20.32)	61.74 (67.05)

As has been mentioned in the previous section (3.2.1) there are synergies between the RVL-CDIP and DocClassVQA datasets since both are document classification datasets and share some of the classes, but up to a point, as the documents from both datasets are very similar and the questions asked are the same, so the model does not have a way of telling from which dataset a document is coming from, therefore training to much on one dataset will result in the model just answering as if the only classes are the ones for that dataset. This is the reason why we see a drop in accuracy for DocClassVQA when the percentage for RVL-CDIP is 60% instead of 20%.

Similarly, we see a big drop on RVL-CDIP when the percentage for DocClassVQA is 60% instead of 20%.

Also of note is the fact that when using 30% SROIE, 40% RVL-CDIP, and 30% DocClassVQA, the accuracy from SingleDocVQA is the highest among all the ones a combination of datasets is used, which is interesting since the accuracy of SingleDocVQA is 49.44% when only fine-tuning on SROIE and close to 0% when fine-tuning on RVL-CDIP or DocClassVQA, and despite this, SingleDocVQA is obtaining here a better accuracy, albeit marginally, than in combinations where more SROIE and less RVL-CDIP and DocClassVQA are used, like 60%, 20%, 20% or 33%, 33%, 33%.

Chapter 4

Final Thoughts

4.1 Conclusion

In the course of this report we have fine-tuned layout+text visual question answer transformers using multiple datasets in two different document analysis tasks, information extraction, and document classification, by framing both tasks as a natural language question answering task.

Not only we have seen that the layout+text transformer we used, the LayoutT5 model, is capable of obtaining good results in these two tasks separately, but also that by fine-tuning simultaneously on both tasks by combining the datasets we had available in the right proportions, we could achieve results that where quite close to the performance of fine-tuning on each dataset separately, as can be seen in Table 4.1.

TABLE 4.1: Accuracy, and inside parenthesis, ANLS. In the first row results for LayoutT5-base fine-tuned on each dataset separately (results form Section 3.1), in the second row results from fine-tuning simultaneously on every dataset by combining them in the following way: 30% SROIE, 40% RVL-CDIP and 30% DocClassVQA, which is the combination that achieves the highest results on each dataset (results from Section 3.2.2). Both metrics as a percentage (%).

	SROIE	RVL-CDIP	DocClassVQA	SingleDocVQA
Separate Fine-Tuning	72.69 (86.33)	89.38 (89.62)	45.72 (46.25)	60.06 (68.72)
Simultaneous Fine-Tuning	**71.89 (86.18)**	**83.25 (83.54)**	**41.52 (44.05)**	**22.33 (26.23)**

In Table 4.2, results from the methods mentioned in Section 1.2 State of the Art (SOTA), LayoutLM, TILT, and Donut, for the SROIE and RVL-CDIP datasets.

Note that a direct comparison between the State of the Art results and our results in not possible for multiple reasons.

In RVL-CDIP we are using a subset of the entire dataset, as described in Section 2.2.3, and in SROIE we are using a custom OCR and evaluating each question (company, date, total, address) separately, while in the SOTA results the F1 score displayed in Table 4.2 is computed considering the performance on each image, considering all questions simultaneously.

It must also be noted that in all of the SOTA cases the fine-tuning is task specific, meaning that it is done separately for each dataset, and therefore two models with different weights would be obtained instead of a single multi-task model which was

TABLE 4.2: State of the Art results on the SROIE and RVL-CDIP datasets.

Model (parameters)	SROIE (F1)	RVL-CDIP (Accuracy)
LayoutLM-Base (160M)	94.38	94.42
LayoutLM-Large (343M)	95.24	94.43
Tilt-Base (230M)	97.65	95.25
Tilt-Large (780M)	98.10	95.52
Donut (156M)	-	94.50

our goal.

We have also seen the effects that pretraining the model before fine-tuning has on its performance, and found, that even in the case of document classification, where the pretraining is not directly related to the task, the performance of fine-tuning after pretraining was significantly better than without pretraining, as can be seen in Table 4.3.

TABLE 4.3: Accuracy, and inside parenthesis, ANLS, when fine-tuning without pretraining, and pretraining & fine-tuning on the LayoutT5-base model. Both metrics as a percentage (%).

	SROIE	RVL-CDIP	DocClassVQA
Fine-Tuning Without Pretraining	70.23 (85.44)	81.25 (81.70)	35.80 (36.4)
Pretraining and Fine-Tuning	**72.69 (86.33)**	**89.28 (89.62)**	**45.83 (46.37)**

Therefore we can reach the conclusion that taking a pretrained visual question answering model like LayoutT5 is a great foundation for fine-tuning on other, maybe unrelated document tasks as long as we are able to frame them using a question, and by fine-tuning simultaneously on multiple tasks we can obtain a single multi-task model that achieves similar performance than it would if fine-tuned on each task separately.

This reduces the cost, as existing pretrainings can be used and a single fine-tuning, albeit on multiple datasets, is required instead of a fine-tune for each dataset separately.

4.2 Future Work

4.2.1 Larger Model and Larger Datasets

As has been mentioned in Chapter 1 the tendency of the field is to use larger and larger language-based models, in fact, T5, the language transformer used as backbone of our model LayoutT5, is available in much larger configurations than the ones we used.

Larger models will be able to generalize better and will most likely achieve even better results and tackle even more challenging tasks.

However larger models will also require larger datasets, which will require more resources to pretrain and fine-tune them.

4.2.2 Design Our Own Pretraining

Pretraining large models is expensive, since it requires a lot of compute, this is why in this report we used an existing pretrained model and shown that even if not pretrained in the exact task that pretraining was still useful, however an option to train a multi-task model that could achieve better performance than ours, would be to pretrain a model using tokens in the query that indicate the type of task we want to tackle.

During pretraining this tokens could be generic and then when fine-tuning they would specify the task, for example "classification:", or "information extraction: {question}", which would end up obtaining better results as the model would have no confusion on which is the current task.

4.2.3 Explainability

While not quite directly relevant in the goal of building general models for document analysis, explainability of the model's answers is very important, specially as models get larger and larger and therefore during its training have seen more and more documents.

Knowing the reason a model has chosen an answer over an other is key in order to trust its answer, because is the model giving you this answer because it has seen it in the document you are asking about or because it remembers something from one of the millions of documents it has previously seen during training?

Answering these kinds of explainability questions is not trivial and is an area of current research, it is not clear what approach would be the best.

One idea would be to not only ask the model for the text of answer but also the coordinates in the form of a bounding box of where in the document has it found the answer, while this might worked for key information extraction tasks like SROIE, what would you do for classification tasks or other tasks in which the answer to the question is not directly in the document.

Simply asking the model to justify its answer like you would tell humans in an exam will not work, or at least not directly out of the box, in order to do so a human generated dataset which not only includes the answer as ground truth but also the reasoning to get there would be necessary, which would be expensive and time consuming to create, but is it not breaking down human reasoning into small steps to hand-code algorithms what the field of artificial intelligence had been doing before deep learning?

I do not think it will scale, due to the aforementioned fact that it would require an step by step explanation of the reasoning, which in many cases would be close to impossible to obtain as humans are more black-box algorithms than we would like.

Pointing at something and saying it will not work is easy, providing a solution is the hard part, and frankly I do not have one, hence why this is the future work section.

4.3 Closing Remarks

Working on this project has been a great experience, I have thoroughly enjoyed it and has taught me a lot.

I want to thank everyone at CVC, from the staff to the other students, thank you for being so welcoming and providing such a nice learning and working environment, it has been a pleasure to share the building with all of you.

There are two persons I want to thank specially, Ruben Perez Tito, who not only developed the LayoutT5 model used in this report, but also has been a great help, answering questions and solving every problem I had, and my tutor, Dimosthenis Karatzas, whose insights and comments have helped me tremendously, I could not have asked for better supervision. So a deep thank you to both as without you this project would not have been possible.

An finally, and since these last words do not only conclude this project, but also my Bachelor's degree, I want to take the opportunity to thank all the teachers I have had, and my classmates who have been the best companions through this four year journey.

Appendix A

SROIE Result Analysis

In this appendix some of the most interesting mistakes of the model in the SROIE dataset are shown in order to get a better understanding of the capabilities and limitations of the model.

Of the 1384 samples in the SROIE inference dataset the model fails, meaning that its answer does not match the answer in the dataset's ground truth, in 557 of them, in Table A.1 the number of correct and incorrect answers for each type of question can be seen.

Most of the incorrect answers are when asked for the company or the address, which makes sense as these two fields are the most challenging, not only they are the longest but also have the most variability, in contrast dates tend to always have the same format and the total amount is a number close to the word "total".

In the case of the address, the most frequent mistake is to answer only with a partial address, this is probably due to two factors, the first one being that both in pretraining and in the other questions the model is used to produce short answers, and the other that the address spans multiple lines inside the receipt and that might confuse the model.

Another cause of mistakes in the address is problems with the OCR since the answer is considered wrong if it does not exactly match the answer in the ground truth, hence why we provide the less strict ANLS metric in all result tables.

Finally one other thing that makes the address question complicated is that the model has been mostly pretrained using English text, but these receipts, while they are also in English, come from Malaysia and the address includes many words the model might have never seen.

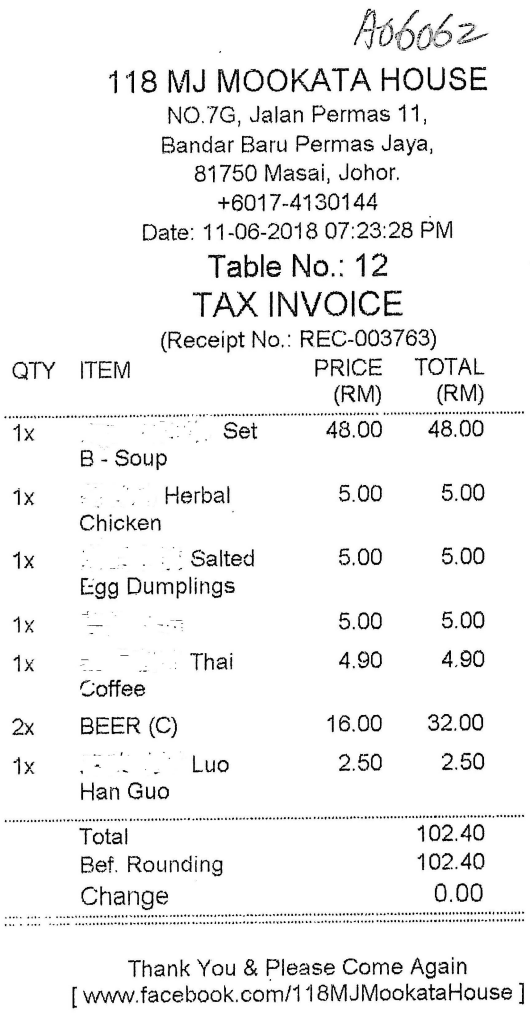
TABLE A.1: Number of correct and incorrect answers for each kind of question in the SROIE inference dataset.

	Company	Date	Address	Total
Correct	222	329	13	263
Incorrect	124	17	333	83

In the case of the company most of the mistakes are either OCR errors in one of the letters or not considering one word as part of the company name.

Figure A.1:
Ground Truth Answer: 118 mj mookata house
Predicted answer: mj mookata house

In this case the model failed to consider that the 118 is part of the company name.



A06062

118 MJ MOOKATA HOUSE
NO.7G, Jalan Permas 11,
Bandar Baru Permas Jaya,
81750 Masai, Johor.
+6017-4130144
Date: 11-06-2018 07:23:28 PM

Table No.: 12

TAX INVOICE
(Receipt No.: REC-003763)

QTY	ITEM	PRICE (RM)	TOTAL (RM)
1x	Set B - Soup	48.00	48.00
1x	Herbal Chicken	5.00	5.00
1x	Salted Egg Dumplings	5.00	5.00
1x		5.00	5.00
1x	Thai Coffee	4.90	4.90
2x	BEER (C)	16.00	32.00
1x	Luo Han Guo	2.50	2.50
	Total		102.40
	Bef. Rounding		102.40
	Change		0.00

Thank You & Please Come Again
[www.facebook.com/118MJMookataHouse]

FIGURE A.1: Id: X51007846361

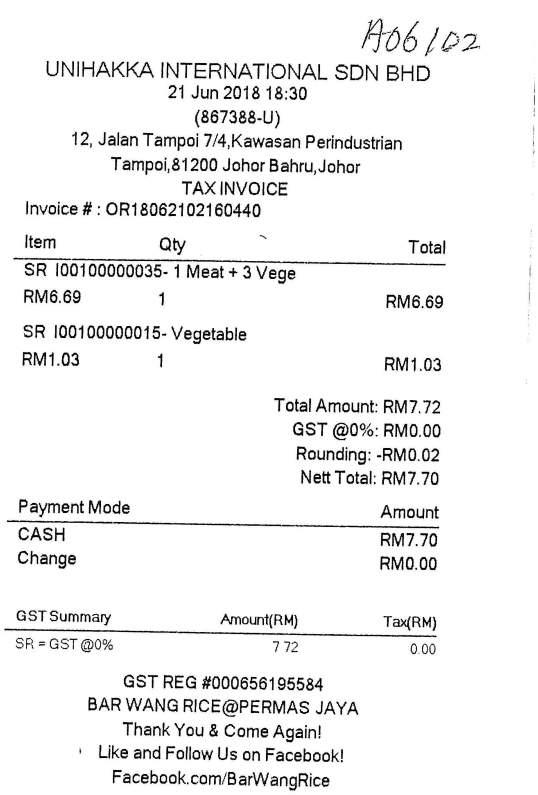
Figure A.2:
Ground Truth Answer: ali baba international sdn bhd
Predicted answer: salon du chocolat ali baba international sdn bhd

This case shows that a manually annotated dataset is always subjective, because it is possible to consider, as the model does, "salon du chocolat" as part of the name of the company.

FIGURE A.2: Id: X51005719857

Figure A.3:
Ground Truth Answer: unihakka interantional sdn bhd
Predicted answer: unihakka international sdn bhd

In this one, the ground truth answer is misspelled.


A06102
UNIHAKKA INTERNATIONAL SDN BHD
21 Jun 2018 18:30
(867388-U)
12, Jalan Tampoi 7/4,Kawasan Perindustrian
Tampoi,81200 Johor Bahru,Johor
TAX INVOICE
Invoice # : OR18062102160440

Item	Qty	Total
SR I00100000035- 1 Meat + 3 Vege		
RM6.69	1	RM6.69
SR I00100000015- Vegetable		
RM1.03	1	RM1.03

Total Amount: RM7.72
GST @0%: RM0.00
Rounding: -RM0.02
Nett Total: RM7.70

Payment Mode	Amount
CASH	RM7.70
Change	RM0.00

GST Summary	Amount(RM)	Tax(RM)
SR = GST @0%	7.72	0.00

GST REG #000656195584
BAR WANG RICE@PERMAS JAYA
Thank You & Come Again!
Like and Follow Us on Facebook!
Facebook.com/BarWangRice

FIGURE A.3: Id: X51007846397

The question of the date is the one the model answers the most correctly, most likely this is due to dates always following the same format and therefore most of the errors are when there are OCR mistakes and for example one of the numbers is read incorrectly.

Figure A.4:
Ground Truth Answer: 26/04/2017
Predicted answer: 28/04/2017

As mentioned, the most common incorrect answers for the date are due to OCR mistakes.

HON HWA HARDWARE TRADING
Company Reg. No. : 001055194X
NO 37, JALAN MANIS 7, TAMAN SEGAR,
56100 CHERAS, KUALA LUMPUR.
+603-9130 2672
GST Reg : 001125220352

TAX INVOICE

CB#: 75688 28/04/2017 11:01:36 AM
M# : C2 - 0
Cashier : cash1 -

Qty	Description	Price	Total(RM)
1.2	3/4" ALUMINIUM ROD	6.00	7.20 SR
	Item Discount		0.00
1.2	Type: 1 Total		7.20
	Discount:		0.00
	Rounding Adj		0.00
	Total Inclusive GST:		7.20
	CASH		7.20

GST Summary

Code	%	Net Amt	GST	Total(RM)
SR	SR	6.79	0.41	7.20
Total		6.79	0.41	7.20

Thank you ! & Please come again !!
Goods sold are not returnable for refund or exchange !!

FIGURE A.4: Id: X51005568887

Figure A.5:
Ground Truth Answer: 27/mar/2018
Predicted answer: 27/mar/2013

This one, while the mistake is an OCR error, the interesting thing is that despite using an slightly unusual format the model is able to understand it is a date.

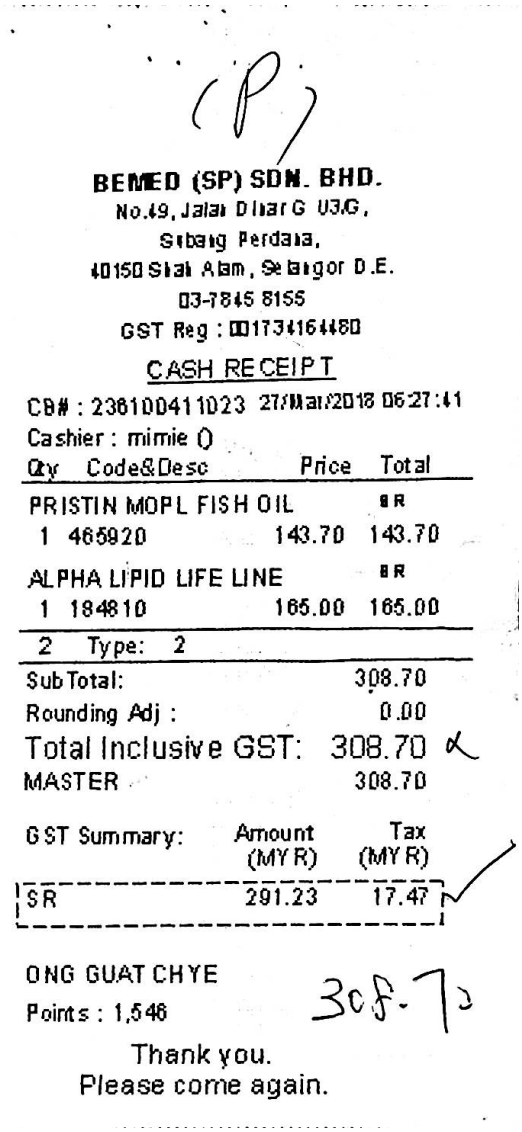


FIGURE A.5: Id: X51005746204

Figure A.6:
Ground Truth Answer: 14/04/2018
Predicted answer: 12/04/18

At first glance it could seem this must be an OCR error, however the OCR is correct in this case, while it is true that in some samples the year is shortened to the last two digits this is only done if the receipt does it as well, therefore I have no idea why the model decided to shorten the date in this case.

FIGURE A.6: Id: X51006414600

Figure A.7:
Ground Truth Answer: 11/11/16
Predicted answer: 17/11/16

This case is hard to tell, because the date is covered by a black dot, but it seems that the model is correct and the ground truth is wrong as I would say the number is indeed a 7 instead of a 1.

FIGURE A.7: Id: X51006334927

In the case of the total amount to be paid most of the mistakes are the model responding with another of the numbers on the receipt, like the total amount without including taxes, on occasion, though, the answer of the model seems fine, and it is the ground truth that might be wrong:

Figure A.8:
Ground Truth Answer: 23.02
Predicted answer: 24.55

In most cases the total amount to be paid is higher than other "totals" that appear on the receipt, like the total amount without taxes, but in this case the total amount is lower, which is probably what confused the model.

FIGURE A.8: Id: X51006556810

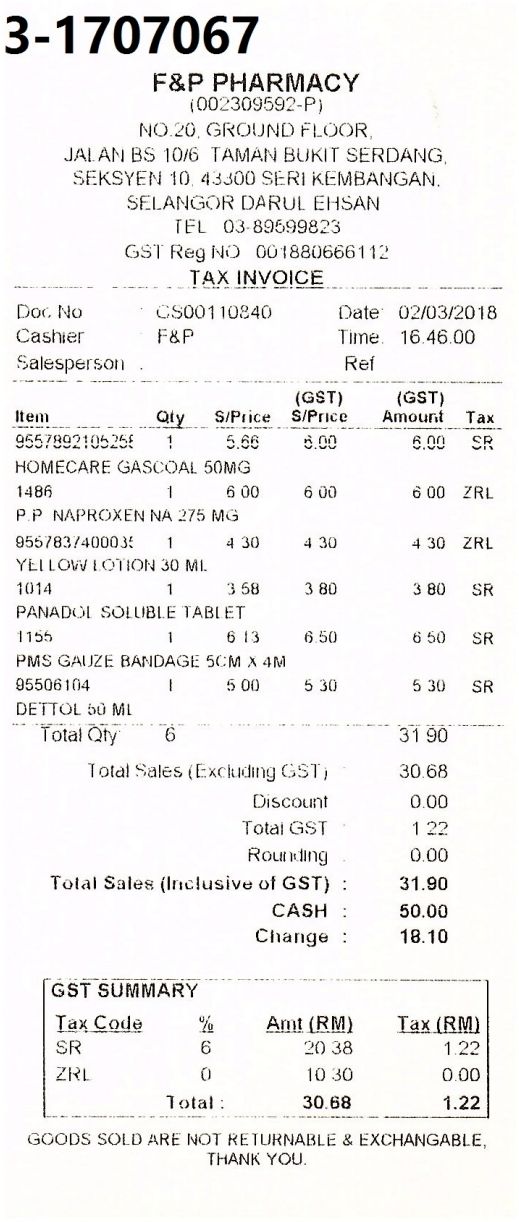
Figure A.9:
Ground Truth Answer: 43.36
Predicted answer: -5.88

It is not really clear why the model picks -5.88 as the answer, specially considering that on similar receipt it answered correctly.

FIGURE A.9: Id: X51006556809

Figure A.10:
Ground Truth Answer: 31.90
Predicted answer: 90

In this case the number the model answers with is the decimal part of the total amount, it is not clear why, as the OCR text is correct.



3-1707067

F&P PHARMACY
(002309592-P)
NO.20, GROUND FLOOR,
JALAN BS 10/6 TAMAN BUKIT SERDANG,
SEKSYEN 10, 43300 SERI KEMBANGAN,
SELANGOR DARUL EHSAN
TEL 03-89599823
GST Reg NO 001880666112

TAX INVOICE

Doc No CS00110240 Date 02/03/2018
Cashier F&P Time 16.46.00
Salesperson . Ref

Item	Qty	S/Price	(GST) S/Price	(GST) Amount	Tax
955789210525[illegible]	1	5.66	6.00	6.00	SR
HOMECARE GASCOAL 50MG					
1486	1	6.00	6.00	6.00	ZRL
P.P NAPROXEN NA 275 MG					
955783740003[illegible]	1	4.30	4.30	4.30	ZRL
YELLOW LOTION 30 ML					
1014	1	3.58	3.80	3.80	SR
PANADOL SOLUBLE TABLET					
1155	1	6.13	6.50	6.50	SR
PMS GAUZE BANDAGE 5CM X 4M					
95506104	1	5.00	5.30	5.30	SR
DETTOL 50 ML					
Total Qty:	6			31.90	

Total Sales (Excluding GST) 30.68
Discount 0.00
Total GST 1.22
Rounding 0.00
Total Sales (Inclusive of GST) : 31.90
CASH : 50.00
Change : 18.10

GST SUMMARY

Tax Code	%	Amt (RM)	Tax (RM)
SR	6	20.38	1.22
ZRL	0	10.30	0.00
Total :		30.68	1.22

GOODS SOLD ARE NOT RETURNABLE & EXCHANGABLE,
THANK YOU.

FIGURE A.10: Id: X51005365187

Figure A.11:
Ground Truth Answer: 7.20
Predicted answer: 7.65

In this case the ground truth seems to be wrong as the amount provided as an answer is the total amount without taxes, while the model correctly answers with the total amount including taxes.

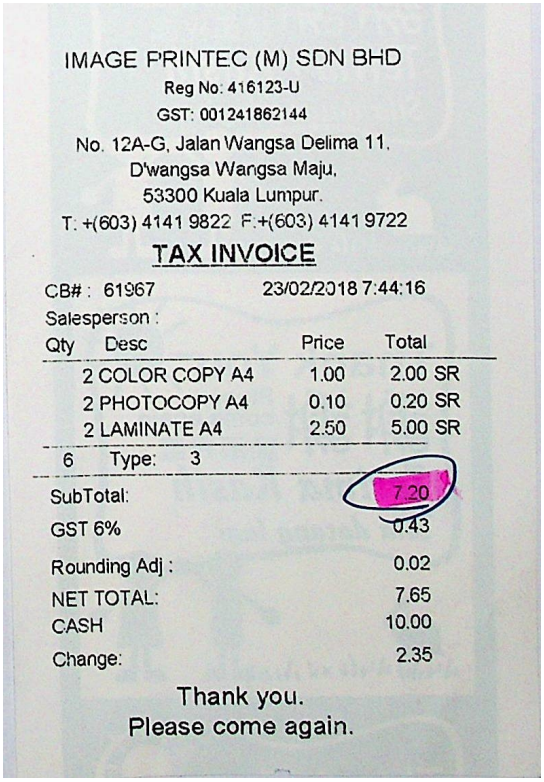


FIGURE A.11: Id: X51005806696

Figure A.12:
Ground Truth Answer: rm63.90
Predicted answer: 63.90

Here the rm before the value in the ground truth stands for Malaysian Ringgit, while it is true that in the receipt RM appears before the value I find it a bit unfair for the model, specially when other receipts where RM appears the ground truth answer does not contain it.

FIGURE A.12: Id: X51006008206

Figure A.13:
Ground Truth Answer: 436.20
Predicted answer: rm63.30

Ground Truth Answer: 23.35
Predicted answer: rm 57.20

Ground Truth Answer: 145.10
Predicted answer: rm 58.80

In these three cases the number the model gives as an answer is nowhere to be found in the receipt, what I find interesting is that when inventing a number it adds "rm" in front.

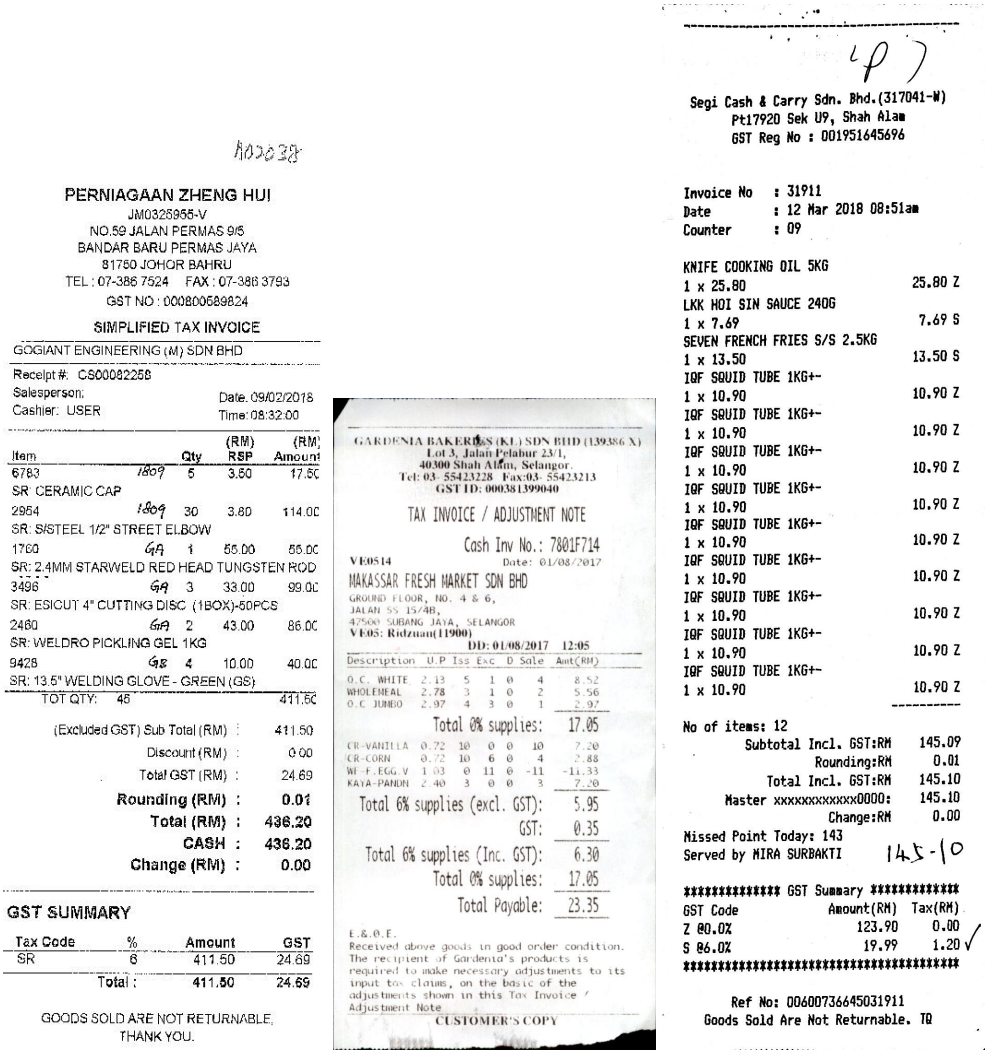


FIGURE A.13: Ids: X51005200931, X51006556740, X51005746210

Figure A.14:
Ground Truth Answer: 79.45
Predicted answer: 79.44

The amount provided in the ground truth is the total amount rounded to the nearest 5 cents, while the model answers with the other total in the receipt which is 79.44.

My understanding is that in places where one cent coins have been removed from circulation amounts are rounded to the nearest 5 cents when paying with cash (as otherwise it would be impossible to pay without the proper coin) and no rounding is done when paying with credit cards, which would not have the same problem. In this case it seems that the costumer paid using a credit card and that is why the total after listing the card is 79.44.

So I would give the model's answer as correct, this goes to show that manually annotating a dataset to generate a ground truth is not as straightforward as it seems, and that the subjective opinion of the annotator will have an impact on the model's performance.

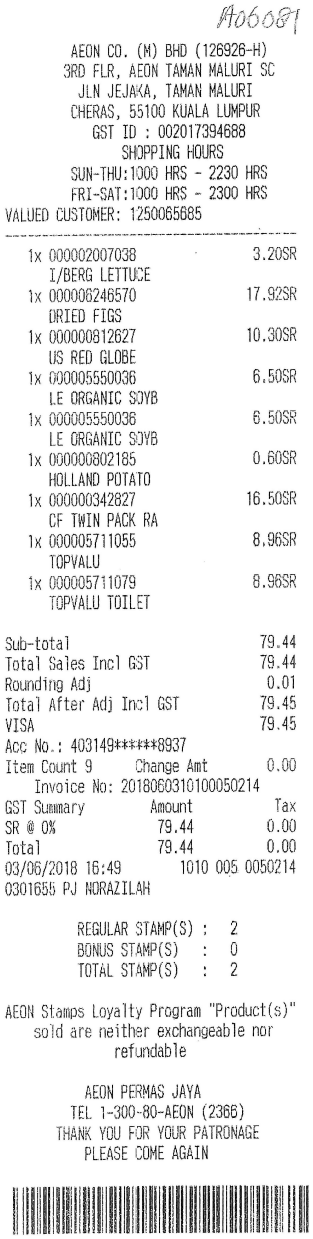


FIGURE A.14: Id: X51007846379

Figure A.15:
Ground Truth Answer: 5.15
Predicted answer: -5.15

The model answers with -5.15 because the number has been crossed with a pen which the Textract OCR read as a "-" sign, this shows that the model does not really understand what is exactly being asked about if it did it would probably have answered without the "-" sign as it would have known that a total amount that is negative probably does not make much sense.

The more interesting story in this receipt is that apparently only one of the two items listed was bought, that is why one is circled and the other is crossed, with the totals also being crossed and changed from 5.20 to 4.20. While the Textract OCR does

read handwriting it can only recognize numbers and letters and not a circle or part of the text being crossed out, this is why just using text and bounding boxes might not be enough to cover all cases and we would need models that also get the visual features of the image.

FIGURE A.15: Id: X51005605295

Figure A.16:
Ground Truth Answer: 0.90
Predicted answer: 5.00

The answer from the model is the cash handed by the costumer, but not the total amount of the receipt, which is 0.90. It is strange that the model fails here, because cash handed by the costumer appears in many receipts, and it answers those correctly, I wonder if the fact that the total is so low is what confused the model.

FIGURE A.16: Id: X51005447842

Figure A.17:
Ground Truth Answer: 5.80
Predicted answer: 5.0

This one is an OCR error, the text in the receipt is half erased and Textract OCR read 5.0 instead of 5.80, which then let to the model answering the incorrect value.

FIGURE A.17: Id: X51006502534

Another way to analyse what has the model learned is to remove the answers from the document and see what will be the model's response. To do so we have changed every appearance of the ground truth answer in the document by the word [MASK], while the bounding boxes have been left the same, this means that in the position the answer used to be, now there will be the word [MASK].

In this case, when asking about the company the most common answer is [MASK], the most likely explanation is that the model has learned that the name of the company tend to be large words on the top of the receipt, since we have changed the word, but not the bounding box, the model sees [MASK] written large on the top of the receipt and assumes it is the name of the company.

When asked about the address, the model does not respond with [MASK], it responds with some gibberish, sometimes including the name of the company. I think the reason why it does not answer with [MASK] is that while the address tends to be on top of the receipt bellow the name of the company, some times it spans just one line and others multiple lines, or in between the company name and the address there is some other information, etc. and therefore the model does not necessarily associate that position with the address, specially when it finds a single word there instead of multiple.

In a receipt there are multiple total values, in this dataset the total that is asked for is the total amount to be paid including taxes, but when that total is removed and replaced with [MASK] the model answers by using one of the other totals, like the total before tax or the cash handed by the client including change. This is a good sign as the model seems to understand that when asked for the total the answer should be a number and that is why it finds another number close to the words "total" rather than answering with [MASK].

In the same way as the total amount, the model seems to understand the format a date should have, but since there are no dates in the receipt after masking then it resorts to inventing one using the format: dd/mm/yyyy which is the most common inside the dataset.

Therefore it does seem that the model takes into account the layout of the document, because it finds the name of the company in a specific region of the receipt and considers a number close to the word "total" to be the total value, but also knows the format of the answer for each type of question, the address is composed of multiple words, the total is a decimal number, and the date also follows some specific formats.

Bibliography

LeCun, Y. et al. (1989). "Backpropagation Applied to Handwritten Zip Code Recognition". In: *Neural Computation* 1, pp. 541–551.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: 25. Ed. by F. Pereira et al. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). "Neural Machine Translation by Jointly Learning to Align and Translate". In: DOI: 10.48550/ARXIV.1409.0473. URL: https://arxiv.org/abs/1409.0473.

Vaswani, Ashish et al. (2017). "Attention Is All You Need". In: arXiv: 1706.03762 [cs.CL].

Devlin, Jacob et al. (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: arXiv: 1810.04805 [cs.CL].

Howard, Jeremy and Sebastian Ruder (2018). "Universal Language Model Fine-tuning for Text Classification". In: arXiv: 1801.06146 [cs.CL].

Peters, Matthew E. et al. (2018). "Deep contextualized word representations". In: DOI: 10.48550/ARXIV.1802.05365. URL: https://arxiv.org/abs/1802.05365.

Radford, Alec et al. (2019). "Language Models are Unsupervised Multitask Learners". In.

Shoeybi, Mohammad et al. (2019). "Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism". In: DOI: 10.48550/ARXIV.1909.08053. URL: https://arxiv.org/abs/1909.08053.

Raffel, Colin et al. (2020). "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: arXiv: 1910.10683 [cs.LG].

Brown, Tom B. et al. (2020). "Language Models are Few-Shot Learners". In: DOI: 10.48550/ARXIV.2005.14165. URL: https://arxiv.org/abs/2005.14165.

Khan, Salman et al. (2022). "Transformers in Vision: A Survey". In: *ACM Computing Surveys*. DOI: 10.1145/3505244. URL: https://doi.org/10.1145%2F3505244.

Dosovitskiy, Alexey et al. (2020). "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: DOI: 10.48550/ARXIV.2010.11929. URL: https://arxiv.org/abs/2010.11929.

Carion, Nicolas et al. (2020). "End-to-End Object Detection with Transformers". In: DOI: 10.48550/ARXIV.2005.12872. URL: https://arxiv.org/abs/2005.12872.

Zhu, Xizhou et al. (2020). "Deformable DETR: Deformable Transformers for End-to-End Object Detection". In: DOI: 10.48550/ARXIV.2010.04159. URL: https://arxiv.org/abs/2010.04159.

Ye, Linwei et al. (2019). "Cross-Modal Self-Attention Network for Referring Image Segmentation". In: DOI: 10.48550/ARXIV.1904.04745. URL: https://arxiv.org/abs/1904.04745.

Xu, Yiheng et al. (2020). "LayoutLM: Pre-training of Text and Layout for Document Image Understanding". In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. DOI: 10.1145/3394486.3403172. URL: http://dx.doi.org/10.1145/3394486.3403172.

Powalski, Rafał et al. (2021). "Going Full-TILT Boogie on Document Understanding with Text-Image-Layout Transformer". In: arXiv: 2102.09550 [cs.CL].

Kim, Geewook et al. (2021). "Donut: Document Understanding Transformer without OCR". In: arXiv: 2111.15664 [cs.LG].

Harley, Adam W., Alex Ufkes, and Konstantinos G. Derpanis (2015). "Evaluation of Deep Convolutional Nets for Document Image Classification and Retrieval". In: DOI: 10.48550/ARXIV.1502.07058. URL: https://arxiv.org/abs/1502.07058.

Graliński, Filip et al. (2020). "Kleister: A novel task for Information Extraction involving Long Documents with Complex Layout". In: arXiv: 2003.02356 [cs.CL].

Mathew, Minesh et al. (2021). "Document Visual Question Answering Challenge 2020". In: arXiv: 2008.08899 [cs.CV].

Zhang, Xingxing, Furu Wei, and Ming Zhou (2019). "HIBERT: Document Level Pre-training of Hierarchical Bidirectional Transformers for Document Summarization". In: arXiv: 1905.06566 [cs.CL].

Mathew, Minesh, Dimosthenis Karatzas, and C. V. Jawahar (2020). "DocVQA: A Dataset for VQA on Document Images". In: DOI: 10.48550/ARXIV.2007.00398. URL: https://arxiv.org/abs/2007.00398.

Biten, Ali Furkan et al. (2019). "ICDAR 2019 Competition on Scene Text Visual Question Answering". In: DOI: 10.48550/ARXIV.1907.00490. URL: https://arxiv.org/abs/1907.00490.