UNIVERSITAT AUTÒNOMA DE BARCELONA

MASTER'S THESIS

Multi-Page Document Understanding

Author:
Artemis Llabrés Brustenga

Supervisor:
Dr. Dimosthenis Karatzas
Dr. Ernest Valveny

Final Project for the
Master's Degree in Research and Innovation in Computer based Science and Engineering

Vision, Language and Reading Group
Computer Vision Center

July 8, 2024

UNIVERSITAT AUTÒNOMA DE BARCELONA

Abstract

Master's Degree in Research and Innovation in Computer based Science and Engineering

Multi-Page Document Understanding

by Artemis Llabrés Brustenga

Document Understanding is a field that focuses on enabling machines to analyze and interpret documents, which are inherently designed for human consumption with complex structures including text, layout, tables, and figures. Traditional Large Language Models (LLMs) struggle with these tasks due to their reliance on text alone. In this work, we focus on Multi-Page Document Visual Question Answering (DocVQA), which presents additional challenges compared to single-page document understanding, such as handling increased information density and navigating between multiple pages. Our contributions include the implementation of an open-source version of Amazon's multi-page GRAM model called MP-VT5 and the development of a new pixel-only multi-page model: MP-Pix2Struct. MP-VT5 improves upon the original architecture by using global tokens in the decoder instead of having an extra sequence compression stage and adding a page prediction module. MP-Pix2Struct adapts a pixel-only model to the multi-page scenario using a global-local encoder strategy. These advancements aim to enhance the efficiency and accuracy of multi-page Document Understanding models.

Contents

Chapter 1

Introduction

1.1 Document Understanding

Documents are fundamental to the functioning of human society, providing a reliable means of recording transactions, preserving information, and facilitating communication. However, documents are usually designed for human consumption, making them difficult to be automatically processed. That is why the field of Document Understanding designs techniques and methods that allow machines to analyse and interpret documents.

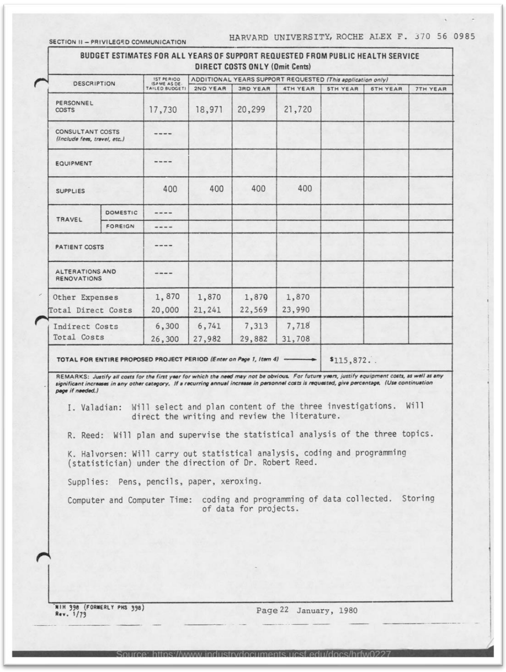
Information in documents is not only conveyed through their text; documents have layout, tables, figures, and various visual elements that contribute significantly to their content and structure. That is why Large Language Models (LLMs), that only rely on text, are not well suited for document analysis tasks.

There are two main approaches to design Document Understanding models: Pixel-Only models [1, 2] that rely on the fact that an image of a document contains all necessary information to understand it. However, in order for the model to read the text, the image must be in high resolution, making these models more computationally expensive. Text-Image-Layout models [3, 4, 5] that use the text, the layout, which is the two-dimensional position of every word, and the image, which, since it only has to provide visual context, can be smaller than in Pixel-Only models. The drawback of these models is that they require an Optical Character Recognition (OCR) system to extract the document text and layout in the first place.

Another challenging aspect of documents is their multi-page nature, which both hinders the understanding of the document and poses a challenge as processing longer and longer documents becomes more computationally expensive.

1.2 Document Visual Question Answering

The task of Document Visual Question Answering (DocVQA) has the goal of training end-to-end models to answer questions over a given document, like in Figure 1.1. This way it moves the Document Understanding field away from having decoupled information extraction and information processing tasks to having end-to-end generic document understanding. DocVQA models will implicitly perform all necessary information extraction and information processing tasks in order to correctly answer the question.



SECTION II – PRIVILEGED COMMUNICATION

HARVARD UNIVERSITY, ROCHE ALEX F. 370 56 0985

BUDGET ESTIMATES FOR ALL YEARS OF SUPPORT REQUESTED FROM PUBLIC HEALTH SERVICE
DIRECT COSTS ONLY (Omit Cents)

DESCRIPTION		1ST PERIOD (SAME AS DETAILED BUDGET)	ADDITIONAL YEARS SUPPORT REQUESTED (This application only)					
			2ND YEAR	3RD YEAR	4TH YEAR	5TH YEAR	6TH YEAR	7TH YEAR
PERSONNEL COSTS		17,730	18,971	20,299	21,720			
CONSULTANT COSTS (Include fees, travel, etc.)		----						
EQUIPMENT		----						
SUPPLIES		400	400	400	400			
TRAVEL	DOMESTIC	----						
	FOREIGN	----						
PATIENT COSTS		----						
ALTERATIONS AND RENOVATIONS		----						
Other Expenses		1,870	1,870	1,870	1,870			
Total Direct Costs		20,000	21,241	22,569	23,990			
Indirect Costs		6,300	6,741	7,313	7,718			
Total Costs		26,300	27,982	29,882	31,708			
TOTAL FOR ENTIRE PROPOSED PROJECT PERIOD (Enter on Page 1, Item 4) →						$115,872.		

REMARKS: Justify all costs for the first year for which the need may not be obvious. For future years, justify equipment costs, as well as any significant increases in any other category. If a recurring annual increase in personnel costs is requested, give percentage. (Use continuation page if needed.)

I. Valadian: Will select and plan content of the three investigations. Will direct the writing and review the literature.

R. Reed: Will plan and supervise the statistical analysis of the three topics.

K. Halvorsen: Will carry out statistical analysis, coding and programming (statistician) under the direction of Dr. Robert Reed.

Supplies: Pens, pencils, paper, xeroxing.

Computer and Computer Time: coding and programming of data collected. Storing of data for projects.

NIH 398 (FORMERLY PHS 398) Rev. 1/73

Page 22 January, 1980

Source: https://www.industrydocuments.ucsf.edu/docs/hrfw0227

Question-Answers:

- **Question:** What is the personnel costs in the 4th year?
  - **Answer:** '21,720'
- **Question:** What is the total costs for proposed project period?
  - **Answer:** '$115,872'
- **Question:** What is the name of the University mentioned on top?
  - **Answer:** 'Harvard University'
- **Question:** What is the personnel costs in the first period?
  - **Answer:** '17,730'
- **Question:** What does supplies include?
  - **Answer:** 'pens, pencils, paper, xeroxing.'
- **Question:** What is the cost of supplies for the 3rd year?
  - **Answer:** '400'

FIGURE 1.1: Examples of questions over a single-page document from the DocVQA dataset.

This task was introduced in 2020 with the DocVQA dataset [6], it consists of 50K questions asked over 12K single-page documents. DocVQA follows the same format as generic VQA [7] or Scene Text VQA [8] tasks which pose questions over natural images. However, approaches working well on these tasks are not well suited for DocVQA, as documents have a much higher information density, and require the understanding on an entirely different set of visual features.

Since its introduction the DocVQA dataset has been a key benchmark to evaluate Document Understanding models, it is used by both specifically designed document models like Microsoft's UDOP [5], and generic Multimodal Large Language Models (MLLMs) like OpenAI's GPT4-Vision[1] and Google DeepMind's Gemini [9].

1.3 Multi-Page Document Visual Question Answering

The original DocVQA dataset has questions over single-page documents, and hence it does not tackle the extra challenges that appear when working on multi-page documents. That is why in 2023 the MP-DocVQA [10] dataset was introduced, which contains questions over documents with multiple pages. The dataset has 46K questions asked over 6K documents with a total 48K pages. At most a document will have 20 pages.

The dataset was constructed by taking the single-page questions from the DocVQA dataset and posing them over the complete multi-page document the single-page came from. This required removing questions that became ambiguous when posed over multiple pages. Since the questions were originally for single-page documents

[1]Evaluation at: https://openai.com/research/gpt-4

all information necessary to answer the question can be found in one page, meaning that the rest of the pages can be thought of as distractors. Therefore the model will have to learn to select the right page to answer the question.

1.4 State of The Art

Document Understanding and DocVQA, much like the rest of the deep learning field, have been taken over by the Transformer architecture [11]. First with encoder only models, often based on BERT [12], with models like M4C [13] and LayoutLMv2 [14], which combine the text embeddings, visual embeddings, and layout embeddings into a single multimodal transformer, with the answer being predicted by either classifying over a fixed set of possible answers or by predicting the start and end location of the answer in the document. Therefore, these models are limited in the kind of answers they can provide. Hence, generative models that use an encoder-decoder transformer emerged. As previously mentioned there exist two big families of models for Document Understanding, Text-Image-Layout and Pixel-Only models.

The idea behind Text-Image-Layout models is to leverage existing encoder-decoder language models, usually T5[15], and include in the encoder the extra visual and layout information necessary to properly understand documents. TILT [3], DocFormerV2 [4], and UDOP [5] are the three main examples of this approach. The three models embed the text, image, and layout as follows:

- **Text Embedding:** In all three models text is encoded in the same way as in T5, by tokenizing the text and performing vocabulary lookup in a learned embedding matrix. Tokenization will split the text into subwords known as tokens, and turn it into a list of indexes according to a predefined vocabulary. For each index, its corresponding row from the embedding matrix is taken as the embedding vector of that token. This is expressed as a matrix multiplication between the token matrix generated by staking the n one-hot encoding vectors of the token indices $T \in \{0,1\}^{n \times z}$ and the embedding matrix $W_{text\_emb} \in \mathbb{R}^{z \times d}$, where n is the number of tokens in the text, z the vocabulary size, and d is the embedding dimension. Therefore, the embedded text is $E_{text} = T \times W_{text\_emb}$.
- **Image Embedding:** Each of the three models take different approaches to add images to the original T5. TILT uses a U-Net convolutional neural network (CNN) that produces a feature map from the entire page image, then they use the bounding box around each text token to extract features from the map with ROI pooling. This features are then summed to the embedding of each text token. Both DocFormerV2, and UDOP, follow the approach of ViT[16]. It consist of first extracting visual tokens or patches from the image, this is done by reshaping the image $I \in \mathbb{R}^{h \times w \times c}$ into a sequence of flattened 2D patches $P \in \mathbb{R}^{m \times (p^2 \cdot c)}$, where h, w, and c are the height, width, and number of channels of the image, and p is the patch size, with $m = \frac{h \cdot w}{p^2}$ being the number of patches. Then the sequence of patches P is embedded with a linear layer $W_{patch\_emb} \in \mathbb{R}^{(p^2 \cdot c) \times d}$, meaning that the image embedding is $E_{image} = P \times W_{patch\_emb}$. In the case of DocFormerV2 the image embedding is concatenated to the text embedding. While in UDOP the text embedding of each text token is summed to the corresponding patch embedding, by checking

in which patch is the centre of the bounding box associated to the text token located in. This means that each text token embedding is summed to one image patch embedding, which are then concatenated with the rest of image patches that have no text token associated.

- **Layout Embedding:** Two approaches are followed to incorporate layout information into the model. DocFormerV2, embeds the normalised bounding box $b = (x_1, y_1, x_2, y_2)$ with four different embedding layers: W_x for horizontal coordinates, W_y for vertical coordinates, W_h for the height of the box, and W_w for the width of the box. Therefore, the layout embedding is $E_{layout} = W_x(\vec{x}_1) + W_x(\vec{x}_2) + W_y(\vec{y}_1) + W_y(\vec{y}_2) + W_h(\vec{y}_2 - \vec{y}_1) + W_w(\vec{x}_2 - \vec{x}_1)$.[2] Then, the corresponding layout embedding is summed to each text token embedding and image patch embedding. Meanwhile, TILT and UDOP, instead of creating a layout embedding, modify the relative attention bias used in T5 from 1D to 2D.

While text-image-layout models have achieved impressive results on Document Understanding tasks, their reliance on OCR to obtain text and layout adds extra cost and makes them unsuitable for some applications. That is why pixel-only models which only require the document image are a good alternative. Donut [1], uses a Swin Transformer [17] as an image encoder, that creates m image patch embeddings of dimension d, these image embeddings are concatenated with the question embeddings, which are the result of tokenizing and embedding the text tokens of the question. The decoder, in this case BART [18], takes the embeddings and generates the answer. One problem with this architecture is that the encoder does not have access to the question, and hence, it creates a generic representation of the document image. If the encoder had the question, then the image embedding would be influenced by the question, creating a more useful representation. Pix2Struct [2] solves this by rendering the question at the top of the image, then like in a ViT the image is tokenized into image patches and each patch is embedded using a linear layer. However, unlike in ViT where the image is resized into a square, Pix2Struct maintains the original aspect ratio of the document image. Since patches are introduced as a sequence, the model must be informed of the aspect ratio and the location of each patch in the image, this is achieved via a layout embedding, where each patch is summed with the embedding of the horizontal and vertical coordinates of its centre. Following the same notation described above, this would be: $E_{image} = P \times W_{patch\_emb} + W_x(\vec{x}_{centre}) + W_y(\vec{y}_{centre})$. Since the question was already present in the image, the only input used by the decoder to generate the answer are the image embeddings.

However, all these models have been designed for single-page document understanding. If given a question over a multi-page document, these models would give poor results, that is, if it is even possible to run the model over the multi-page document, as the computational requirements might be too demanding. The principal culprit that hinders transformer models from working with long input sequences (like all the text and image tokens of many pages) is the self-attention mechanism. In self-attention each input element is transformed into three vectors: a query vector Q, a key vector K, and a value vector V. The attention scores are computed by taking

[2] A simplification of the notation is used: $\vec{x}_1$, $\vec{x}_2$, $\vec{y}_1$, $\vec{y}_2$ are vectors containing each coordinate from all bounding boxes, and doing the function W over the vector $\vec{v}$ is understood as computing $W(\vec{v}) = V \times W$, where V is the stacking of the one-hot encoding of each element of $\vec{v}$.

the dot product of the query vectors with the key vectors, scaled by the square root of the dimension of the key vectors, and applying a softmax function.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1.1}$$

where $Q, K, V \in \mathbb{R}^{n \times d_k}$ for a sequence of length n and a key dimension d_k. The term QK^T involves a matrix multiplication that results in an $n \times n$ matrix, representing the pairwise attention scores for all elements in the sequence. This operation is computationally expensive, with a time complexity of $O(n^2 d_k)$, leading to quadratic scaling with respect to the sequence length n. Consequently, if the input is a very long sequence of concatenated tokens from all pages, the self-attention mechanism can become a bottleneck as the required computation and memory resources increase quadratically. That is why multi-page models, find strategies to not encode all pages as a single sequence, but rather to separate the encoding between pages.

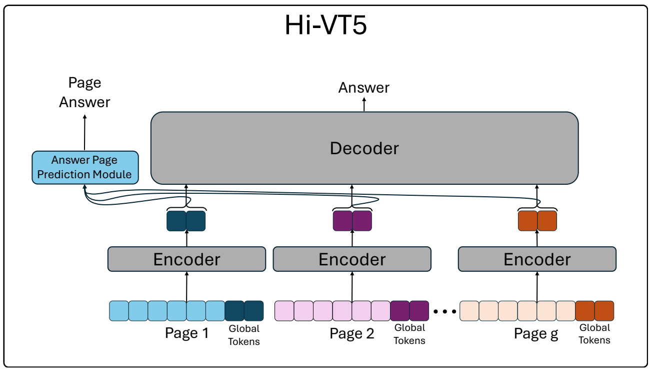


FIGURE 1.2: Hi-VT5 architecture with the fully separate encoders that process each page independently, the decoder which takes in the global tokens of all pages and generates the answer, and the answer page prediction module, that from the global tokens produces a prediction of which page contained the answer.

Hierarchical VT5 or Hi-VT5 [19], pictured in Figure 1.2, is a multi-page version of VT5, which is a text-image-layout model very similar to DocFormerV2. To reduce computational complexity each page is encoded independently, this means that for each page the encoder receives the question, the document text, and the image. This not only has the advantage of being more efficient, but also means that for the encoder the task is not different from single-page DocVQA. However, the answer is generated in the decoder, and giving the decoder the output of encoding all pages would still be too much, also, there is the fact that most pages will contain information that is irrelevant to answer the question, that is why together with each page a set of special learnable tokens are encoded, and only these tokens from every page will be the ones used in the decoder to produce the answer. Therefore, if the document has g pages and k special tokens are used per page, the decoder instead of receiving a sequence of length $g \cdot (n \cdot m)$, where n and m are the number of text tokens and image patches, it would only receive a sequence of $g \cdot k$. In order to provide some explainability of the model answer and a complementary supervisory signal during training, a prediction of which page was the answer found is obtained by sending the k special tokens to a page prediction module.

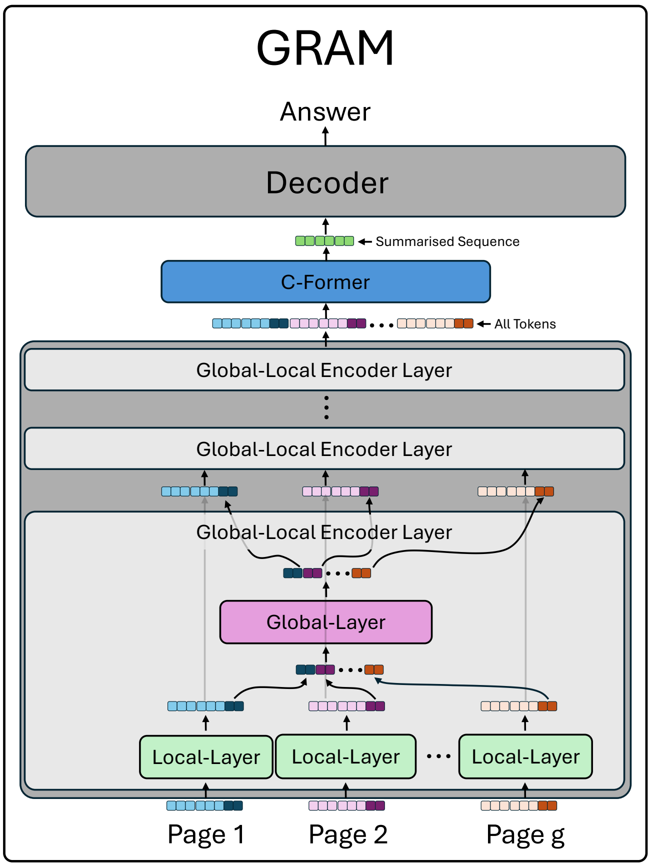


FIGURE 1.3: GRAM's architecture with a Global-Local Encoder, that interleaves local-layers with global-layers. The C-Former reduces the input sequence length to the decoder that will produce the answer.

One issue with the Hi-VT5 approach is that since each page is encoded independently there is no information shared between the pages until the decoder, GRAM solves this by introducing a global-local encoder, as seen in the architecture diagram of Figure 1.3. The idea is to interleave local-layers that process each page independently with global-layers that have a complete vision of the document by processing all pages simultaneously. This is implemented on top of DocFormerV2, keeping its original encoding layers as local-layers and adding the new global-layers. Similarly to Hi-VT5, k global tokens, are added to each page, in local-layers these tokens are processed with their corresponding page, and in global-layers only the global tokens of all pages are processed together. After the global-local encoder, all tokens from all pages are concatenated and the sequence is passed through a small transformer decoder, they call C-Former (Compression Transformer), which uses N_c learnable query tokens to reduce the length of the sequence.

While these approaches scale better than simply concatenating all pages, their computational footprint will still increase with the number of pages. In cases where only a few of the document pages are needed to answer the question, an approach like the one proposed in Multi-Page Document Visual Question Answering using Self-Attention Scoring Mechanism [10] can be used. The idea is to separately encode each page with a copy of the question, then independently between pages, a score is computed from the encoded features. The score is computed through a self-attention mechanism, which since it only processes a page at a time is efficient enough. The higher the score the more likely the question is to be related to the contents of the page, hence only the pages with highest scores are send to the decoder to produce the answer.

1.5 Contributions

In this work we implement an open-source version of Amazon's multi-page GRAM model, while also proposing and evaluating modifications to the architecture. Like GRAM we base our multi-page model on an existing single-page model, in our case VT5, hence why we call it MP-VT5. The main proposed modification is to remove GRAM's C-Former and directly use the global tokens as a summarised representation of each page for the decoder. We show that adding a page prediction module improves results, similar to Hi-VT5, but in our case the page prediction is done after the decoder. We test the possibility of having the question only in the global layers, instead of having copies of the question for each page.

We also introduce the MP-Pix2Struct model, which adapts an existing pixel-only model to the multi-page scenario, using the same global-local encoder strategy as GRAM and MP-VT5.

Chapter 2

Datasets

2.1 MP-DocVQA

As introduced in section 1.3 the Multi-Page Document Visual Question Answer (MP-DocVQA) dataset [19], expands the original DocVQA dataset [6] to the multi-page setting. The questions of the DocVQA dataset were over a single page, however, the documents from which the page was extracted contained more pages. While some of the documents contained over a hundred pages, the documents in the dataset are kept to a maximum of 20 pages. Since the questions were originally designed to be asked over only one page, some questions, like "What is the page number?" or "What is the title of the section?", became ambiguous with the extra pages, as they could be answered with the information from another page than the one intended, this required the removal of some questions.

In total the dataset contains $46,176$ questions posed over $47,952$ pages from $5,928$ documents. They are split in train, validation, and test sets that follow the same distribution as the single-page DocVQA dataset to ensure that a model trained on the single-page scenario has not seen the same questions and pages from the validation or test set of the multi-page dataset.

To facilitate the use of Text-Image-Layout models, not only the page images are provided, but also a high-quality OCR of the text in the page, and the location of each word in the form of bounding boxes.

2.2 OCR-IDL

The OCR-IDL dataset [20] is a dataset of OCR annotations for a large corpus of industry documents. It contains 4.6 million documents, with a total 26 million pages. The documents were obtained from UCSF's Industry Document Library [1], with documents from 1930 to 2019. There is a great variety of document formats, from letters and emails, to forms and news articles, meaning that the layout of the documents is also varied. Many documents also include visual elements like tables, charts, and figures.

Since the only annotation available for this dataset is the OCR text and layout of each page, we use this dataset for unsupervised pretraining through masked span de-noising. However, we also created a synthetic multi-page question-answering

[1] https://www.industrydocuments.ucsf.edu/

dataset by generating questions with Llama3[2], over a subset of the ORC-IDL documents. Since Llama3 is a pure language model it can only process documents as unidimensional string of text, with no knowledge of the document layout or visual characteristics. Despite this, the questions generated by this model are a good pretraining for the MP-DocVQA task. In total we generated 1.6 million questions for 160 thousand two-page documents. This dataset will be published and open-sourced together with the developed models to facilitate reproducibility of results and spur further research.

[2] https://ai.meta.com/blog/meta-llama-3/

Chapter 3

Model Development

3.1 MP-VT5

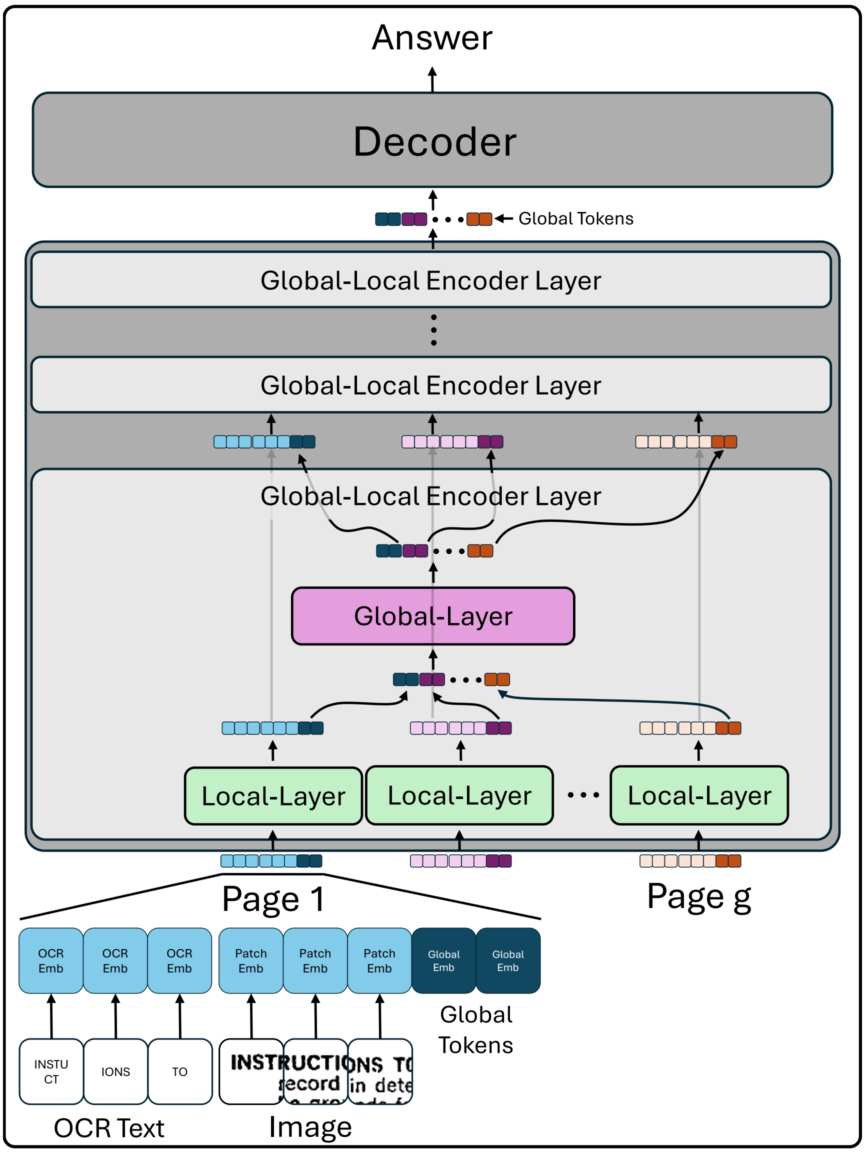


FIGURE 3.1: The MP-VT5 architecture.

The MP-VT5 is our open-source version of Amazon's multi-page GRAM model. As explained in section 1.4, GRAM implements a global-local decoder on top of DocFormerV2, converting this way a single-page model into a multi-page one.

The goal of this encoder is to allow the communication of information between pages in the model encoder without having to actually encode all pages as a single sequence. As can be seen in Figure 3.1 this is done by interleaving local-layers with global-layers in the encoder. The local-layers are the same as in the single-page model used as backbone, their job is to encode each page independently. Each page consists of the embedded text $E'_{text} \in \mathbb{R}^{n \times d}$ which also includes the embedding of the question as the initial text, the image patches $E'_{image} \in \mathbb{R}^{m \times d}$, and k learnable global tokens $E'_{global} \in \mathbb{R}^{k \times d}$, note that in all cases these embeddings are layout contextualised, meaning that the embedding of each token has been summed with the embedding of its bounding box[1], thus we use the notation of E' instead of E to indicate this difference. The assumption is that during self-attention in the local-layer the global tokens will capture information of the page, while the other tokens of the page will receive the global information the global tokens contain. The global layers are newly initialised layers that operate only over the global tokens, but they do so over the global tokens of all pages simultaneously. If there are g pages in the document, the global layer will receive $E'_{global} \in \mathbb{R}^{(g \cdot k) \times d}$. In these layers, global tokens can communicate information from their page to the global tokens of other pages, and also capture the global information that in a local-layer will be transmitted to the tokens in their page. While the input of global-layers also grows with the number of pages, since k is much smaller than the full page sequence $n + m$, the global layers remain efficient even for a large number of pages. To further ensure a low computational complexity, the number of parameters of global layers is reduced by changing the dimension of the feed-forward projection matrix, the number of attention heads, and the dimension of the key/value projection matrix.

Since DocFormerV2 is also a proprietary Amazon model, we base our implementation on another Text-Image-Layout model, the VT5. Like, DocFormerV2, and unlike TILT and UDOP, VT5 also creates a layout embedding from the bounding boxes of the tokens which is then summed with the embeddings of those tokens. However, instead of composing the layout embedding from two corners of the bounding box and its height and width, only the corners are used, as height and width can be computed from the corners and is redundant. Another difference is the image embedding, DocFormerV2 simply splits the image in patches and embeds them with a linear layer, the original VT5 uses a pretrained ViT on document images, instead we modified VT5 to also use a single linear layer to embed the patches, however, while DocFormerV2 resizes the image to a square, we follow the Pix2Struct approach to keep the original aspect ratio. Furthermore, since document images contain a lot of empty space, we remove patches that have no meaningful content, as described in section 3.3. Besides these architectural differences, there are also training differences. DocFormerV2 has two extra sub-tasks in the encoder during pretraining, a token-to-line task, and a token-to-grid task, which are supposed to improve its layout understanding. Furthermore, DocFormerV2 has received an extensive pretraining, using 64 million document pages.

When implementing MP-VT5 we made some modifications to the GRAM architecture, the most significant is to remove the Compression Transformer or C-Former. In GRAM, the C-Former is a small transformer decoder, located between the encoder

[1]The global tokens do not have a bounding box, as they do not have a location inside the page, however, for consistency with the other embeddings and to simplify the implementation, they are summed with the embedding of the bounding box [0, 0, 1, 1] which covers the entire page.

and the decoder, its goal is to take the entire sequence of all tokens from all pages and produce a shorter sequence that summarises all pages and can be used by the decoder to produce the answer. While we agree that it is a good idea to have some form of summarisation in order to reduce the computational demands of the decoder, we believe that a better approach is to leverage the existing global tokens, that already contain a condensed representation of the entire document. Therefore, instead of using a C-Former, the decoder of our model receives the global tokens from all pages. One drawback of our approach is that the input sequence length to the decoder will be $g \cdot k$, where g is the number of pages, and k the number of global tokens per page, while the C-Former produces a sequence with a predefined length, meaning that our decoder input length will grow with the number of pages while GRAM's will not.

Another implementation change is regarding the initialisation of the global tokens, in GRAM k global tokens are created for each page, however, these tokens are identical between pages, while it is true that after the first local-layer the tokens will be differentiated due to each containing information from their page, we decided to further differentiate the global tokens from their initialisation. What we do is to add the embedding of the page number the global tokens belong to. Instead of doing this through a learned embedding, we use a sinusoidal positional embedding, which by sampling from a set of sine and cosine functions will create vectors of the model embedding size d for each of the page numbers.

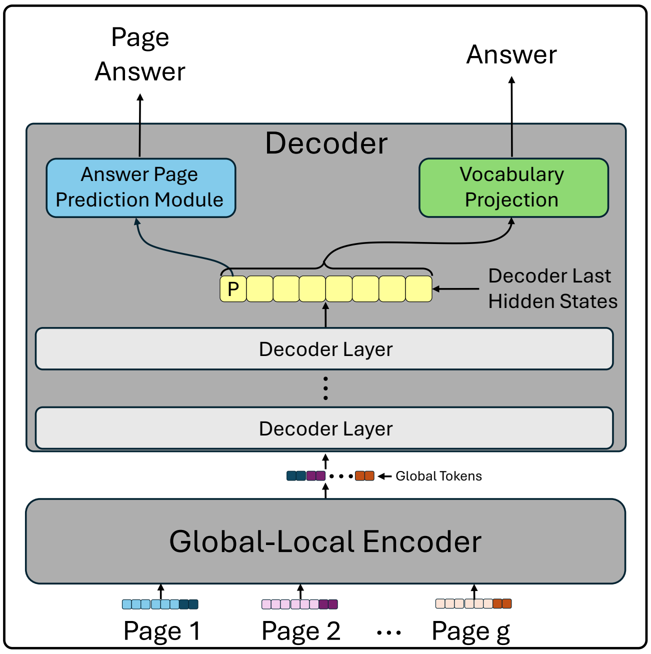


FIGURE 3.2: Diagram of the answer page prediction in MP-VT5.

The Hierarchical VT5 proposes using an answer page prediction module that from the global tokens predicts in which page of the document the answer is located. This is offered as a way to provide some explainability for the model answer, but also is shown to slightly improve the model answers, since it provides an extra supervisory signal during training. We decided to also implement answer page prediction in the MP-VT5, however, instead of doing the prediction from the global tokens after the encoder, we do it after the decoder, this way the answer page prediction signal will influence both the encoder's and decoder's parameters, instead of only the former's.The idea, shown in Figure 3.2 is to take the output tokens of the decoder after its last transformer layer, just before the tokens are projected to the vocabulary space. The answer page prediction module will be a linear layer that takes each output token of size d and produces a single real value for each. Of course, we do not need an answer page predicted for each token of the answer, since the entire answer will belong to one page, still we wanted to create an approach that was generic and that could indicate that multiple answers or part of the answer come from different pages. To do so, we included the token *<page>*, when this token is produced by the decoder we know that we must check the value produced by the answer page prediction module, for other tokens the predicted value is ignored. During training we added the *<page>* token at the beginning of each answer, and added an answer page prediction loss that is computed only for this first token. Therefore, the full model loss is the cross entropy loss between the tokens of the ground-truth answer and the model logits from all tokens, plus the mean squared error loss between the ground-truth answer page and the value predicted in the first token by the answer page prediction module.

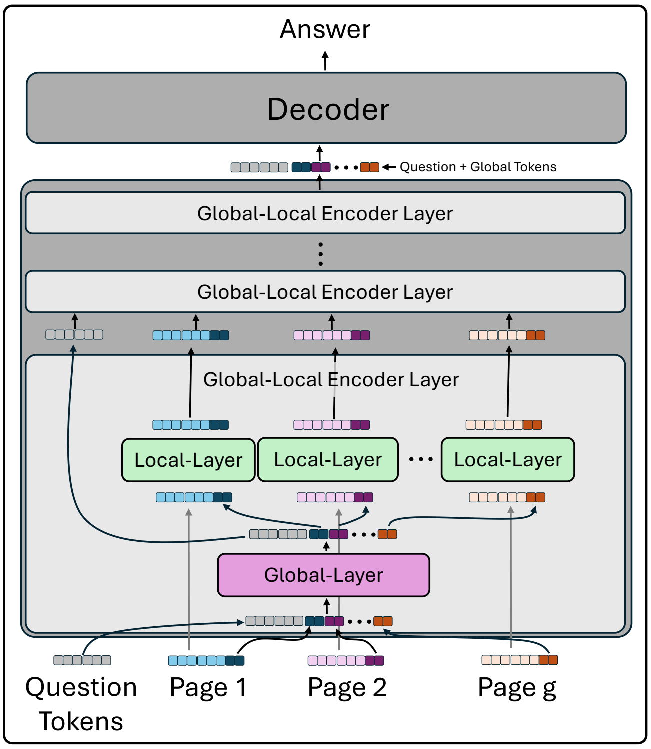


FIGURE 3.3: The MP-VT5 architecture with the question in the global-layers and in the decoder. Since the local-layers do not have the question, the encoder starts with a global-layer to get the question information to global tokens.

Another idea we wanted to test was to have the question in the global-layers, see Figure 3.3. Having the question in the global layers would mean only having a single copy of the question tokens in the model instead of each page having a copy in the local-layers, this way, the question can also be added to the decoder input. With the question in the global-layers it made sense to start with one of these layers instead of a local one, this way through self-attention the global tokens would get contextualised with the question, bringing this information to their pages in the local-layer. While this is a sound idea it creates a big departure from the single-page model, since the local-layers are initialised from the single-page model layers, they have been trained to have the question together with the rest of the page tokens, but now these layers find the question information in the global tokens represented in a way that is far from the original embedding of the question. For the model to learn to properly use the question in this way significant training is needed, that is why we came up with pretraining strategies that included questions, which we thought would be useful to make this version of the MP-VT5 work.

3.1.1 Pretraining

The local-layers and the decoder of our model are initialised from an already trained single-page model, but the global-layers are newly initialised, furthermore, the original decoder is only trained to use the entire token sequence of one page, while in our case it will be getting only the global tokens. While the model can learn these new things only by training on the MP-DocVQA train set, it is still a good idea to design pretraining tasks on a larger dataset, this way the model parameters will start from a more optimal state when finetuned on MP-DocVQA.

The pretraining strategy of the original language-only T5 model was span de-noising. This is a self-supervised task that consists of giving the model a text from which some spans have been masked, and the model's goal is to predict what was in those masked spans. Therefore, the input could be something like:

"Documents <extra_id_0> to the functioning of <extra_id_1>, providing a <extra_id_2> transactions, <extra_id_3>, and <extra_id_4> communication."

Where the <extra_id_x> indicate the masked spans. Then the answer the model should produce would be:

"<extra_id_0> are fundamental <extra_id_1> human society <extra_id_2> reliable means of recording <extra_id_3> preserving information <extra_id_4> facilitating <extra_id_5>"

Which is the inverse of the original input text, here the extra ids are used to mask the spans that were present in the input while the rest is unmasked. This task was extended for VT5 to include the layout and the image of the document page, calling it layout-aware span de-nosing. The layout is added to each token by summing the embedding of its bounding box, for the masked spans a single bounding box formed by joining the bounding boxes of the tokens of the span is used. This gives the model the location of the masked spans in the document, which encourages the model to fully utilise the layout information and the visual information to solve this task. In the case of MP-VT5 this pretraining is further extended to use multiple pages as inputs, and since only the global tokens are used in the decoder, the model is forced to learn to create good representations of each page in those tokens.

To further pretrain the model this time in a task closer to the MP-DocVQA dataset, we created a dataset of synthetically generated questions with Llama3, as described in section 2.2. Generated data is always more noisy and prone to errors than human annotated data, however, it makes up for it through quantity, with 1.6 million questions instead of the 46K from the human annotated dataset.

For the MP-VT5 with the question in the global-layers we came up with a modification of multi-page layout-aware span de-noising, which included questions. Instead of having the <extra_id_x> tokens to indicate which regions had been masked, we used a question with a layout location. Therefore, the question and the input to the model would be: Question:

> "What are the missing words at <loc_0> <loc_1> <loc_2> <loc_3> <loc_4>?

"Documents to the functioning of, providing a transactions, , and communication."

Where the <loc_x> tokens are summed with the embedding of the bounding box of the missing text, giving the model a clue of where are the spans to be reconstructed. The answer of the model would be the missing text at each location as follows:

> "<loc_0> are fundamental <loc_1> human society <loc_2> reliable means of recording <loc_3> preserving information <loc_4> facilitating <loc_5>"

This task has two main benefits, the first is that it includes a question that can be separated from the input text and hence be used in the global-layers, the second is that it is a much more layout intensive task than regular layout-aware span de-noising, as the model needs to understand the layout locations in the <loc_x> tokens in order to complete the task.

3.1.2 Model and training hyperparameters

The MP-VT5 model that we uses is based on a T5-base, this means that both the encoder and the decoder have 12 layers, it has an embedding dimension of 768, the feed-forward projection matrix of each layer is of size $3,072$, and the attention mechanism is composed of 12 heads with a key/value projection matrix of dimension 64, with a vocabulary size of $32,128$ tokens it has in total 220 million trainable parameters. When adding the layout embedding, and image embedding of VT5 the number of parameters increases to 225 million. The MP-VT5 version with its extra global-layers in the encoder which have a feed-forward projection matrix of size $1,024$, and 4 attention heads with dimension 64 for the key/value projection matrix, has a total number of parameters of 253 million.

Finetuning on the MP-DocVQA dataset is done for 10 epochs, using two pages per question, while the dataset contains up to twenty pages per question, during testing, inference is done using the full 20 pages of each document in the dataset. A batch size of 10 is used, and a learning rate of $2 \cdot 10^{-4}$ with a linear scheduler. For the multi-page layout-aware de-noising pretraining 3 million pages of the OCR-IDL dataset are used, and pretraining on the generated questions is one epoch over the 1.6 million questions.

3.2 MP-Pix2Struct

The multi-page Pix2Struct is an extension of the single-page Pix2Struct following the same approach as in MP-VT5. The main difference is that since Pix2Struct is a pixel-only model, the only necessary input to MP-Pix2Struct are the images of each page of the document. One change we make from the original Pix2Struct, besides the changes required for multi-page, is to reduce the number of patches in the original images by removing patches that are empty of content, this is explained in section 3.3. This change makes the training of this model much easier, as instead of using the original 4,096 patches per page, we are able to use only 1,024.

Like in MP-VT5 we use the base version of the model as a backbone, however, in this case it is larger than T5-base, with 282 million parameters, meaning that the multi-page version grows to 377 million parameters when the global-layers are added.

3.3 Reduction of document images through patch removal

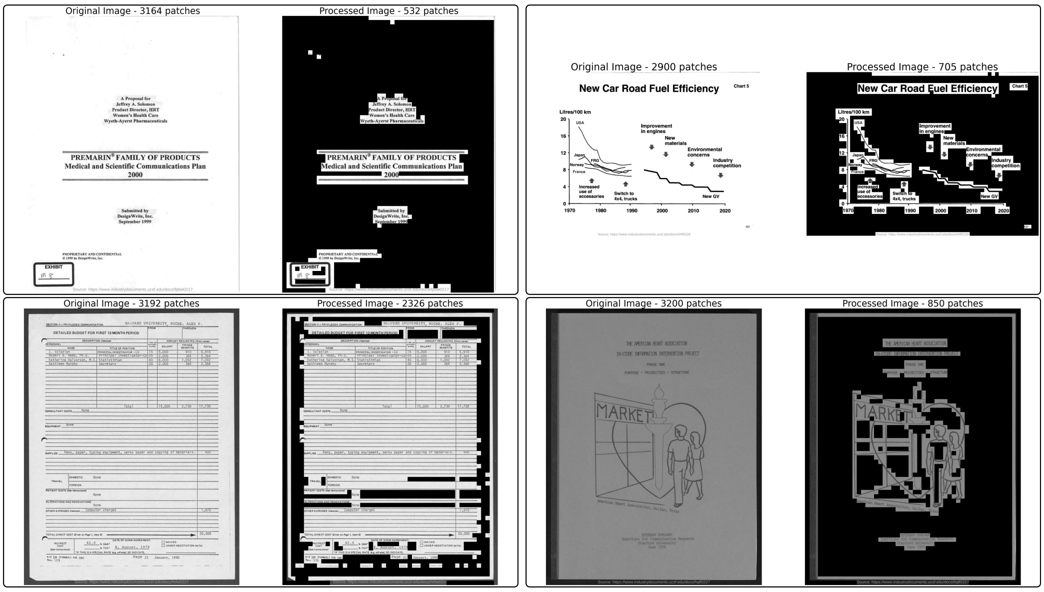


FIGURE 3.4: Examples of the patch removal method.

Following the simple observation that document images are mostly white or background space, it makes sense to remove patches that do not contain content. However, this can only be done if the model uses some sort of layout encoding for the patches, as the empty patches do provide information which is the distance that separates the other patches. Since both MP-VT5 and MP-Pix2Struct do add the positional information of the bounding boxes to each image patch, the model does not need the other patches to know where each patch belongs to.

The heuristic used to compute if a patch contains information or not is to check the standard deviation of its pixels, if it is low, it means that all pixels in the patch are of the same colour, and hence can be removed safely, while if a patch contains a letter or a line, some pixels will have very different values than others and hence the standard deviation will be high. We predefined a threshold to choose which patches are discarded and which are kept. See Figure 3.4 for examples of the patch removal, where it can be clearly seen that all information in the original document is still present in the document image with the removed patches.

Chapter 4

Results and Discussion

Comparisons with DocFormerV2 and GRAM are done using the results of their base size models, which have 257 million, and 286 million parameters respectively. At this size, the models have a parameter count in the same ballpark as our models MP-VT5 with 253 million and MP-Pix2Struct with 377 million parameters.

First we compare the performance of the single-page backbones, then we show results of the multi-page models. Finally, qualitative results of questions asked over this very same thesis are shown.

All results are shown using the Average Normalised Levenshtein Similarity (ANLS) [20], which is the standard metric use to evaluate VQA models. Unlike accuracy which would be zero for any difference between the model answer and the ground-truth answer, ANLS does not penalise as much small errors which are usually due to OCR mistakes rather than the model failing to produce the correct answer. The ANLS metric is defined as follows:

$$ANLS = \frac{1}{N}\sum_{i=0}^{N} s(a_i, o_i) \tag{4.1}$$

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

where a_i is the ground truth answer to the ith question and o_i is the model's answer. With $NL(a_i, o_i)$ being the Normalised Levenshtein distance between a_i and o_i. ANLS will be displayed as a percentage, and therefore will have values between 0 and 100.

The number of global tokens k to be used was determined on an ablation study, $k = 5$, $k = 10$, and $k = 20$ were tested. The obtained ANLS results were 52.88, 53.91, and 52.20. Since the highest results were obtained when using $k = 10$ we set the number of global tokens to 10 in all experiments, which is also the same number of global tokens used by Hi-VT5, while GRAM uses 32.

4.1 Results of the Single-Page models

Since we are using as a backbone single-page models, it is interesting to compare how these backbones perform. In Table 4.1 the three single-page models VT5, Pix2Struct, and DocFormerV2 are evaluated. The first results are when having to answer the question over only the page that contains the answer, that is, the single-page scenario they have been trained for. The second results are when all pages of the document are concatenated, this is rather inefficient but still manageable for 20 pages. DocFormerV2 only has published results for the large version of the model in the single-page scenario, and since the model is not public, we could not test the base model ourselves.

TABLE 4.1: Results of the single-page models that are used as backbones for the multi-page versions.

Model	Single-Page	Multi-Page Concat
VT5	69.48	50.05
Pix2Struct	68.81	49.64
DocFormerV2	-	69.67

There is a substantial loss in ANLS performance between the single-page and the multi-page concatenated results. Showing the need for models specifically designed for the multi-page scenario. Another point to note is that DocFormerV2 has much better results than the two other models, most likely due to having received a much more extensive pretraining. This means that GRAM that builds on top of DocFormerV2 has a much better starting point than our MP-VT5 and MP-Pix2Struct models.

4.2 Results of the Multi-Page models

The main results of this work are in Table 4.2, which shows the ANLS performance on MP-DocVQA for multi-page models of both the Text-Image-Layout modality and Pixel-Only modality. The table distinguishes between results obtained using some approach to sequence reduction to limit the number of tokens used by the decoder, and results obtained from using all tokens from all pages in the decoder, these lasts ones are marked in grey. Regarding Text-Image-Layout models, we compare our MP-VT5 (highlighted in bold), with Hi-VT5 and GRAM. For the Pixel-Only modality, we show results for our MP-Pix2Struct (highlighted in bold) and a Pix2Struct with a Self-Attention Scoring Mechanism.

Our MP-VT5 manages to obtain an ANLS of 60.57 on the MP-DocVQA dataset. While it is the lowest value compared to the state of the art models, it is close to both Hi-VT5 and the Pix2Struct with a Self-Attention Scoring Mechanism, while Amazon's model GRAM stays further up. It is not surprising that MP-VT5 is far from GRAM's score, in Table 4.1 which compares the results of the single-page backbones used in the multi-page models, DocFormerV2, from which GRAM builds on, achieves significantly better results than VT5 and Pix2Struct. GRAM benefits a lot by starting from this more extensively pretrained model, making it hard for MP-VT5 to compete. In regards to using all tokens from all pages in the decoder, all models obtain better results, however it must be noted that it comes at an extra computational cost. In this scenario MP-VT5, with 65.85 ANLS, manages to shorten the gap with GRAM, from a gap of 10.23% when performing sequence reduction to a gap of 7.80%. MP-Pix2struct results are far from the other pixel-only method when using global tokens, but surpasses it when using all tokens.

Table 4.3 contains the results of an ablation study that tests the benefits of the layout-aware span de-noising pretraining, the pretraining with generated questions, and the use of an answer page prediction module. The results on the final column are

TABLE 4.2: ANLS results on the MP-DocVQA dataset. The table compares our results with state of the art models, separating the Text-Image-Layout and Pixel-Only modalities. The approach used to reduce the number of decoder input tokens is specified, marking in grey the ones that use all tokens.

Modality	Model	Decoder Tokens	MP-DocVQA (ANLS)
Text-Image-Layout	Hi-VT5	Global Tokens	61.84
	GRAM	C-Former	70.80
		All	73.64
	MP-VT5 (ours)	Global Tokens	60.57
		All	65.84
Pixel-Only	Pix2Struct with Self-Attention Scoring Mechanism	One Page	61.99
	MP-Pix2Struct (ours)	Global Tokens	44.94
		All	64.95

always after finetuning the model on 10 epochs of MP-DocVQA. Therefore, if a row contains a check mark selecting a pretraining it means that first the model has been pretrained and then finetuned. If both pretrainings are selected, the de-noising pretraining is always done first, and the generated questions pretraining second, as this last one is closer to the actual task. If page prediction is selected, that means that answer page prediction is done during finetuning. The first row shows the results of only finetuning the model on MP-DocVQA, in the next two rows both pretrainings are compared, with both obtaining a higher ANLS than only finetuning. The pretrainig with generated questions obtains better results than the other one. Adding answer page prediction is shown to improve results. Finally by doing both pretrainings and answer page prediction the best results are obtained.

TABLE 4.3: Ablation study testing the benefits of the de-noising pretraining, the pretraining with generated questions, and the use of an answer page prediction module.

Model	De-noising Pretraining	Generated Questions Pretraining	Page Prediction	MP-DocVQA (ANLS)
MP-VT5	✗	✗	✗	54.13
	✓	✗	✗	55.01
	✗	✓	✗	57.24
	✗	✓	✓	58.68
	✓	✓	✓	**60.57**

There are no unexpected results in this ablation study, training the model for longer with more data improves results, even if the task is different, hence pretraining improves results. The pretraining on generated questions is closer to the MP-DocVQA task, and therefore, obtains better results than de-noising pretraing. Both pretrainings combined means even more data, and better results. Answer page prediction also improves results, the reason is that the model gets an extra signal during training, and having to explicitly learn in which page the answer to the question is found helps to answer the question correctly.

No results are shown for the MP-VT5 with the question in the global-layers since no good results were obtained. Training that version of the model is trickier, as it is a larger departure from the original VT5 which expects to find the question with the rest of the page tokens. Even with the modified layout-aware de-noising that contains a question, we did not manage to satisfactorily train the model. While the idea behind this pretraining makes sense, it also introduces a new concept which is having query locations in the question as special <loc_x> tokens with the actual location embedding summed to the token, learning this while also learning to take the question from the global-layers was too complex and hence why the training was not successful.

4.3 Qualitative results

Can our model answer questions about this thesis?

In order to show some qualitative results about the kind of questions our model can or can not answer correctly, we asked the MP-VT5 model the following questions over the entire thesis, which, since this section was of course not yet written, had 25 pages (including front page, abstract page and index page).

First we started with some easy questions from the first page:

- **Q:** What is the title of this document? **A:** Master's thesis (**Correct Answer:** Multi-Page Document Understanding)
- **Q:** Who is the author of this document? **A:** Artemis Llabrés
- **Q:** What is the date at the bottom of the first page? **A:** July 8, 2024
- **Q:** From which university is the author? **A:** University of Miami (**Correct Answer:** Universitat Autònoma de Barcelona)
- **Q:** What is this document about? **A:** Master's thesis
- **Q:** Who supervised this work? **A:** Dr. Dimosthenis Karatzas (**Correct Answer:** Dr. Dimosthenis Karatzas, Dr. Ernest Valveny)
- **Q:** From which degree is this master thesis? **A:** Masters Degree in Research and Innovation in Computer Based Science and Engineering

The model gets wrong the first question about the title of the document, the answer it provided "Master's thesis" appears above the title in the first page, so perhaps that is the source of the confusion. When asked about the supervisor of the work, the model only answers with "Dr. Dimosthenis Karatzas", missing Dr. Ernest Valveny. If the question is rephrased as follows we do get two supervisors, however, the names are close, but not correct:

- **Q:** Who are the two supervisors of this work? **A:** Diomethenia Karatzas and Ernest Valentenian (**Correct Answer:** Dr. Dimosthenis Karatzas, Dr. Ernest Valveny)

The most outrageous mistake is the university name, where the model completely invents the answer "University of Miami", probably recalling it from the train set where "Miami" appears in 13 answers, one of which as "Florida International University, Miami", and also once as "University of Miami". Trying to give clues in the question of where to find the university name did also not work:

- **Q:** What university is written in the first page of the document? **A:** University of Miami (**Correct Answer:** Universitat Autònoma de Barcelona)
- **Q:** What university is written at the top of the first page? **A:** Washington University (**Correct Answer:** Universitat Autònoma de Barcelona)

The likely reason for these failures is that the university name is written in catalan, we tried changing "university" by "universitat" in the question to see if it helped:

- **Q:** From which universitat is the author? **A:** AUSTNOMA DE BALCELONA (**Correct Answer:** Universitat Autònoma de Barcelona)
- **Q:** What universitat is written in the first page of the document? **A:** Universitat Universitat ad in Milan (**Correct Answer:** Universitat Autònoma de Barcelona)
- **Q:** What universitat is written at the top of the first page? **A:** Universitatat Autnoma de Barcelona (**Correct Answer:** Universitat Autònoma de Barcelona)

Now we get answers that are closer to being correct. Part of the issue here is that the tokenizer the model uses does not have the 'ò' character, and hence "autònoma" is tokenized as "aut<unk>noma".

The following are questions about Chapter 1, which the model mostly gets correct:

- **Q:** What is the title of the first chapter? **A:** Introduction
- **Q:** When was the DocVQA dataset created? **A:** 2023 (**Correct Answer:** 2020)
- **Q:** When was the MP-DocVQA dataset created? **A:** 2023
- **Q:** Which encoder-only models are mentioned? **A:** m4c (**Correct Answer:** M4C, LayoutLMv2)
- **Q:** What approach does DocFormerV2 and UDOP use to create the image embedding? **A:** global-local (**Correct Answer:** ViT)
- **Q:** What image encoder does Donut use? **A:** Win Transformer (**Correct Answer:** Swin Transformer)
- **Q:** Which figure shows the Hi-VT5 architecture? **A:** Figure 1.2
- **Q:** In GRAM, how is the Compression Transformer called? **A:** C-Forma (**Correct Answer:** C-Former)
- **Q:** What is the contribution of this work? **A:** The implementation of an open-source version of Amazon's multi-page gram model called MP-VT5 and the development of a new pixel-only multi-page model: MP-Pix2struct.

Questions about Chapter 2 and Chapter 3. In the first question the model only answers with one of the two datasets. The model also fails when asked about the number of parameters of the MP-VT5 model, answering instead with the parameters of T5.

- **Q:** What datasets are listed in the second chapter? **A:** OCR-IDL (**Correct Answer:** MP-DocVQA and OCR-IDL)
- **Q:** How many questions are in the MP-DocVQA dataset? **A:** 46,176
- **Q:** How many documents are in the OCR-IDL dataset? **A:** 4.6 million
- **Q:** What does MP-VT5 do to replace GRAM's C-Former? **A:** directly use the global tokens as a summarised representation of each page for the decoder
- **Q:** In which figure is the architecture for the MP-VT5 with the question in the global-layers shown? **A:** Figure 3.3
- **Q:** What is the total number of parameters of MP-VT5? **A:** 220 (**Correct Answer:** 253 million)
- **Q:** How many pages from the OCR-IDL dataset are used during pretraining? **A:** 3 (**Correct Answer:** 3 million)
- **Q:** How is it computed if a patch contains information or not? **A:** standard deviation of its pixels

Finally, questions about Chapter 4 and the bibliography. The model has some trouble with questions about the results presented in the table. The model also fails when asked to count the number of items in the bibliography, although, to be fair, counting questions are not present in the training dataset.

- **Q:** Which metric is used to evaluate the models? **A:** anls
- **Q:** What number of global tokens are used in the MP-VT5 model? **A:** 10
- **Q:** What results does the DocFormerV2 model obtain in the Multi-Page Concat scenario? **A:** 69.67
- **Q:** What is the highest ANLS shown in the results section? **A:** 60.57 (**Correct Answer:** 73.64)
- **Q:** What is the ANLS of MP-VT5 when using all tokens? **A:** 60.57 (**Correct Answer:** 65.84)
- **Q:** Which model obtains an ANLS of 61.84? **A:** global tokens (**Correct Answer:** Hi-VT5)
- **Q:** How many items are cited in the bibliography? **A:** 17 (**Correct Answer:** 20)
- **Q:** Who is the first author of the 8th citation in the bibliography? **A:** Arthur J. Darby (**Correct Answer:** A. F. Biten)

The model completely fails to answer the last question, answering with the made up name "Arthur J. Darby", but from where does the model get that name? The name "J. Darby" appears 159 in the MP-DocVQA training dataset answers, but that is a "William J. Darby". The name "Arthur" appears 12 times, with one "Arthur J. Rubel". So it seems that the model recalled two names from its training set to answer this question.

Chapter 5

Final Thoughts

5.1 Conclusion

We succeeded in our goal of implementing an open-source version of Amazon's GRAM model, our model MP-VT5, while a little bit far from GRAM's results, shows good results on the task of Multi-Page Document Visual Question Answering, and is close to other methods like Hi-VT5, and Pix2Struct with a Self-Attention Scoring Mechanism. While we did not manage to obtain results with our more extensive modification of the architecture which consisted of having the question in the global-layers instead of copied in each local-layer, we did successfully add an answer page prediction module that helped improve results. Our pixel-only multi-page model, MP-Pix2Struct did not obtain good results when using the global tokens in the decoder, however, it did show promising results when using all tokens.

Overall, with this work we did achieved our main objective which was learning about multi-page models for document understanding by implementing state of the art approaches. The knowledge gained and the models produced will serve as a foundation for us to continue this line of research.

5.2 Open Research Directions

While the first steps of further research after this thesis are fixing the MP-Pix2Struct when using global tokens in the decoder and the MP-VT5 with the question in the global layers. New research lines that open from here include: to more extensively pretrain our models, leveraging to the fullest datasets like OCR-IDL, from which at most we utilised 3 million pages of the 26 million it contains. And to work on extending docvqa models to output layout coordinates in a similar approach to the way we do answer page prediction, which not only would allow the models to point to locations of the document where the information used to provide the answer has been found, but also by outputting coordinates these models would be able to perform other tasks like object detection, table extraction or layout analysis.

5.3 Acknowledgements

I would like to end by acknowledging and thanking my supervisors Dimosthenis Karatzas and Ernest Valveny, their mentorship and guidance has been invaluable to the completion of this work. I would also like to thank all my colleagues at the Computer Vision Center, with special thanks to Pau, Adri, and Carlos.

Bibliography

[1] G. Kim, T. Hong, M. Yim, J. Nam, J. Park, J. Yim, W. Hwang, S. Yun, D. Han, and S. Park, "OCR-Free Document Understanding Transformer," in *Computer Vision – ECCV 2022* (S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, eds.), (Cham), pp. 498–517, Springer Nature Switzerland, 2022.

[2] K. Lee, M. Joshi, I. R. Turc, H. Hu, F. Liu, J. M. Eisenschlos, U. Khandelwal, P. Shaw, M.-W. Chang, and K. Toutanova, "Pix2Struct: Screenshot Parsing as Pretraining for Visual Language Understanding," in *Proceedings of the 40th International Conference on Machine Learning*, pp. 18893–18912, PMLR, July 2023.

[3] R. Powalski, Ł. Borchmann, D. Jurkiewicz, T. Dwojak, M. Pietruszka, and G. Pałka, "Going Full-TILT Boogie on Document Understanding with Text-Image-Layout Transformer," in *Document Analysis and Recognition – ICDAR 2021* (J. Lladós, D. Lopresti, and S. Uchida, eds.), (Cham), pp. 732–747, Springer International Publishing, 2021.

[4] S. Appalaraju, P. Tang, Q. Dong, N. Sankaran, Y. Zhou, and R. Manmatha, "DocFormerv2: Local Features for Document Understanding," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 709–718, Mar. 2024.

[5] Z. Tang, Z. Yang, G. Wang, Y. Fang, Y. Liu, C. Zhu, M. Zeng, C. Zhang, and M. Bansal, "Unifying Vision, Text, and Layout for Universal Document Processing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19254–19264, 2023.

[6] M. Mathew, D. Karatzas, and C. V. Jawahar, "DocVQA: A Dataset for VQA on Document Images," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2200–2209, 2021.

[7] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, "VQA: Visual Question Answering," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2425–2433, 2015.

[8] A. F. Biten, R. Tito, A. Mafla, L. Gomez, M. Rusinol, E. Valveny, C. V. Jawahar, and D. Karatzas, "Scene Text Visual Question Answering," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4291–4301, 2019.

[9] G. Team, "Gemini: A Family of Highly Capable Multimodal Models," June 2024.

[10] L. Kang, R. Tito, E. Valveny, and D. Karatzas, "Multi-Page Document Visual Question Answering using Self-Attention Scoring Mechanism," Apr. 2024.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. ukasz Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (J. Burstein, C. Doran, and T. Solorio, eds.), (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.

[13] R. Hu, A. Singh, T. Darrell, and M. Rohrbach, "Iterative Answer Prediction With Pointer-Augmented Multimodal Transformers for TextVQA," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9992–10002, 2020.

[14] Y. Xu, Y. Xu, T. Lv, L. Cui, F. Wei, G. Wang, Y. Lu, D. Florencio, C. Zhang, W. Che, M. Zhang, and L. Zhou, "LayoutLMv2: Multi-modal Pre-training for Visually-rich Document Understanding," *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021.

[15] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[16] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," June 2021.

[17] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021.

[18] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension," *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, p. 7871, 2020.

[19] R. Tito, D. Karatzas, and E. Valveny, "Hierarchical multimodal transformers for Multipage DocVQA," *Pattern Recognition*, vol. 144, p. 109834, Dec. 2023.

[20] A. F. Biten, R. Tito, L. Gomez, E. Valveny, and D. Karatzas, "OCR-IDL: OCR Annotations for Industry Document Library Dataset," in *Computer Vision – ECCV 2022 Workshops* (L. Karlinsky, T. Michaeli, and K. Nishino, eds.), (Cham), pp. 241–252, Springer Nature Switzerland, 2023.