GRAM: Global Reasoning for Multi-Page VQA

Tsachi Blau* Technion, Israel
Sharon Fogel AWS AI Labs
Roi Ronen* Technion, Israel
Alona Golts† AWS AI Labs
Shahar Tsiper AWS AI Labs
Elad Ben Avraham AWS AI Labs
Aviad Aberdam AWS AI Labs
Roy Ganz* Technion, Israel
Ron Litman‡ AWS AI Labs

Abstract

The increasing use of transformer-based large language models brings forward the challenge of processing long sequences. In document visual question answering (DocVQA), leading methods focus on the single-page setting, while documents can span hundreds of pages. We present GRAM, a method that seamlessly extends pre-trained single-page models to the multi-page setting, without requiring computationally-heavy pretraining. To do so, we leverage a single-page encoder for local page-level understanding, and enhance it with document-level designated layers and learnable tokens, facilitating the flow of information across pages for global reasoning. To enforce our model to utilize the newly introduced document tokens, we propose a tailored bias adaptation method. For additional computational savings during decoding, we introduce an optional compression stage using our compression-transformer(C-Former),reducing the encoded sequence length, thereby allowing a tradeoff between quality and latency. Extensive experiments showcase GRAM's state-of-the-art performance on the benchmarks for multi-page DocVQA, demonstrating the effectiveness of our approach.

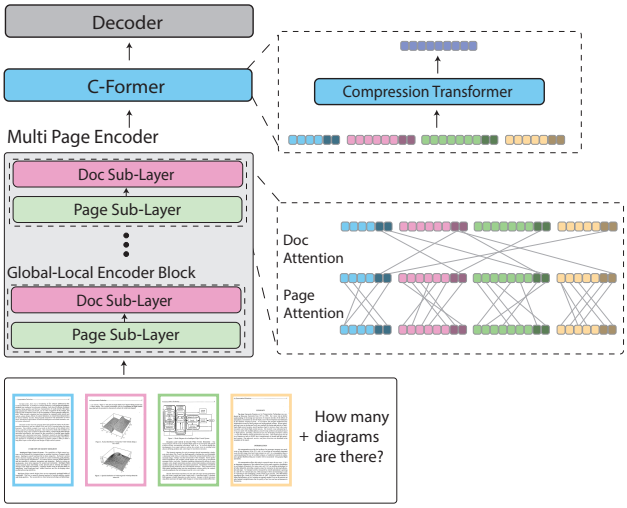

Figure 1. **An Overview of GRAM.** We suggest an interleaved encoder architecture combining page- with document-attention layers, allowing information to propagate between different pages. An optional compression transformer (C-former) is introduced to allow a trade-off between quality and latency.

1. Introduction

Document understanding, particularly in the context of DocVQA, has gained substantial research interest [5, 6, 16, 25, 36, 37] and offers a wide array of practical applications, focusing on data extraction and analysis of single page documents. However, Multi-Page DocVQA (MP-DocVQA) poses a more realistic challenge, considering that the majority of documents, including contracts, manuals and scientific papers, often extend well beyond a single page. Despite the practical relevance of MPDocVQA, it has received limited attention, primarily due to the absence of suitable datasets. Two recently introduced datasets, MP-DocVQA [33] and DUDE [18], have opened up new avenues for MP-DocVQA research.

Recent DocVQA approaches rely on transformers [35], at the heart of their architecture. While transformers are a powerful tool, they face challenges when dealing with long input sequences [4, 7, 10–12, 27, 38]. This difficulty stems from the self-attention mechanism, which scales quadratically in terms of computation and memory, with respect to the input sequence length. Addressing this limitation has attracted significant research efforts, primarily in the field

*Work conducted during an internship at Amazon.
†Corresponding author: alongolt@amazon.com
‡Corresponding author: litmanr@amazon.com

of natural language processing (NLP). Proposed NLP-based solutions can be divided into two main directions: The former aims to modify the attention mechanism to cut computational costs [4, 7, 38]. The latter involves altering the positional embedding mechanism to improve performance on longer sequences, with minimal fine-tuning [11, 12, 27].

A possible option of tackling MPDocVQA is to extend NLP-based approaches to handle multi-modal document data, including visual representations, along with OCR text and corresponding 2D locations and relative page position. However, this requires extensive pre-training, with relatively scarce multi-page document data, and thus is also sub-optimal in terms of performance. Instead, we opt for leveraging powerful single-page DocVQA models, especially pretrained on millions of single-page documents, and finetuning them to the multi-page scenario. For this purpose, we combine concepts of local (page) and global (document) tokens, which promote an exchange of information within and across pages, while keeping computational cost in check. We choose pages as atomic units in our proposed scheme, as page structure often represent a semantic unit in DocVQA.

We present GRAM (**G**lobal **R**e**A**soning for **M**ulti-page VQA), a novel approach for endowing multi-page processing capabilities to existing single-page DocVQA models. Alongside page tokens that encapsulate both textual and visual contents of each page, we introduce doc(ument) learnable tokens, which aim is dispersing global information across all pages. These two sets of tokens interact within our newly-devised two-stage encoder blocks. The initial stage utilizes an existing single-page layer and enhances it by including both page and doc tokens as input, allowing them to freely interact. In the second stage, we prioritize computational efficiency by restricting self-attention solely to the global doc tokens. This global reasoning layer captures collective information from multiple pages, enabling the system to respond to cross-page inquiries, as illustrated in Fig. 1. Considering that doc tokens did not appear in pretraining, to boost their significance during finetune, we employ a designated bias adaptation mechanism which strikes a balance between local and global learnable tokens.

While our method inherently deals with long sequences, we circumvent a quadratic reliance on sequence length by segmenting the document into pages — its semantically logical parts. We restrict interaction solely among doc learnable tokens, across all pages, thereby mitigating the computational burden of depending quadratically on the page count. Apart from encoding, the auto-regressive decoding stage poses a computational burden in long sequences. To this end, we introduce a compression stage that precedes the decoder, implemented with a compression transformer, termed **CFormer**. The CFormer receives the concatenated output of all pages and compresses it to a much shorter sequence, distilling the most pertinent information in the document. Our key contributions are:

- We propose GRAM, an approach to endow single-page DocVQA methods with multi-page capabilities, without pretraining, allowing the model to process multi-page documents, while preserving single-page performance.
- We introduce document learnable tokens and bias adaptation that enable an effective communication and collaboration between individual pages to support reasoning over multiple page documents.
- Our C-Former module suggests a trade-off between accuracy and compute, distilling information from multi-page sequences into more compact representations.
- We obtain SOTA results over the MPDocVQA and DUDE datasets, and provide extensive ablations to each component in our method.

2. Related Work

Long Sequence Approaches are an active field of research in NLP, aiming to improve the design of chat-systems [24] and image instruction tasks [19, 21]. In these applications, the ability to manage and process long sequences is vital, as conversations cannot be cut short, or limited to just a few interactions. Common approaches to tackle long sequences include sparse attention mechanisms [4, 7, 38] and methods to improve results on long sequences during inference [11, 27, 28]. 'Sliding window' approaches of limiting the range of neighbors each token can attend to, lead to a significant reduction in computation and memory consumption. Prominent works of this kind include LongFormer [7], where each token attends to a set of its nearest neighbors, along with additional global tokens. The work of Big-Bird [38] adds additional non-neighboring tokens at random, whereas Colt5 [4] uses the same sliding window approach, but performs heavier computations for important tokens and shallow operations for filler words or punctuation. Although Tito *et. al.* [33] have demonstrated that the above approaches do not perform as well on the task of MPDocVQA, we do incorporate the ideas of combining both local and global tokens throughout encoding to expand the attention onto additional pages.

DocVQA has attracted increasing attention [5, 6, 9, 16, 25, 26, 31, 36, 37] with the introduction of the DocVQA dataset by Mathew et al. [22]. Most methods in DocVQA leverage OCR [1–3, 20, 23] to input both text and layout information (bounding box coordinates and possibly font type) into the model, where some further explore different techniques to combine the two types of data streams, or alternatively, clever schemes of pretraining. DocVQA methods can be roughly divided to two categories: extractive and abstractive. Extractive methods [16, 25, 36, 37] rely on the fact that the explicit answer resides in the written text, thus only output a corresponding text span within the input

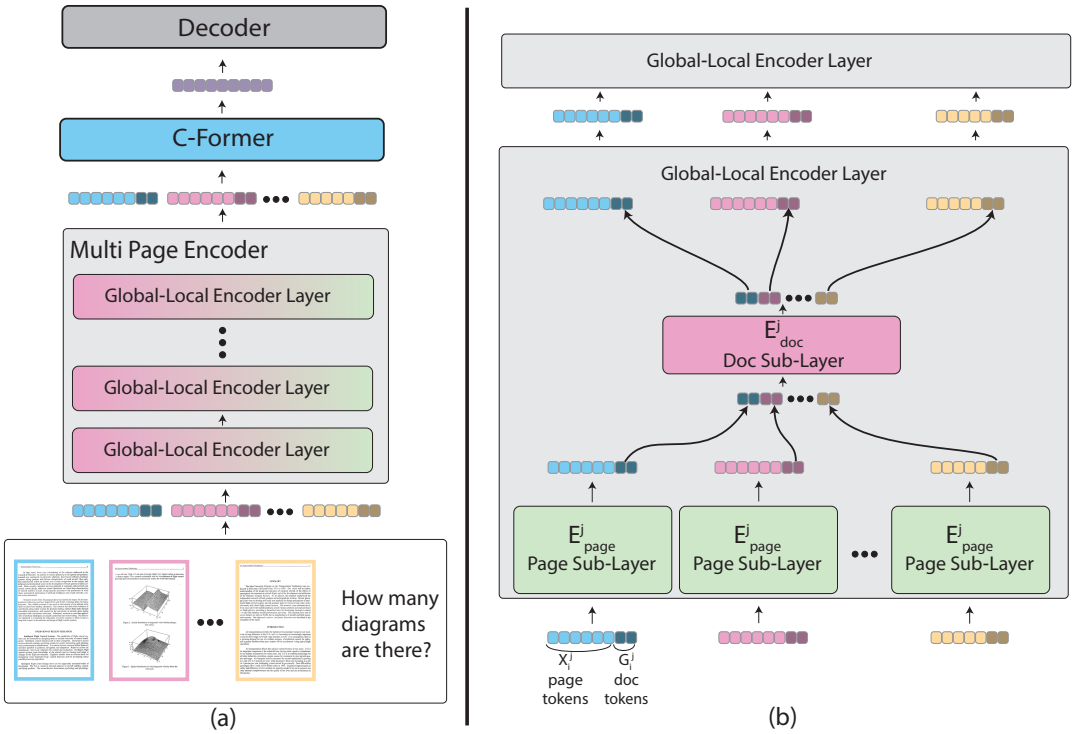

Figure 2. **GRAM Architecture. (a)** Depicts a high-level architecture overview. For each page, the visual, textual and question tokens are concatenated together with learnable doc tokens (darker color shade). The processed information is fed into the multi-page encoder. The encoder output can be fed directly into the decoder to create the final prediction. Optionally, a compression model, C-Former , can be used between the encoder and the decoder to compress the encoder output into a predetermined length, thus reducing overall latency for long documents. **(b)** Shows a global-local encoder layer, containing two sub-layers. The first sub-layer uses self-attention that operates on each page separately, while the second applies a self-attention step on the doc tokens to fuse information between the different pages. The corresponding tokens are then routed back to their respective page and go into the next global-local encoder layer.

sequence. Abstractive methods [5, 6, 14, 26, 31], on the other hand, have the capacity to generate free-form answers which do not necessarily appear in the text, thus providing flexibility in real-world applications. Notably, existing research in DocVQA does not scale in a straightforward way to deal with the more realistic multi-page scenario.

MPDocVQA has recently gained momentum with the launch of two new multi-page datasets: MP-DocVQA [33] and DUDE [18], offering two separate recipes to tackle longer documents. The first approach of Tito *et. al.*, referred to as HiVT5 [33], suggests compressing the encoding of each page separately, and feeding the decoder with the concatenation of the compressed outputs from each single-page encoder. While this approach is advantageous in terms of computation, we later show the compression may severely hinder the results. In addition, there is no communication between the single-page encoders until the final stage of decoding, whereas in our method, we allow for exchange of page and document information throughout all stages of encoding. Another prominent approach, proposed by Landeghem *et. al.* [18], which relatively preserves quality, involves concatenating all the pages into one long sequence and feeding it to a standard encoder-decoder structure. This, however, poses a heavy computational burden as transformers' self-attention component scales quadratically with input sequence length.

3. GRAM

3.1. Base Architecture

The underlying idea in our approach is using existing encoder-decoder single-page models for document understanding and extending them to multi-page scenarios, without additional pretraining. In this work, we provide such a recipe over the notable DocFormerv2 [6]. For the sake of completeness, DocFormerv2 is a T5-based [28] encoder-decoder transformer model which operates over both visual and textual features to support document understanding. Each page is represented by textual features $\mathbf{T} \in R^{N_t \times d}$ which encapsulate OCR tokens and their corresponding 2D bounding box positions [36], along with visual $\mathbf{V} \in R^{N_v \times d}$ and question $\mathbf{Q} \in R^{N_q \times d}$ embeddings. Where N_t, N_v and

N_q are the lengths of the OCR, visual features and the question. The output result $\mathbf{Y}$ is obtained by passing the concatenated inputs through the encoder-decoder model,

$$\mathbf{Y} = \mathbf{D}(\mathbf{E}(\text{concat}(\mathbf{T}, \mathbf{V}, \mathbf{Q}))), \tag{1}$$

where $\mathbf{E}$ and $\mathbf{D}$, are the encoder and decoder, respectively.

Our method uses these basic building blocks in designing a multi-page solution. To this end, we introduce a bi-level global-local encoder, as illustrated in Fig. 2. At the local page-level of each block, we utilize the layers of the existing single-page encoder $\mathbf{E}$ to process each page separately, together with learnable doc tokens. Next, we introduce a slim global layer in each block that facilitates communication between doc tokens across all pages. This bi-level localized processing ensures the model can understand the content of each page effectively, while also combining information across pages in the document. After M such blocks, we feed the encoded features from all pages into the existing decoder $\mathbf{D}$ to produce the overall output.

3.2. Global-Local Reasoning

To operate on multiple pages we break down the document to K pages, and the single-page encoder to M encoder layers, $\mathbf{E}^j, j = 0, ..., M-1$. We then construct M blocks, with two sub-layers each. The first page sub-layer originates from the existing pretrained encoder layer, referred to as $\mathbf{E}^j_{page}$, and operates in parallel, with shared weights, for all pages in the document. This layer receives both page and doc tokens. The second, newly introduced, document layer $\mathbf{E}^j_{doc}$ collects only the doc tokens from all pages and promotes sharing information across all of the document.

Formally, we augment the input of the standard single-page encoder with page-specific indexing $(\mathbf{T}_i, \mathbf{V}_i, \mathbf{Q})$ and incorporate page-positional embedding $\mathbf{P}_i$ to both text and visual features, where $i = 0, ..., K-1$:

$$\tilde{\mathbf{T}}_i = \mathbf{T}_i + \mathbf{P}_i, \quad \tilde{\mathbf{V}}_i = \mathbf{V}_i + \mathbf{P}_i, . \tag{2}$$

Next, we formulate our bi-level global-local block. The input to the first page-level sub-layer in each block is the concatenation of the textual, visual and question features, denoted $\mathbf{X}^j_i = \text{concat}(\tilde{\mathbf{T}}_i, \tilde{\mathbf{V}}_i, \mathbf{Q}_i)$, along with page-specific doc tokens $\mathbf{G}^j_i \in R^{N_g \times d}$,

$$\mathbf{X}^{j+1}_i, \tilde{\mathbf{G}}^{j+1}_i = \mathbf{E}^j_{page}(\text{concat}(\mathbf{X}^j_i, \mathbf{G}^j_i)). \tag{3}$$

Here, the features undergo self-attention, normalization and feed-forward layers. The layer output $\mathbf{X}^{j+1}_i$ is passed on as input to the next bi-level block, whereas only the doc tokens $\tilde{\mathbf{G}}^{j+1}_i$, enter the second doc sub-layer, which again includes self-attention, normalization and feed-forward

$$\{\mathbf{G}^{j+1}_i\}^{K-1}_{i=0} = \mathbf{E}^j_{doc}(\text{concat}(\{\tilde{\mathbf{G}}^{j+1}_i\}^{K-1}_{i=0})). \tag{4}$$

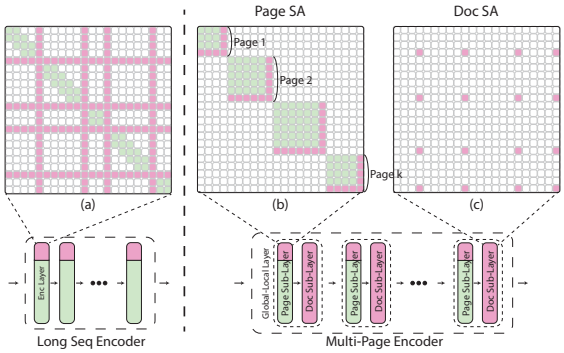

Figure 3. **Global-Local Attention**: In long sequence approaches (a), attention is applied jointly to the entire sequence of concatenated local and global tokens. Our method, separates the computation into two steps — page-level (b) and document-level (c)— leveraging the natural division of documents into pages.

In this stage, the doc tokens can interact and pass information from page to page, after which being passed on to the next block, as depicted in Eq. (3). This design allows information to flow between pages while keeping computational costs in check. When concluding the traversal over M such layers, the outputs across all pages are concatenated and fed to the decoder $\mathbf{D}$,

$$\mathbf{Y}_{multi} = \mathbf{D}(\text{concat}(\{\mathbf{X}^M_i, \mathbf{G}^M_i\}^{K-1}_{i=0})). \tag{5}$$

To visualize the difference between the attention masks in our method, we compare it with previous long sequence approaches [4, 7, 38] in Fig. 3. These prior methods optimize computation by using attention masking on nearby tokens and allowing limited global connections. However, naively applying such methods to multi-page documents will treat it as a single stream, which does not consider the division into pages. Our global-local blocks, with a two-stage attention-masking mechanism, better suit multi-page documents. In addition, our two-level design benefits from existing, extensively pretrained single page models.

3.3. Bias Adaptation

An already-pretrained model, introduced with a new stream of data, might disregard it altogether [13, 14, 34, 39]. To overcome this, we force the system to account for the newly-introduced doc tokens by modifying the encoder's bias method. Originally, the bias method intervenes in the attention mechanism, diminishing the relationships between distant tokens. However, in our specific case, the distance between doc and page tokens does not represent their actual relevance. To enforce the encoder to pay closer attention to the doc tokens, we assign them a positive constant bias value. Particularly, we replace the values in the bias matrix, corresponding with the doc tokens, with fixed ones. Instead of a single bias value, we utilize a different value

for each attention head, as performed in ALiBi [27], enabling more fine-grained control of the global features in each head. Specifically, the constant doc bias value is set to $c \cdot \frac{1}{2^a}$, where c is a constant and a is the attention head index. This yields a decaying bias value across different attention heads, resulting in hierarchical importance of the document information, where the first heads are more oriented towards doc tokens and the last towards page tokens.

3.4. Compression Transformer

Our global-local solution to MP-DocVQA resolves the problematic quadratic dependency on the number of pages K during encoding. However, the auto-regressive decoding complexity scaling linearly with K also poses a practical challenge during inference time, as we later discuss in Sec. 3.5. To alleviate this burden, we place an optional transformer-based model, named C-Former (**C**ompression Trans**Former**), between the encoder outputs and the decoder, as depicted in Fig. 2. The C-Former has the ability to revise the information across all pages and distill only the important details, required to correctly answer the question.

Specifically, the C-Former is a light-weight transformer-based decoder [28], denoted as $\mathbf{D}_C$, featuring cross-attention, layer norm and feed-forward layers in each block. The input to C-Former includes N_c learnable tokens $\mathbf{C} \in R^{N_c \times d}$, concatenated with the input question $\tilde{\mathbf{C}} = \text{concat}(\mathbf{C}, \mathbf{Q})$. In addition, we feed it with the outputs of the global-local interlaced encoder, concatenated to one long sequence, referred as $\mathbf{O}$, where $\mathbf{O} = \text{concat}(\{\mathbf{X}_i^M, \mathbf{G}_i^M\}_{i=0}^{K-1})$. The output of C-Former is thus

$$\mathbf{O}_C = \mathbf{D}_C(Q{=}\tilde{\mathbf{C}}, K{=}\mathbf{O}, V{=}\mathbf{O}),$$

where we pass forward only the first set of N_c output embeddings and ignore the rest, setting the output sequence dimension to N_c. C-Former offers flexibility in controlling the tradeoff between ANLS quality and computational efficiency by controlling the output sequence length N_c.

3.5. Computation Analysis

Next, we turn to provide a thorough computational complexity analysis. We consider a document that comprises of K pages, each with N tokens, and the maximum answer length is L. For simplicity, we assume that all encoders and decoders have one layer. The naïve way to support multi-page documents is using an existing single-page encoder-decoder model, fusing all of the textual page inputs together, and feeding them as one long sequence. We refer to this approach as 'concat'. The self-attention complexity of such a configuration scales quadratically with the sequence length, $O((N \cdot K)^2)$. Conversely in our method, we operate on the document pages with two alternating encoding stages in each layer. The first stage performs a self-attention over both the page and doc tokens. Hence, the complexity of such sub-layer is $O((N + N_g)^2 \cdot K)$, where N_g is the number of doc tokens. The second stage features a self-attention operation over the doc tokens, across all pages in the document. The complexity of this operation is $O((N_g \cdot K)^2)$. Overall, the total complexity for one global-local encoder block is $O((N + N_g)^2 \cdot K + (N_g \cdot K)^2)$. Since N_g is a constant, and the number of pages is usually less than the number of words in each page ($K < N$), we obtain a complexity of $O(N^2 \cdot K)$, which is not quadratic in K.

Prior to decoding, the outputs of all per-page encoders are concatenated, thus the output sequence length is $(N + N_g) \cdot K$. Since the decoder is auto-regressive, its complexity depends quadratically on the maximum output length, L, namely, $O((N + N_g) \cdot K \cdot L^2) = O(N \cdot K \cdot L^2)$. Since this operation of decoding is performed iteratively during inference, the combined sequence length $(N + N_g) \cdot K$ becomes computationally heavy. To alleviate this concern, we propose an optional C-Former model, which performs compression prior to decoding. The overall complexity in this decoding scheme includes passing through the C-Former and then through the decoder, leading to $O((N + N_g) \cdot K \cdot N_c) + N_c \cdot L^2)$ which is equivalent to $O(N \cdot K + L^2)$, since N_c is a constant, denoting the number of compression tokens in C-Former.

4. Experiments

4.1. Experimental Settings

Datasets and Metrics The MPDocVQA dataset [33] features 46K questions, spanning over 48K images, and includes layout elements as figures, tables, lists and diagrams, with printed, handwritten and typewritten text. MP-DocVQA contains mostly extractive questions, for which answers are present in the given text. DUDE is smaller in size (23.7K questions over 3K documents), but offers complex questions that require a reader to rationalize beyond the written text content. We report our results using the ANLS metric, introduced in [8], computing a generalized accuracy. Results for DUDE can be broken apart to several types of questions, categorized to four groups: 'extractive' – for which the answer is found directly in the text; 'abstractive' – requiring a free-form answer that does not necessarily appear in the document; 'list of answers' – requiring a list of answers, as opposed to a single one, and 'unanswerable' – where the result cannot be determined using the text.

Implementation Details Our underlying architecture is based on Docformerv2 [6]. Recall, our interlaced encoder features M blocks (12 in 'base' and 24 in 'large'), where each block contains a page sub-layer which originates from an extension of Docformerv2's encoder layer. Every structure contains self-attention, normalization and feed-forward

Method	Params	MPDocVQA	DUDE				
		ANLS	ANLS	ANLS per Question Type			
				Extractive	Abstractive	List of answers	Unanswerable
Longformer [7]	$148M$	55.06	27.14	43.58	8.55	10.62	10.78
BigBird [38]	$131M$	58.54	26.27	40.26	7.11	8.46	12.75
LayoutLMv3 [38]	$125M$	55.13	20.31	32.60	8.10	7.82	8.82
Hi-VT5$^{\dagger}_{beamsearch}$ [15]	$316M$	–	35.74	28.31	32.98	10.60	62.90
Hi-VT5[33]	$316M$	62.01	23.06	17.60	33.94	6.83	61.67
Hi-VT5*	$257M$	60.78	23.86	7.21	16.56	3.53	72.77
DocFormerv2$_{concat}$ [6]	$257M$	69.67	44.21	41.66	41.86	15.13	**65.19**
GRAM$_{C-Former}$	$286M$	70.80	40.07	40.43	39.61	11.42	52.55
GRAM	$281M$	**73.68**	**46.15**	**46.07**	**44.82**	**15.27**	62.18
T5-2D [18]	$770M$	–	46.06	**55.65**	**50.81**	5.43	**68.62**
DocGptVQA [30]	$> 3.5B$	–	50.02	51.86	48.32	28.22	62.04
DocBlipVQA [29]	$> 3.5B$	–	47.62	50.69	46.31	**30.73**	55.22
Hi-VT5* [33]	$784M$	71.35	28.89	18.21	26.17	6.84	58.99
DocFormerv2$_{concat}$ [6]	$784M$	76.40	48.44	50.82	48.06	17.67	59.04
GRAM$_{C-Former}$	$864M$	77.60	45.47	47.63	44.91	14.34	56.99
GRAM	$859M$	**80.32**	**51.15**	53.67	50.35	18.40	63.23
Hi-VT5*† [33]	$784M$	73.51	49.18	49.29	48.35	13.30	**65.95**
DocFormerv2$^{\dagger}_{concat}$ [6]	$784M$	76.77	50.79	52.70	49.61	17.33	65.14
GRAM$^{\dagger}_{C-Former}$	$864M$	78.12	50.97	55.15	50.46	17.26	61.04
GRAM†	$859M$	**79.67**	**53.36**	**56.83**	**52.32**	**19.96**	65.43

Table 1. **Quantitative Results**. We present ANLS results for the MPDocVQA [33] and DUDE [18] test sets. The methods are grouped according to the model type and size, starting from encoder-only models (top), T5-base models (middle) and T5-large models (bottom). † denotes training with both MPDocVQA and DUDE.

Method	Training Data		ANLS	
	DocVQA	MPDocVQA	DocVQA	MPDocVQA
DocFormerv2$_{concat}$	✓	✗	86.60	72.73
	✗	✓	85.28	76.40
	✓	✓	86.47	75.37
GRAM	✓	✗	86.70	73.12
	✗	✓	85.29	80.32
	✓	✓	86.32	78.66

Table 2. **DocVQA vs. MPDocVQA Performance.** Test results over both datasets using the large model variants. A checkmark denotes whether a dataset was included in training or not.

layers. We extend the page layer from Docformerv2 to feature also the doc learnable embeddings. The second doc sub-layer is similar in structure to the first sub-layer, only it is initialized from scratch, with the following specification: $d_{ff} = 1024$, $d_{kv} = 64$, $nheads = 4$, $d = 256$. We implement 32 doc learnable tokens for each page, uniformly initialized to random values. For bias adaptation, the initial bias value is set to $c = 20$, with variations between encoder heads, as described in Sec. 3.3. We incorporate an additional optional compression stage using C-Former – a randomly-initialized T5 [28] tiny decoder, with an encoder mask instead of a causal one. The output sequence extracted from C-Former is $N_c = 256$. Finally the decoder is initialized with pretrained weights from DocformerV2.

The model is trained with the Hugging Face Trainer [17] for $200k$ steps, starting with a warm-up of $1k$ steps, with linear learning rate decay. We use learning rates of $3e^{-5}$ and $1e^{-4}$ for the already pretrained encoder and decoder weights, versus the newly initialized doc sub-layer weights. Training is performed on a cluster of $8 \times A100$ GPUs, each with $40GB$ of RAM. During training, each page encoder receives 800 tokens, dealing with up to 4 pages. During testing, we increase the maximum length of tokens to $8,000$.

Baselines We report the results of previous work on both MP-DocVQA and DUDE datasets (if those exist), including the NLP-based Longformer [7] and BigBird [38], which were adapted to MPDocVQA by [33]; LayoutLMv3 [16], originally designed for DocVQA; and Hi-VT5 [33] and T5-2D [32], specifically suggested for MP-DocVQA Task. We also add for reference the results of methods published in the leader-boards of MPDocVQA and DUDE, which do not have corresponding papers, including DocGptVQA [30], DocBlipVQA [30], and Hi-VT5$_{beamsearch}$ [15] ([15] was trained on both MP-DocVQA and DUDE). In our approach, we present two variations: GRAM and GRAM$_{C-Former}$. While GRAM utilizes the full length of the encoder output,

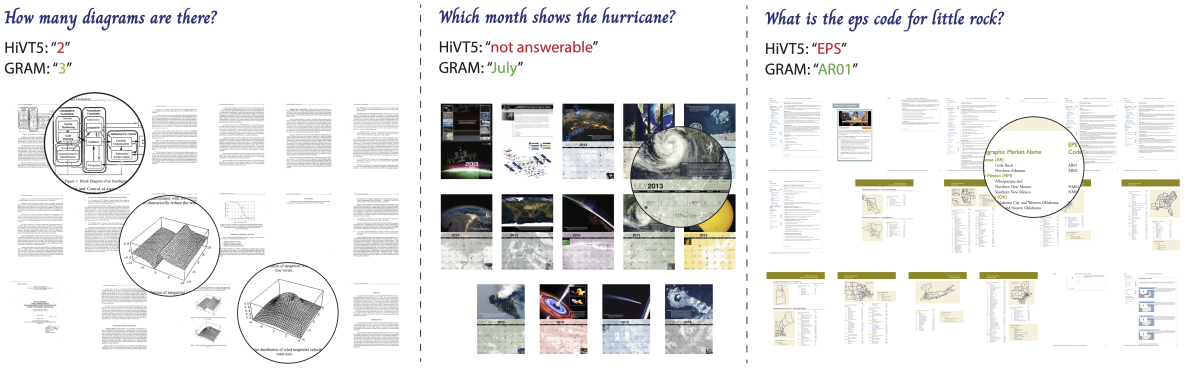

Figure 4. Qualitative comparison between our approach and Hi-VT5 [33] indicate that the integration of our global-local encoder enhances reasoning capabilities, especially when the inquiries require multi-page context.

GRAM$_{C-Former}$ allows the user to control the trade-off between performance and latency.

To ensure a fair comparison, since we use the pretrained model of DocFormerv2 [6], we implement two additional baselines, referred to as Hi-VT5* and DocFormerv2$_{concat}$. The first follows a similar structure as Hi-VT5 [33], with the encoder originating from DocFormerv2, however without the page answer prediction, as it does not exist in DUDE. The second recreates the approach of [18], where only the textual tokens of all pages are concatenated to one long sequence, then passing through the DocFormerv2 model. The second approach poses a computational burden, thus we use only 600 tokens during training per page, with up to 4 pages, and during test only 400 tokens.

4.2. Results

We present the performance of our method over the MP-DocVQA [33] and DUDE [18] datasets in Tab. 1. The methods are divided into three groups: the top contains encoder-only, and methods that rely on the T5-base model (up to $316M$ parameters); the middle section, approaches that use the T5-large model (over $770M$ parameters), and finally the bottom, T5-large models, trained on both datasets.

As can be seen, in the first group, the encoder-only NLP methods, LongFormer [7], BigBird [38] and LayoutLMv3 [16] can only handle relatively well 'extractive' style tasks as in MPDocVQA dataset [33], but often struggle with 'abstractive' questions that are more abundant in DUDE [18]. As to T5-'base' models, versus our best competitor Docformerv2$_{concat}$, we obtain an improvement of $(+4\%, +1.9\%)$ on MP-DocVQA and DUDE datasets. As to methods that combine an additional compression before decoding (Hi-VT5, Hi-VT5*), our C-Former achieves an increase in $(+8.8\%, +16.2\%)$ over the best candidates on the MP-DocVQA and DUDE datasets.

As for the group of 'large' models, we include the results of T5-2D [18] DocGptVQA [30] and DocBlipVQA [29]. Note that our model surpasses DocFormer$_{concat}$, the primary baseline, achieving improvements of $(+3.9\%, +2.7\%)$ on MP-DocVQA and DUDE, respectively. We also outperform DocGptVQA [30], a method that appears in the leaderboard of DUDE, by $+1.1\%$, thereby obtaining SOTA results for GRAM 'large'.

The final category showcases large encoder-decoder models, fine-tuned on both MP-DocVQA and DUDE training sets, showcasing the benefits of augmented training data. GRAM consistently demonstrates performance gains over the baseline, illustrating its robustness across different datasets and training scenarios. Next, we present in Tab. 2 the effect of training on DocVQA vs. MPDocVQA . Our method achieves performance on-par on the single page task, while enhancing performance on the multi-page scenario by $+3.3\%$, compared to the baseline.

In Fig. 4, We show qualitative results on the DUDE dataset of GRAM versus Hi-VT5* [33]. Our method demonstrates proficiency in addressing questions that involve attention over multiple pages (*'how many diagrams are there'*), an increased visual analysis capability (*'Which month shows the hurricane?'*), and heightened abstractive ability (*'What is the EPS code for Little Rock?'*).

5. Ablation Study

We perform an ablation study on our approach, evaluating the influence of each constituent component using DUDE's validation set [18]. This validation set enables the grouping of documents by their respective page counts: 1, 2–4, 5–10, 11–end, encompassing $1747, 2259, 1062, 1241$ samples in each category, respectively. Our investigation delves into the impact of the number of doc tokens and the bias adaptation methods. Moreover, we employ the C-Former for sequence compression, adjusting the compression ratio and examining the balance between performance and latency (see supplementary for more details).

#Doc Tokens	Bias Type	Compression Dimension	ANLS by Number of Pages DUDE validation dataset				
			All	1	2-4	5-10	11-end
✗	✗	✗	46.16	47.18	48.66	43.34	42.57
16		✗	46.39	48.35	49.06	43.16	41.56
32	Decaying	✗	**47.88**	**49.29**	**49.90**	**45.90**	**43.94**
64		✗	46.70	47.98	49.22	44.00	42.60
	✗	✗	47.52	**49.85**	**49.93**	44.90	42.10
32	Constant	✗	46.14	47.41	48.13	44.44	42.19
	Decaying	✗	**47.88**	49.29	49.90	**45.90**	**43.94**
		8	39.83	39.73	41.41	36.98	39.52
		32	40.42	40.95	41.64	38.12	39.39
32	Decaying	256	41.99	42.57	43.77	38.40	41.01
		1024	42.56	42.97	44.30	38.94	41.93
		4096	**43.59**	**44.75**	**44.64**	**40.54**	**42.67**

Table 3. **GRAM Ablation Study**. Results on DUDE validation set ablating over (a) the dimension of doc tokens, (b) the attention bias employed and (c) the C-former input dimension.

GRAM Components We focus our initial exploration on the impact of the number of doc tokens N_g. As can be seen in Tab. 3, while $N_g = 16$ leads to performance on-par with not using doc tokens at all, for the optimal value of $N_g = 32$, we obtain an increase of $+1.7\%$ in ANLS. Shifting our focus to bias adaptation methods, Tab. 3 shows that using constant bias has a negative effect on the results, suggesting this method is not flexible enough in maintaining a balance between the page and doc tokens. However, our decaying bias-adaptation approach does improve results overall, versus no-bias $(+0.36\%)$, especially for longer documents ($+1\%$ improvement for 5-10 pages and $+1.84\%$ for 11 pages and more). This is to be expected, since incorporating new doc tokens and increasing their importance can potentially affect single-page performance. Finally, in Tab. 4, we reinforce our choice of pages as semantic logical units for MPDocVQA. We first ablate our method with and without page embedding. Next, we compare our page-based division with varying fixed-length division of tokens for encoder. Results in Tab. 4 clearly demonstrate an advantage towards page-level encoding in MPDocVQA. This aligns with our initial assumption that structured documents are often designed with page-division in mind.

Performance-Latency Trade-off We assess the impact of C-Former on performance, considering compression output lengths of $8, 32, 256, 1024, 4096$. Note, performance gradually improves with an increase in the compression output length. However, longer output lengths correspond to heightened model latency. Note that using C-Former for shorter documents can be redundant, as there is little to no compression compared to the input sequence length and results decrease. In Fig. 5, we scrutinize the trade-off between computational efficiency and compression rate

Page Embedding	Segment Length	ANLS by Number of Pages DUDE validation dataset				
		All	1	2-4	5-10	11-end
✓	✗	**47.88**	**49.29**	**49.90**	**45.90**	**43.94**
✗	✗	46.12	48.74	48.11	43.59	40.99
✓	256	45.22	46.38	46.69	44.13	41.83
✓	512	45.09	45.90	47.32	42.65	41.98
✓	1024	44.39	44.98	46.63	41.69	41.78

Table 4. **The Significance of Pages as Semantic Units**. Results on DUDE validation set ablating over (a) utilization of page-embedding, (b) segment length for fixed-size encoding inputs.

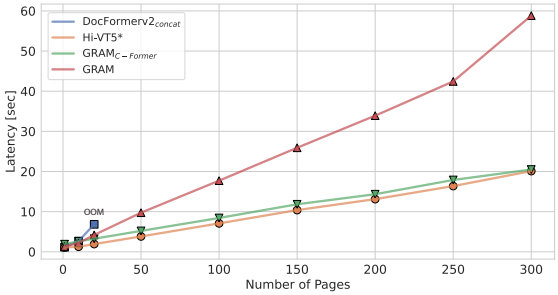

Figure 5. **Latency comparison**. We compare the dependency between overall latency and the number of pages in input document for GRAM, GRAM$_{C-Former}$, DocFormerv2$_{concat}$ and Hi-VT5.

by comparing to DocFormerv2$_{concat}$ [6] and Hi-VT5* [33]. We discover that DocFormerv2$_{concat}$ reaches a memory limit at approximately 20 pages, due to its quadratic memory increase with sequence length. At this juncture, GRAM$_{C-Former}$ surpasses DocFormerv2$_{concat}$ by performing 3.5 seconds faster. Notably, GRAM$_{C-Former}$ can gracefully handle documents surpassing 300 pages, effectively bridging the gap between performance and latency.

6. Conclusions

Our method, termed GRAM, extends existing single-page document models to efficiently handle multi-page documents without necessitating computationally-intensive pre-training. Leveraging the single-page encoder for local page-level comprehension, we introduce document learnable tokens and designated layers, enabling seamless information exchange across pages. Additionally, our proposed bias adaptation method enforces effective utilization of our newly introduced document tokens. The incorporation of a C-Former model reduces sequence length, balancing quality with latency in the decoding step. Extensive experiments demonstrate GRAM's state-of-the-art performance across multi-page DocVQA benchmarks.

References

[1] Aviad Aberdam, Ron Litman, Shahar Tsiper, Oron Anschel, Ron Slossberg, Shai Mazor, R Manmatha, and Pietro Perona. Sequence-to-sequence contrastive learning for text recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15302–15312, 2021. 2

[2] Aviad Aberdam, Roy Ganz, Shai Mazor, and Ron Litman. Multimodal semi-supervised learning for text recognition. *arXiv preprint arXiv:2205.03873*, 2022.

[3] Aviad Aberdam, David Bensaïd, Alona Golts, Roy Ganz, Oren Nuriel, Royee Tichauer, Shai Mazor, and Ron Litman. Clipter: Looking at the bigger picture in scene text recognition. *arXiv preprint arXiv:2301.07464*, 2023. 2

[4] Joshua Ainslie, Tao Lei, Michiel de Jong, Santiago Ontañón, Siddhartha Brahma, Yury Zemlyanskiy, David Uthus, Mandy Guo, James Lee-Thorp, Yi Tay, et al. Colt5: Faster long-range transformers with conditional computation. *arXiv preprint arXiv:2303.09752*, 2023. 1, 2, 4

[5] Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R Manmatha. Docformer: End-to-end transformer for document understanding. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 993–1003, 2021. 1, 2, 3

[6] Srikar Appalaraju, Peng Tang, Qi Dong, Nishant Sankaran, Yichu Zhou, and R Manmatha. Docformerv2: Local features for document understanding. *arXiv preprint arXiv:2306.01733*, 2023. 1, 2, 3, 5, 6, 7, 8

[7] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020. 1, 2, 4, 6, 7

[8] Ali Furkan Biten, Ruben Tito, Andres Mafla, Lluis Gomez, Marçal Rusinol, Ernest Valveny, CV Jawahar, and Dimosthenis Karatzas. Scene text visual question answering. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4291–4301, 2019. 5

[9] Ali Furkan Biten, Ron Litman, Yusheng Xie, Srikar Appalaraju, and R Manmatha. Latr: Layout-aware transformer for scene-text vqa. pages 16548–16558, 2022. 2

[10] Aydar Bulatov, Yuri Kuratov, and Mikhail S Burtsev. Scaling transformer to 1m tokens and beyond with rmt. *arXiv preprint arXiv:2304.11062*, 2023. 1

[11] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595*, 2023. 2

[12] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019. 1, 2

[13] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999. 4

[14] Roy Ganz, Oren Nuriel, Aviad Aberdam, Yair Kittenplon, Shai Mazor, and Ron Litman. Towards models that can see and read. *arXiv preprint arXiv:2301.07389*, 2023. 3, 4

[15] JiangLong He, Mamatha N, Shiv Vignesh, and Deepak Kumar. Hivt5beam. Hi-VT5 model pretrained with private custom document collection using span masking objective. Pretrained model is then trained with DUDE dataset and Multi-Page DocVQA dataset, 2023. 6

[16] Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. Layoutlmv3: Pre-training for document ai with unified text and image masking. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4083–4091, 2022. 1, 2, 6, 7

[17] HuggingFace. Huggingface trainer. https://huggingface.co/docs/transformers/training. 6

[18] Jordy Landeghem, Rubén Tito, Łukasz Borchmann, Michał Pietruszka, Paweł Józiak, Rafał Powalski, Dawid Jurkiewicz, Mickaël Coustaty, Bertrand Ackaert, Ernest Valveny, et al. Document understanding dataset and evaluation (dude). *arXiv preprint arXiv:2305.08455*, 2023. 1, 3, 6, 7

[19] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023. 2

[20] Ron Litman, Oron Anschel, Shahar Tsiper, Roee Litman, Shai Mazor, and R Manmatha. Scatter: selective context attentional scene text recognizer. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11962–11972, 2020. 2

[21] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023. 2

[22] Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209, 2021. 2

[23] Oren Nuriel, Sharon Fogel, and Ron Litman. Textadain: Paying attention to shortcut learning in text recognizers. In *European Conference on Computer Vision*, pages 427–445. Springer, 2022. 2

[24] R OpenAI. Gpt-4 technical report. *arXiv*, pages 2303–08774, 2023. 2

[25] Qiming Peng, Yinxu Pan, Wenjin Wang, Bin Luo, Zhenyu Zhang, Zhengjie Huang, Teng Hu, Weichong Yin, Yongfeng Chen, Yin Zhang, et al. Ernie-layout: Layout knowledge enhanced pre-training for visually-rich document understanding. *arXiv preprint arXiv:2210.06155*, 2022. 1, 2

[26] Rafał Powalski, Łukasz Borchmann, Dawid Jurkiewicz, Tomasz Dwojak, Michał Pietruszka, and Gabriela Pałka. Going full-tilt boogie on document understanding with text-image-layout transformer. In *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part II 16*, pages 732–747. Springer, 2021. 2, 3

[27] Ofir Press, Noah A Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021. 1, 2, 5

[28] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and

Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020. 2, 3, 5, 6

[29] RenZhou, QiaolingDeng, XinfengChang, LuyanWang, XiaochenHu, HuiLi, and YaqiangWu. Docblipvqa. We integrated the prediction outputs from the UDOP model and Blip2 to enhance our results,and we optimized the image encoder and included page number features to address the challenge of multi-page documents. GPT to generate python-like modular programs., 2023. 6, 7

[30] RenZhou, QiaolingDeng, XinfengChang, LuyanWang, XiaochenHu, HuiLi, and YaqiangWu. Docgptvqa. We integrated the prediction outputs from the UDOP model and Blip2 to enhance our results,and we optimized the image encoder and included page number features to address the challenge of multi-page documents. GPT to generate python-like modular programs., 2023. 6, 7

[31] Zineng Tang, Ziyi Yang, Guoxin Wang, Yuwei Fang, Yang Liu, Chenguang Zhu, Michael Zeng, Cha Zhang, and Mohit Bansal. Unifying vision, text, and layout for universal document processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19254–19264, 2023. 2, 3

[32] Rubèn Tito, Dimosthenis Karatzas, and Ernest Valveny. Document collection visual question answering. In *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part II 16*, pages 778–792. Springer, 2021. 6

[33] Rubèn Tito, Dimosthenis Karatzas, and Ernest Valveny. Hierarchical multimodal transformers for multi-page docvqa. *arXiv preprint arXiv:2212.05935*, 2022. 1, 2, 3, 5, 6, 7, 8

[34] Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212, 2021. 4

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1

[36] Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 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*, pages 1192–1200, 2020. 1, 2, 3

[37] Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. *arXiv preprint arXiv:2012.14740*, 2020. 1, 2

[38] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297, 2020. 1, 2, 4, 6, 7

[39] Xiaohua Zhai, Xiao Wang, Basil Mustafa, Andreas Steiner, Daniel Keysers, Alexander Kolesnikov, and Lucas Beyer. Lit: Zero-shot transfer with locked-image text tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18123–18133, 2022. 4