Efficient Architectures for High Resolution Vision-Language Models

Miguel Carvalho **Bruno Martins**
INESC-ID and Instituto Superior Técnico, University of Lisbon
{miguelcarvalho00,bruno.g.martins}@tecnico.ulisboa.pt

Abstract

Vision-Language Models (VLMs) have recently experienced significant advancements. However, challenges persist in the accurate recognition of fine details within high resolution images, which limits performance in multiple tasks. This work introduces Pheye, a novel architecture that efficiently processes high-resolution images while training fewer parameters than similarly sized VLMs. Notably, Pheye achieves a high efficiency while maintaining strong performance, particularly in tasks that demand fine-grained image understanding and/or the handling of scene-text.

1 Introduction

The integration of visual capabilities as extensions to Large Language Models (LLMs) has led to the emergence of large Vision-Language Models (VLMs), currently excelling in tasks like image captioning and visual question answering. Notable examples include Flamingo (Alayrac et al., 2022) or Monkey (Li et al., 2023d), in this last case bringing improvements to tasks that involve scene-text by processing higher-resolution images. LLaVA-NeXT (Liu et al., 2024) followed Monkey's lead with similar enhancements, but at the cost of a quadratic increase in the computational complexity of the language model, as it skipped a resampler module that compresses large sequence lengths, from the encoder, into a fixed size of query vectors.

Recent focus has shifted towards smaller VLMs that can run on hardware-constrained devices. Models like MoE-LLaVA (Lin et al., 2024) or moondream2 (Vikhyat, 2024) achieve impressive performance with a fraction of the parameters of their predecessors. This work further advances small VLMs with Pheye, i.e. a family of compact models that can process high-resolution images with fewer parameters and computational demands, expanding VLM applications to resource-limited environments where understanding fine details is crucial.

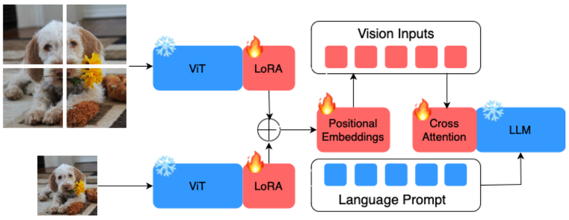

Figure 1: Overview on the proposed architecture, where input images are split into regular non-overlapping patches that match the input resolution of a pre-trained ViT. Two sets of LoRA adapters are respectively used to adjust the ViT to both global and local sub-images, and a frozen LLM is conditioned on the concatenated vision representations through dense cross-attention layers.

Specifically, Pheye employs a frozen instruction-tuned language model (Li et al., 2023c) in conjunction with a frozen pre-trained CLIP (Radford et al., 2021) vision model, which are linked by dense cross-attention layers inserted before the language model's layers. To process high-resolution images, we use two sets of LoRA (Hu et al., 2021) adapters in the vision encoder, one for the global image and another for local high-resolution patches.

Notably, Pheye is competitive with similarly sized models, particularly in tasks involving scene-text, such as TextVQA (Singh et al., 2019). Pheye requires only a fraction of the training parameters, being more efficient in connecting the vision and language modalities, and in processing input images at higher resolutions. The source code associated to the experiments reported on this paper, as well as the trained models, are available from a public GitHub repository[1].

2 An Efficient High-Resolution VLM

This section presents the proposed method for building high-resolution efficient VLMs, starting with architectural design choices, and then analysing the model's computational complexity.

[1] https://github.com/miguelscarv/pheye

Proceedings of the 31st International Conference on Computational Linguistics, pages 10520–10530
January 19–24, 2025. ©2025 Association for Computational Linguistics

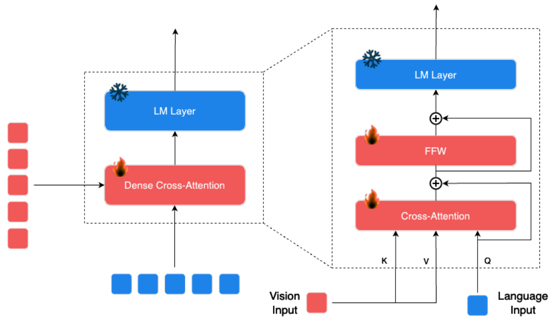

Figure 2: An illustration for dense cross-attention layers. To condition the language model on visual inputs, we add new cross-attention layers between existing pre-trained and frozen language model layers. The keys and values for these layers are derived from vision features, while the queries come from language inputs. These layers are followed by dense feed-forward layers. The output matrices of both of these modules are initialized with values close to zero to maintain the integrity of the language model at initialization.

2.1 The Proposed Architecture

The Pheye architecture is illustrated in Figure 1. It employs a frozen instruction-tuned language model and a vision encoder that adapts a pre-trained CLIP model, linking the two components with dense cross-attention layers inserted before the language model's layers. The use of cross-attention and the design of the vision encoder were informed by preliminary experiments described in Appendix B.

The vision encoder consists of a ViT with two sets of LoRA adapters, one for encoding global images and another for local high-resolution patches, as outlined in Appendix B.2. To better distinguish between global and local patch embeddings and improve convergence, we introduce two LayerNorm (Ba et al., 2016) layers after the ViT, and before adding learned positional embeddings. These layers are applied separately, respectively processing the global and local patch embeddings.

Given the computational efficiency, strong task performance, and reduced number of trainable parameters, we use cross-attention to combine modalities while keeping the language model frozen. Inspired by the Flamingo architecture (Alayrac et al., 2022), we replace vanilla cross-attention modules with dense cross-attention modules, inserting them at regular intervals before the decoder layers, as shown in Figure 2. This design was motivated Flamingo's demonstration that gated cross-attention outperforms vanilla cross-attention even when parameter counts are equal. However, we deviate from Flamingo's gated cross-attention layers, as early experiments showed that the gating mechanism hindered convergence. To preserve the language model's initial integrity, we initialize the output matrices of the dense cross-attention modules with values sampled from a normal distribution with a mean of zero and a variance approaching zero. This way, the cross-attention layers have a minor influence during the initial training epochs.

2.2 Analysis of the Computational Complexity

This section analyzes the computational complexity of the proposed methods, without accounting for the LoRA adapters in the vision encoder, by calculating the number of multiplications that are involved in all linear layers and attention operations, separately considering the vision encoder and the language model with dense cross-attention layers. The analysis assumes a sequential order of operations in all linear layers and attention mechanisms, although implementation optimizations can be latter used. We compare our method with the most widely used approach of using a higher-resolution ViT and a LLaVA-style architecture.

For reference, the cost of a matrix multiplication C between matrices $N \in \mathbb{R}^{n\times d}$ and $W \in \mathbb{R}^{d\times o}$ would be $C = (n \times o) \times d$, where N represents a matrix of n rows with dimensionality d, and W represents a weight matrix with an input dimensionality of d and an output dimensionality of o.

Vision Encoder. The computational complexity of a ViT depends on the number of input tokens N, corresponding to the number of image patches plus one for the [CLS] token, and the model dimensionality D. The number of multiplications for a single Transformer layer can be expressed as:

$$\mathbf{T}_{\text{ViT}} = 4ND^2 + DN^2 + 8ND^2. \tag{1}$$

This complexity breaks down into three components, namely (i) the multiplications in the W_Q, W_K, W_V and W_O matrices, (ii) the attention mechanism, and (iii) a feed-forward module with two layers, where the intermediate dimensionality is four times the model dimensionality.

In the case of our strategy, instead of computing self-attention across all the high-resolution input tokens, the image is broken down into P sub-images of equal size and lower-resolution, and the number of multiplications can be expressed by the following equation, where N' is now reflecting the number of patches per sub-image, plus one [CLS] token also per sub-image (i.e., each sub-image involves a total of $N' = \frac{(N-1)}{P-1} + 1$ tokens).

$$\mathbf{T}_{\text{Pheye}} = (4N'D^2 + DN'^2 + 8N'D^2) \times P. \tag{2}$$

We can compare the efficiency of our vision encoder against a ViT that operates at a resolution of 672×672 pixels with a patch size of 14 pixels, resulting in $N = 2305$ input tokens. Our method would process 10 images, where 1 is global and 9 are local sub-images, at a resolution of 224×224 pixels with a patch size of 14, yielding $N' = 2570$ tokens. With $D = 1280$, our method is approximately 1.02 as efficient as the alternative. While this corresponds to little to no improvement, our method does offer a significant advantage: it eliminates the need to fine-tune the underlying ViT, allowing us to focus solely on training the LoRA parameters to increase the input resolution.

Language Model. For the language model, we analyze the computational complexity of a LLaVA-style architecture, against that of our method. The complexity of the LLaVA style architecture per layer is given by Equation 3 and our method has an average complexity per layer given by Equation 4, where the fractional term refers to the average complexity introduced by dense cross-attention layers. Specifically, the aforementioned term has 4 components in the numerator: (i) the multiplications in the W_K and W_V matrices, (ii) the multiplications in W_Q and W_O, (iii) the attention mechanism, and (iv) the feed-forward module. In both formulas, N_T represents the number of text tokens, N_I represents the number of image tokens, D and D_{ViT} correspond to the dimensionality of the language model and the ViT, respectively, and I denotes the interval at which dense cross-attention layers are inserted in the language model.

$$\begin{aligned}\mathbf{T}_{\text{LLaVA}} =& 4(N_T + N_I)D^2 + D(N_T + N_I)^2 + \\ & 8(N_T + N_I)D^2.\end{aligned} \tag{3}$$

$$\begin{aligned}\mathbf{T}_{\text{Pheye}} =& 4N_T D^2 + DN_T^2 + 8N_T D^2 + \\ & \frac{2N_I D_{ViT} D + 2N_T D^2 + DN_T N_I + 8N_T D^2}{I}.\end{aligned} \tag{4}$$

For assessing the efficiency gain of our language model against a LLaVA-style architecture, assume $N_T = 65$, which is the typical prompt length for captioning, $N_I = 2305$, corresponding to the number of vision tokens output by a ViT with an input resolution of 672x672 pixels, $I = 2$, $D = 2048$ and $D_{ViT} = 1280$. Our method is, in this case, approximately 12.1 times more efficient than its LLaVA-style counterpart. Furthermore, if we increase the interval to $I = 4$, our method becomes approximately 18.5 times more efficient than the corresponding LLaVA-style architecture.

3 Main Experimental Evaluation

This section begins by outlining the Pheye training setup. It then presents the results achieved by our models on academic task-oriented datasets.

3.1 Experimental Setup

Our experiments aimed to assess the effects of increasing the input image resolution, and to examine the impact of augmenting the frequency of dense cross-attention layers in the language model. To do this, we compare four model settings, varying the image input resolution between 448×448 pixels and 672×672 pixels, and adjusting the dense cross-attention interval to every 4 or 2 decoder layers.

In all four settings, we used a Phi 1.5 language model (Li et al., 2023c), finetunned on the SlimOrca (Lian et al., 2023) instruction dataset. The ViT is initialized from a CLIP-ViT-H-14 model finetunned on DFN-5B (Fang et al., 2023).

The models were trained in separate stages, similarly to MoE-LLaVA (Lin et al., 2024), using a cross-entropy loss over the output tokens. The different stages are described next, while Appendix A summarizes the datasets used in Stage III.

• **Stage I.** We initially aimed for a model that can effectively describe images, including their finer details. We used ShareGPT4V-PT (Chen et al., 2023a), featuring 1,246K images with detailed descriptions, of which 570K are high resolution images from SAM (Kirillov et al., 2023). To reduce overfitting to a particular prompt, ten different captioning instructions were manually generated. One of these instructions was then randomly selected to be associated with each image-description pair.

• **Stage II.** The second stage empowers Pheye to go beyond captioning, using a mixture of complex multi-modal instruction following examples: MIMIC-IT (Li et al., 2023a), LRV (Liu et al., 2023), SViT (Zhao et al., 2023) and LVIS (Wang et al., 2023). This comes to a total of 964K samples. For each sample with multiple instruction-response turns, a single turn was randomly selected.

• **Stage III.** Previous stages used synthetic GPT-4 (Achiam et al., 2023) and GPT-4V (OpenAI, 2023) data for finetunning, which is naturally prone to noise and hallucinations. To alleviate this, we further finetunned our models on a mixture of academic task-oriented VQA and captioning datasets, as well as some synthetic multimodal instruction following examples, based on LLaVA 1.5-mix-665K. We specifically considered ST-VQA

	Model Size			Evaluation Results				
Model	Res.	Trained	Data	VQAv2	NoCaps	TextVQA	TextCaps	DOCCI
BLIP-2	224	187M	129M	63.0	104.5	43.1*	-	-
InstructBLIP-2	224	187M	130M	-	119.9	46.6*	82.4†	5.7†
MobileVLM 1.7B	336	1.4B	3.9M	-	-	41.5	-	-
MobileVLM V2 1.7B	336	1.4B	6.3M	-	90.0†	52.1*	48.5†	4.5†
MoE-LLaVA-1.8B×4	336	2.8B	6.6M	76.2	-	48.0*	-	-
MoE-LLaVA-2.7B×4	336	5.0B	6.6M	77.6	-	51.4*	-	-
moondream1	384	1.86B	3.9M	74.7	-	35.6	-	-
moondream2	384	1.86B	-	77.7	92.5†	49.7	120.2†	0.2†
Pheye-x4	448	295M	2.9M	75.2	110.3	45.9	106.4	5.7
Pheye-x4	672	295M	2.9M	75.5	110.8	49.2	111.9	5.4
Pheye-x2	448	578M	2.9M	76.0	111.8	47.3	108.9	5.6
Pheye-x2	672	578M	2.9M	76.4	110.5	50.5	115.9	5.9
Pheye-x2 *(upscaled low-res inputs)*	672	578M	2.9M	76.3	112.0	44.3	105.7	5.7

Table 1: Results on different academic task-oriented datasets. "Res.", "Trained" and "Data" represent the input image resolution, the number of trainable parameters, and the number of training instruction-response pairs, respectively. Both BLIP-2 (Li et al., 2023b) and InstructBLIP-2 (Dai et al., 2024) refer to the FlanT5$_{XL}$ (Chung et al., 2024) variants, while moondream2 refers to the 2024-04-02 model version. The symbol * denotes evaluations made with OCR tokens in the instruction, while † refers to our own evaluation of the models, using the prompt `"Provide a one-sentence caption for the provided image"`, or instead the prompt `"Generate a highly detailed description for the provided image using multiple sentences"` for the case of DOCCI (Onoe et al., 2024), without OCR tokens in the instruction. Pheye-xN denotes a Pheye model with dense cross-attention layers inserted every N layers of the Phi 1.5 model. VQAv2 refers to the test-dev split, NoCaps (Agrawal et al., 2019) and TextVQA refer to the validation splits, and TextCaps and DOCCI refer to the test splits of the corresponding datasets. VQAv2 and TextVQA report VQA accuracy, while NoCaps, TextCaps, and DOCCI report CIDEr.

(Biten et al., 2019), TextVQA, and the LLaVAR finetunning dataset for better scene-text performance, removed the RefCOCO (Kazemzadeh et al., 2014) and VisualGenome (Krishna et al., 2017) datasets, and sampled 2 random turns from the VQAv2 (Antol et al., 2015) and GQA (Hudson and Manning, 2019) examples for each image. As a result, the model is only trained on 22.4% of the VQAv2 and GQA examples in the original LLaVA 1.5 mix. We also sampled single turns from LLaVA, LLaVAR, OCR-VQA (Mishra et al., 2019) and OKVQA (Marino et al., 2019). From A-OKVQA (Schwenk et al., 2022), we randomly sampled unique image-question pairs.

We trained the dense cross-attention layers, LoRA, positional embeddings, and LayerNorm layers, at every stage, using mixed precision Bfloat16. The frozen parameters, which correspond to the ViT and the language model, were loaded in Bfloat16. Resolutions were also kept constant across stages for the same model. The first stage uses a learning rate of 2e-4, the second stage uses 1e-4, and the third stage uses 5e-5. All stages use a cosine decay learning rate scheduler. We used gradient accumulation with an effective batch size of 128 across stages as well.

Since gradient accumulation for varying sequence lengths implicitly gives more weight to tokens in smaller sequences, we used sum reduction for the loss, instead of the typical mean reduction, tracking the number of tokens used to calculate the loss for a full batch. Before each optimization step, we divide the gradients by the number of output tokens, thereby mimicking a batch size of 128.

3.2 Experimental Results

Table 1 summarizes our experimental results. With less training data and less trainable parameters, Pheye surpasses other generalist models of similar size in scence-text tasks, without using pre-extracted OCR tokens in the textual instructions. Note, for instance, that MobileVLM V2 (Chu et al., 2024) exhibits subpar performance on TextCaps, which suggests that the model relies heavily on the presence of OCR tokens in the instruction.

The performance on scene-text tasks also increases more with a higher image resolution, compared to an increase in parameter count. The opposite happens for more general image understanding tasks, like VQAv2 and NoCaps. Increasing the resolution allows the model to capture finer details in images, while increasing the parameter count allows the model to learn more visual concepts, like objects and relationships between objects.

		VQAv2			TextVQA		
Model	Resolution	Bottom	Middle	Top	Bottom	Middle	Top
Pheye-x4	448	73.55	75.19	76.09	40.67	44.23	52.74
Pheye-x4	672	74.26	75.17	75.99	46.83	48.43	52.48
%Δ		+0.96	-0.03	-0.13	+15.15	+9.50	-0.49
Pheye-x2	448	74.50	75.47	76.67	42.19	45.59	54.18
Pheye-x2	672	74.73	76.25	76.71	47.31	50.67	53.34
%Δ		+0.31	+1.03	+0.05	+12.14	+11.14	-1.55

Table 2: Relative change in VQA accuracy for VQAv2 and TextVQA instances, according to data tertiles that reflect the relative dimensions of relevant image areas.

The last row of Table 1 presents a test in which the inputs to our best model are first down-scaled to 224x224, this way resulting in images without fine details. A higher performance drop is again seen on tasks such as TextVQA, confirming the importance of the high image resolution.

3.3 Assessing the Use of Fine Image Details

In order to further evaluate the impact that increasing the input image resolution has on tasks that involve the comprehension of fine-grained details, we followed a strategy similar to that of Zhang et al. (2023a) and partitioned the TextVQA validation set into three groups of approximately equal size, based on the relative size of the ground-truth bounding box $S = \frac{A_{bb}}{A_{total}}$, where A_{bb} denotes the area of the answer bounding box, and A_{total} denotes the total area of the image. Specifically, we divided the data into three tertiles: the bottom tertile (finer details), the middle tertile (medium), and the top tertile (broader entities). The selection of the ground-truth bounding box was based on the average string similarity with all the ground truth answers, using the longest contiguous matching subsequence algorithm.

We also applied the same approach to a randomly sampled subset of the VQAv2 validation split, resulting in 10,000 questions pertaining to 8,453 images. For this dataset we used the segmentation area of objects as opposed to bounding boxes, since this quantity can better represent an object's area in the image, and used its category name for the string similarity algorithm. Since a large portion of VQAv2 answers correspond to yes/no or a number, in these cases we applied the same algorithm to the question, instead of the ground truth answers.

Table 2 presents the results of our analysis. In both datasets, increasing resolution seems to lead to a greater improvement in accuracy for the samples that require understanding finer details within the image. This is particularly evident in TextVQA, as tertiles corresponding to smaller image-question-answer triplets have a greater relative improvement in performance, in comparison with larger tertiles. In VQAv2, however, this pattern is not as consistent, likely due to the less frequent appearance of category names for each object in questions and answers, compared to OCR tokens in TextVQA.

Appendix C further analyses how the model makes use of the high-resolution image inputs, presenting results on how the cross-attention module uses local versus global sub-images.

4 Conclusions

We presented an approach for building efficient Vision-Language Models (VLMs), processing high-resolution images while maintaining parameter efficiency. The approach was used to develop the Pheye family of VLMs, which achieve a high effectiveness on various academic task-oriented datasets, surpassing other generalist models of similar size, and achieving particularly strong results on tasks that involve understanding scene-text.

Future work directions include investigating the use of different vision encoders, for instance incorporating strategies to process images at close to native resolutions and aspect ratios, e.g. building upon the approach proposed by Dehghani et al. (2024). We can also explore ways to increase the amount of training data, given that our method still requires training hundreds of millions of randomly initialized parameters. To address this, we could generate additional task-specific synthetic data, particularly for tasks that involve scene-text, where the amount of available human generated data is smaller. Building on the work of Zhang et al. (2023a), we could generate VQA examples that focus on finer image details, which have been shown to be crucial for performance.

Limitations and Ethical Considerations

While our work does not raise new ethical issues within the domain of vision-language models (e.g., we conducted our experiments on public datasets, carefully designed for academic research and extensively used in previous studies), there are some general important concerns.

Vision-Language Models (VLMs) are, for instance, notorious for their internal biases, inherited from the training data itself or from the use of pre-trained models such as CLIP. We therefore recommend caution in the use of the approach proposed in this paper, and anticipate further research into model biases, before relying on our work beyond research environments.

Another important limitation in the work reported on this paper concerns the fact that our experiments relied exclusively on English datasets. Multilingual models have shown potential in leveraging diverse datasets and providing more robust and versatile language understanding capabilities, which could be beneficial for creating VLMs that can handle a wider variety of tasks and languages. Future work can perhaps explore the use of efficient multilingual models like Qwen (Bai et al., 2023) to enhance our approach, although additional efforts would be required in the design of an effective mixture of multilingual data for training.

Acknowledgements

This research was supported by the Portuguese Recovery and Resilience Plan through project C645008882-00000055 (i.e., the Center For Responsible AI), and also by the Fundação para a Ciência e Tecnologia (FCT), specifically through the project with reference UIDB/50021/2020 (DOI: 10.54499/UIDB/50021/2020).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. 2019. No-Caps: Novel object captioning at scale. In *Proceedings of the IEEE International Conference on Computer Vision*.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual question answering. In *Proceedings of the International Conference on Computer Vision*.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the Association for Computational Linguistics*.

Ali Furkan Biten, Ruben Tito, Andres Mafla, Lluis Gomez, Marçal Rusinol, Ernest Valveny, CV Jawahar, and Dimosthenis Karatzas. 2019. Scene text visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*.

Lin Chen, Jisong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. 2023a. ShareGPT4V: Improving large multimodal models with better captions. *arXiv preprint arXiv:2311.12793*.

Xi Chen, Xiao Wang, Lucas Beyer, Alexander Kolesnikov, Jialin Wu, Paul Voigtlaender, Basil Mustafa, Sebastian Goodman, Ibrahim Alabdulmohsin, Piotr Padlewski, et al. 2023b. PaLI-3 vision language models: Smaller, faster, stronger. *arXiv preprint arXiv:2310.09199*.

Xiangxiang Chu, Limeng Qiao, Xinyu Zhang, Shuang Xu, Fei Wei, Yang Yang, Xiaofei Sun, Yiming Hu, Xinyang Lin, Bo Zhang, et al. 2024. MobileVLM V2: Faster and stronger baseline for vision language model. *arXiv preprint arXiv:2402.03766*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al.

2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. 2024. InstructBLIP: Towards general-purpose vision-language models with instruction tuning. *Advances in Neural Information Processing Systems*.

Mostafa Dehghani, Basil Mustafa, Josip Djolonga, Jonathan Heek, Matthias Minderer, Mathilde Caron, Andreas Steiner, Joan Puigcerver, Robert Geirhos, Ibrahim M Alabdulmohsin, et al. 2024. Patch n'pack: NaViT, a vision transformer for any aspect ratio and resolution. *Advances in Neural Information Processing Systems*.

Alex Fang, Albin Madappally Jose, Amit Jain, Ludwig Schmidt, Alexander Toshev, and Vaishaal Shankar. 2023. Data filtering networks. *arXiv preprint arXiv:2309.17425*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Drew A Hudson and Christopher D Manning. 2019. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. 2014. ReferItGame: Referring to objects in photographs of natural scenes. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. In *Proceedings of the IEEE International Conference on Computer Vision*.

Jing Yu Koh, Ruslan Salakhutdinov, and Daniel Fried. 2023. Grounding language models to images for multimodal inputs and outputs. In *Proceedings of the International Conference on Machine Learning*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual Genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*.

Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. 2024. What matters when building vision-language models? *arXiv preprint arXiv:2405.02246*.

Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Fanyi Pu, Jingkang Yang, Chunyuan Li, and Ziwei Liu. 2023a. MIMIC-IT: Multi-modal in-context instruction tuning. *arXiv preprint arXiv:2306.05425*.

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

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023c. Textbooks are all you need II: Phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*.

Zhang Li, Biao Yang, Qiang Liu, Zhiyin Ma, Shuo Zhang, Jingxu Yang, Yabo Sun, Yuliang Liu, and Xiang Bai. 2023d. Monkey: Image resolution and text label are important things for large multi-modal models. *arXiv preprint arXiv:2311.06607*.

Wing Lian, Guan Wang, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". 2023. SlimOrca: An open dataset of GPT-4 augmented FLAN reasoning traces, with verification.

Bin Lin, Zhenyu Tang, Yang Ye, Jiaxi Cui, Bin Zhu, Peng Jin, Junwu Zhang, Munan Ning, and Li Yuan. 2024. MoE-LLaVA: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft COCO: Common objects in context. In *Proceedings of the European Conference on Computer Vision*.

Fuxiao Liu, Kevin Lin, Linjie Li, Jianfeng Wang, Yaser Yacoob, and Lijuan Wang. 2023. Aligning large multi-modal model with robust instruction tuning. *arXiv preprint arXiv:2306.14565*.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024. LLaVA-NeXT: Improved reasoning, ocr, and world knowledge.

Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. 2019. OK-VQA: A visual question answering benchmark requiring external knowledge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Anand Mishra, Shashank Shekhar, Ajeet K Singh, and Anirban Chakraborty. 2019. OCR-VQA: Visual question answering by reading text in images. In *Proceedings of the International Conference on Document Analysis and Recognition*.

Yasumasa Onoe, Sunayana Rane, Zachary Berger, Yonatan Bitton, Jaemin Cho, Roopal Garg, Alexander Ku, Zarana Parekh, Jordi Pont-Tuset, Garrett Tanzer, Su Wang, and Jason Baldridge. 2024. DOCCI: Descriptions of Connected and Contrasting Images. *arXiv preprint arXiv:2404.19753*.

OpenAI. 2023. GPT-4V(ision) system card.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the the Association for Computational Linguistics*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *Advances in Neural Information Processing Systems*.

Rita Ramos, Bruno Martins, Desmond Elliott, and Yova Kementchedjhieva. 2023. SmallCap: lightweight image captioning prompted with retrieval augmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. 2022. A-OKVQA: A benchmark for visual question answering using world knowledge. In *Proceedings of the European Conference on Computer Vision*.

Oleksii Sidorov, Ronghang Hu, Marcus Rohrbach, and Amanpreet Singh. 2020. TextCaps: a dataset for image captioning with reading comprehension. In *Proceedings of the European Conference on Computer Vision*.

Amanpreet Singh, Vivek Natarjan, Meet Shah, Yu Jiang, Xinlei Chen, Devi Parikh, and Marcus Rohrbach. 2019. Towards VQA models that can read. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. CIDEr: Consensus-based image description evaluation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Vikhyat. 2024. Tiny vision language model.

Junke Wang, Lingchen Meng, Zejia Weng, Bo He, Zuxuan Wu, and Yu-Gang Jiang. 2023. To see is to believe: Prompting GPT-4V for better visual instruction tuning. *arXiv preprint arXiv:2311.07574*.

Jiarui Zhang, Mahyar Khayatkhoei, Prateek Chhikara, and Filip Ilievski. 2023a. Visual cropping improves zero-shot question answering of multimodal large language models. *arXiv preprint arXiv:2310.16033*.

Yanzhe Zhang, Ruiyi Zhang, Jiuxiang Gu, Yufan Zhou, Nedim Lipka, Diyi Yang, and Tong Sun. 2023b. LLaVAR: Enhanced visual instruction tuning for text-rich image understanding. *arXiv preprint arXiv:2306.17107*.

Bo Zhao, Boya Wu, and Tiejun Huang. 2023. SVIT: Scaling up visual instruction tuning. *arXiv preprint arXiv:2307.04087*.

A Data Mixture for Final Training Stage

Table 3 summarizes the datasets considered for Stage III of the proposed model training procedure, showing the response formatting prompt associated to each of the considered datasets.

Data	Size	Response Formatting Prompts
LLaVA	158K	-
LLaVAR	20K	
VQAv2	166K	Answer the question using a single word or phrase
GQA	144K	
OK-VQA	9K	
OCR-VQA	80K	
TextVQA	35K	
ST-VQA	26K	
A-OKVQA	17k	Answer with the option's letter from the given choices directly.
TextCaps	22K	Generate a one-sentence caption for the provided image, incorporating textual elements visible in the image.
Total	676K	

Table 3: Data mixture used in Stage III of training.

B Preliminary Experiments Assessing Architectural Choices

Through a set of preliminary experiments, we compared different approaches to combine the vision and language modalities, as well as approaches for encoding high resolution images, using relatively small Vision Transformers (ViTs) and language models in order to validate architectural decisions.

The experiments followed the main principles of Flamingo (Alayrac et al., 2022), where the language and vision models were frozen to preserve their pre-training knowledge.

B.1 Vision-Language Model Architectures

One initial experiment evaluated three different alternatives for combining the vision and language modalities, taking inspiration from the FROMAGe (Koh et al., 2023), LLaVA, and SmallCap (Ramos et al., 2023) neural architectures.

Building upon FROMAGe, our first approach involves transforming an image into a set of visual embeddings using a pre-trained ViT. The resulting visual inputs are then summarized through the extraction of the [CLS] token, which is subsequently mapped to the language model's dimensionality via a linear transformation. In contrast, the second architecture (i.e., the one inspired by LLaVA) leverages all the embeddings generated by the ViT to represent an image, rather than relying solely on the [CLS] token. In both cases, the language

	Train	Param.	CIDEr	B@4	M
FROMAGe	✗	590K	48.7	14.9	17.8
FROMAGe	✓	88M	66.5	21.9	20.7
LLaVA	✗	590K	81.2	24.1	22.1
LLaVA	✓	88M	92.4	28.3	24.0
SmallCap	✗	28M	106.8	32.6	26.4
SmallCap	✓	116M	107.4	33.7	26.8

Table 4: Comparison of different approaches for combining the vision and language modalities. The column "Train" denotes models in which the ViT was trained, and the column "Param." denotes the number of trainable parameters. "B@4" and "M" represent the BLEU-4 (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005) metrics, respectively.

model takes as input the concatenation of the vision embeddings and the text prompt embeddings, allowing it to contextualize the image with the accompanying text, to generate a relevant response.

The third architecture, inspired by SmallCap, also makes use of a ViT that extracts features from an image. However, it differs from the last two architectures as it makes use of cross-attention modules inserted between the self-attention and the feedforward modules in the decoder, similarly to the original encoder-decoder Transformer, to bridge the modalities. The inner dimensionality of the cross-attention modules matches the hidden dimensionality of the language model.

To compare the effectiveness of the aforementioned three architectures, we trained and evaluated them on a captioning task using the COCO dataset (Lin et al., 2014). Specifically, we trained our models on the training split, using a cross-entropy loss, and evaluated them on the validation split. The ViT was initialized with pre-trained weights from a CLIP-ViT-B-32 model, while the language model was initialized with pre-trained weights from the GPT-2 base model (Radford et al., 2019). During training, the language model was kept frozen, while the connector module, comprising either a linear mapping in the first two architectures or cross-attention modules in the third, was trained. Additionally, we experimented with training the ViT. The models were trained for 5 epochs using the AdamW optimizer, with a batch size of 64 instances, and a learning rate of 1e-4 for the cross-attention modules and 1e-5 for the vision encoder. The GPT-2 model was trained to generate captions using the prefix `"This image shows "`, and the model was also prompted with this prefix during the evaluation stage.

The results of this experiment are summarized in Table 4, and they reveal that training the vision encoder together with the connector module yields improved performance across all architectures, when compared to training the connector module alone. This is expected, given that training the ViT results in training a larger number of parameters. Furthermore, our findings suggest that using the full image embeddings results in better performance relative to using solely the `[CLS]` token. This result is also intuitive, as multiple visual tokens can capture more nuanced details about the input image than a single vector. More importantly, in an experimental setup where the language model is not trained, the architecture inspired by SmallCap clearly demonstrates superior performance. Interestingly, using cross-attention modules as the modality bridge seems to outperform other architectures, like LLaVA, even when considering more trainable parameters associated to training the ViT.

Similar findings to those reported in this section have also been reported by Laurençon et al. (2024), although these authors have also showed that LLaVA-style linear projections can outperform cross-attention if, besides the ViT, the language model is also carefully trained.

B.2 Encoding High Resolution Images

Previous attempts at increasing the input resolution of VLMs mostly chose to finetune the vision encoder on higher resolution images (Chen et al., 2023b), which is costly due to the quadratic computational complexity of the attention mechanism, and which also requires large amounts of image-text data. To overcome this issue, we experimented with an approach similar to that of Li et al. (2023d), in which we scale up the input resolution by splitting the image into smaller patches that match the resolution of the vision encoder, afterwards encoding the smaller sub-images individually. To provide the model with global context, we also encode the full image as normal, concatenate all feature maps, and use the result as our image representation.

Since the smaller image patches can have a different distribution than that of the images in which the ViT was trained on, we introduce two sets of LoRA adapters to the vision encoder. One is used on the global image and the other is used on the smaller local patches, allowing each resolution to specialize on different size aspects of the input image. The adapters were inserted at every linear layer of the ViT with the following hyperparame-

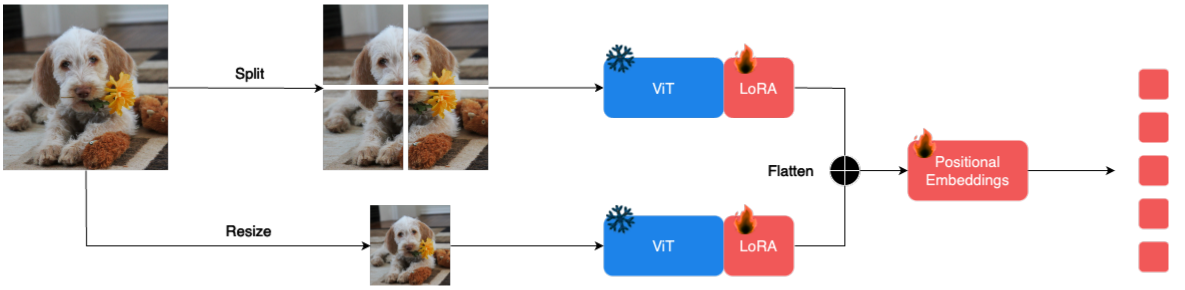

Figure 3: Architecture for high resolution multi-patch image encoding.

ters: rank of 8, alpha of 16, and dropout probability of 0.05. We concatenate the resulting visual tokens, and add to each patch learned positional embeddings, as shown in Figure 3.

The proposed method generates image representations with larger sequence lengths, which can be computationally demanding. To mitigate this issue, we explored the use of two resampler architectures, namely a 6-layer Perceiver Resampler, as introduced in Flamingo (Alayrac et al., 2022), and the Monkey Resampler, which can be characterized as a single layer Perceiver Resampler that does not include a feed-forward module and residual connections for the query vectors. In our experiments, both modules used 257 query vectors with the same dimensionality of the vision encoder. Additionally, we compare the aforementioned resamplers, which encode images at 448x448 pixels, with an encoder that processes images at 224x224 pixels, and also with a 448x448 pixels encoder that does not compress the image features. For a fair comparison, the smaller 224x224 pixels encoder also includes one set of LoRA adapters.

Due to the fact that scaling up the resolution is likely to benefit tasks that involve scene-text and OCR the most, we pre-train our models on the LLaVAR (Zhang et al., 2023b) dataset for 1 epoch, finetune on the TextCaps (Sidorov et al., 2020) train split for 3 epochs, and finally evaluate on the TextCaps validation split. When finetuning our models, we use the prompt "This image shows ", similarly to the previous experiment. Our vision Transformer and language model are initialized from pre-trained CLIP-ViT-L-14 and GPT-2 large models, respectively, and are combined with cross-attention modules, following SmallCap. The use of larger a ViT and language model, when compared to the experiment described in the previous appendix, relates to the fact that tasks involving scene-text and OCR are more demanding, and hence slightly larger models can facilitate the assessment of differences between the architectural alternatives being compared. We freeze the ViT and language model, and only train the cross-attention modules, positional embeddings, LoRA, and resamplers, using a cross-entropy loss over non-prompt tokens. We use the AdamW optimizer, with a learning rate of 1e-4 for the pre-training stage and 2e-5 for the fine-tuning stage, and with a batch size of 64 in both stages. The results are shown in Table 5.

The experiment revealed that increasing the input resolution from 224x224 to 448x448 pixels increases captioning performance, independently from whether we compress the resulting visual tokens or not. There does not seem to be a large performance difference in how we compress the visual tokens, since the Monkey Resampler and the Perceiver Resampler have similar CIDEr (Vedantam et al., 2015) scores on the TextCaps validation split. However, this might not be the case for other tasks. Finally, the architecture with the best results is the one that increases the input resolution from 224x224 pixels to 448x448 pixels without compressing the vision features.

C Assessing Changes in Cross-Attention According to Different Path Sizes

This appendix further analyses the use of high-resolution inputs, assessing the reliance of the model on the global versus local sub-images.

To investigate how the model uses the different patches in the cross-attention module, we calculated the average of the attention scores for the generated captions across all TextCaps validation set images. Specifically we compute the average of the cross-modal attention scores across the global image tokens and the local patches tokens, at each step of generating caption tokens and across all attention heads. Figure 4 shows the global patch average attention scores for the Pheye-x4 models, separately for each of the layers in which cross-

Resolution	Resampler	Visual Token Length	Trainable Parameters	CIDEr	BLEU-4	METEOR
224	-	257	241M	104.5	27.6	24.6
448	Monkey	257	248M	110.1	28.1	24.9
448	Flamingo	257	317M	110.3	28.2	24.9
448	-	1285	244M	113.2	28.5	25.2

Table 5: Comparison of different strategies for handling high resolution images as input.

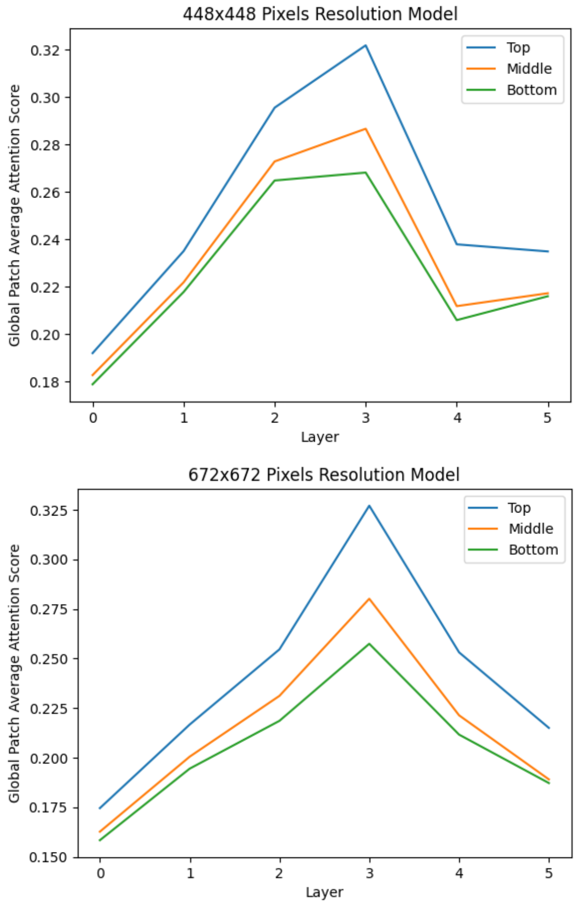

Figure 4: Attention scores for global patch tokens across data tertiles that reflect the relative dimensions of relevant image areas. Both graphs were calculated using the Pheye-x4 models. The average cross-attention score for the local patches is given by $1 - A_G$, were A_G denotes the cross-attention scores for the global patch tokens.

attention operations were inserted.

After computing the attention scores, we segmented the results using a similar approach to that described in Section 3.3, with the only difference being that we replace the ground-truth answers with reference captions. This analysis reveals that images requiring finer detail understanding tend to favor local patches over global patch tokens, as they necessitate higher resolution inputs. In contrast, images with larger scene-text tend to rely more on global patch tokens. Moreover, we see that although the visual token length is different in both models present in Figure 4, the attention scores for the global patch are similar across all dense cross-attention layers.