Shaking Up VLMs: Comparing Transformers and Structured State Space Models for Vision & Language Modeling

Georgios Pantazopoulos[1,2] **Malvina Nikandrou**[1] **Alessandro Suglia**[1,2]
Oliver Lemon[1,2] **Arash Eshghi**[1,2]

[1]Heriot-Watt University; [2]Alana AI

{gmp2000, mn2002, a.suglia, o.lemon, a.eshghi}@hw.ac.uk

Abstract

This study explores replacing Transformers in Visual Language Models (VLMs) with Mamba, a recent structured state space model (SSM) that demonstrates promising performance in sequence modeling. We test models up to 3B parameters under controlled conditions, showing that Mamba-based VLMs outperforms Transformers-based VLMs in captioning, question answering, and reading comprehension. However, we find that Transformers achieve greater performance in visual grounding and the performance gap widens with scale. We explore two hypotheses to explain this phenomenon: 1) the effect of task-agnostic visual encoding on the updates of the hidden states, and 2) the difficulty in performing visual grounding from the perspective of in-context multimodal retrieval. Our results indicate that a task-aware encoding yields minimal performance gains on grounding, however, Transformers significantly outperform Mamba at in-context multimodal retrieval. Overall, Mamba shows promising performance on tasks where the correct output relies on a summary of the image but struggles when retrieval of explicit information from the context is required[1].

1 Introduction

Modern Visual Language Models (VLMs) (Bai et al., 2023a; Li et al., 2024; Alayrac et al., 2022) typically treat patch representations from vision encoders (Radford et al., 2021; Fang et al., 2023; Zhai et al., 2023) as tokens that are mapped to the embedding space of a Transformer-based Large Language Model (LLM). This patch-as-token approach has fostered the development of VLMs that have achieved unprecedented performance on established Vision & Language (VL) on many coarse-grained tasks, for example, image captioning (Lin et al., 2014) or visual question answering (Goyal et al., 2017; Hudson and Manning, 2019). However, fine-grained tasks such as localizing regions within an image (Peng et al., 2023b; Kazemzadeh et al., 2014), or reading text (Sidorov et al., 2020; Mathew et al., 2021) from the image are significantly more challenging for these models. These tasks require the model to grasp nuances within the image beyond summarizing the visual context in a few words as in conventional image captioning.

A straightforward countermeasure is to scale up the resolution of images, allowing the VLM to "see greater details". (Liu et al., 2023b; Karamcheti et al., 2024; McKinzie et al., 2024). On the other hand, increasing the context length requires substantial overhead as Transformer-based VLMs have quadratic complexity with respect to the input. Structured state space models (SSMs) (Gu et al., 2022; Poli et al., 2023) have recently emerged, providing competitive performance against Transformers. Mamba (Gu and Dao, 2023) is a recent SSM that promises computational efficiency as well as performance that surpasses Transformer-based language models of similar size.

In this paper, we investigate whether a Mamba LLM is a competitive alternative to a Transformer across established multimodal tasks including both fine-grained and coarse-grained multimodal tasks. The choice of the LLM plays a crucial role for modern VLMs, as recent work (Laurençon et al., 2024b) has shown that for a fixed number of total parameters, the quality of the language backbone has a higher impact than that of the vision backbone. More specifically, we train three Mamba-VL variants and compare them against Pythia-VL, a series of equally sized models that follow the established paradigm to train VLMs with a state-of-the-art Transformer-based LLM backbone (Biderman et al., 2023). Notably, the performance of Pythia-VL is comparable with that of existing VLMs, thus establishing it as a robust baseline model. We emphasize that both models are trained on the exact

---

[1]Code available here.

same data presented in the same order, and with identical training hyperparameters, allowing us to provide precise indications of the strengths and weaknesses of the two approaches.

We find that Mamba-VL outperforms Pythia-VL in captioning, question answering, and reading comprehension, but Pythia-VL models consistently achieve greater performance in grounding tasks, and this gap widens in larger models. To identify the issue of the difference in performance, we explore the impact of task-agnostic visual encoding, where the model produces embeddings for image representations without information about the task. While task-aware image encoding provides a modest improvement in Mamba-VL's grounding capabilities, it remains inferior to the performance of the Transformer-based VLMs. We investigate this further by casting visual grounding as an in-context multimodal retrieval task, where the model has to retrieve the correct token from the sequence associated with the query. Our results show that Transformers are notably more sample efficient, indicating an inherent limitation of Mamba in retrieval-oriented tasks, despite the promising results in sequence modeling. All in all, these experiments showcase that Mamba can be quite effective when the downstream task requires a summary of the image but struggles in tasks where it has to retrieve fine-grained details from the image.

2 Related Work

2.1 VLMs

Early works showcase the capabilities of LLMs combined with pretrained vision encoders, in VL tasks (Tsimpoukelli et al., 2021). Consequently, current VLMs (Bai et al., 2023b; Dai et al., 2024; Alayrac et al., 2022; Laurençon et al., 2024b; Liu et al., 2024a; Chen et al., 2023b) are based on the same foundational formula: a visual expert (Zhai et al., 2023; Fang et al., 2023), a language backbone (Touvron et al., 2023; Jiang et al., 2023; Bai et al., 2023a; Team et al., 2024), and a connector between the two modules. The vast majority of these models are based on highly capable Transformer-based LLMs. In this work, while we do not modify this formula, we investigate the effect of replacing the Transformer LLM with Mamba.

2.2 Structured State Space Models

Structured state space sequence models (S4) are a family of models of sequence models using principles from RNNs, CNNs, and classical state space models that attempt to combat the limitations of Transformers in modeling long sequences (Fu et al., 2023; Poli et al., 2023; Gu et al., 2022; Smith et al., 2023). These models showcase convincing results in modeling long-range dependencies across several synthetic tasks (Tay et al., 2021). Previous research shows, in a controlled study of moderately sized models, that Transformers outperform S4 models in terms of language modeling (Arora et al., 2024). However, Mamba (Gu and Dao, 2023) builds upon previous S4 models by introducing a selective scan operation (Section 3.1) showing competitive performance against Transformers.

Mamba applications Inspired by its results in sequence modeling, recent work applies Mamba to computer vision tasks, by introducing inductive biases that better match the domain of image encoding (Zhu et al., 2024a; Huang et al., 2024; Ruan and Xiang, 2024; Liu et al., 2024b). Within NLP, Jamba (Lieber et al., 2024) is a hybrid architecture with interleaved Transformer and Mamba blocks, while MambaByte (Wang et al., 2024), is a language model operating on bytes instead of subwords. To the best of our knowledge, there is not yet a comprehensive study showcasing the effectiveness of Mamba in multimodal settings. Concurrent work has applied Mamba in multimodal tasks (Zhao et al., 2024a; Qiao et al., 2024). However, these studies offer limited insights, because 1) they do not facilitate a fair comparison under controlled conditions, and 2) they do not investigate multimodal tasks that require both high-level and fine-grained information, such as visual grounding.

Transformers vs SSMs The development of SSMs and similar RNNs (Katharopoulos et al., 2020; Fu et al., 2023; Peng et al., 2023a; Poli et al., 2023) with competitive performance, has motivated comparisons with Transformers. Recent studies (Park et al., 2024; Grazzi et al., 2024) show that SSMs can match the in-context learning performance of Transformers on certain tasks, but Akyürek et al. (2024) demonstrate that Transformers retain an advantage for in-context language learning. Moreover, Merrill et al. (2024) provide theoretical and empirical evidence contrary to previous claims (Gu et al., 2021), showing that SSMs and Transformers have limited expressivity making them unsuitable for state-tracking problems. In terms of the in-context retrieval (e.g., copying) ca-
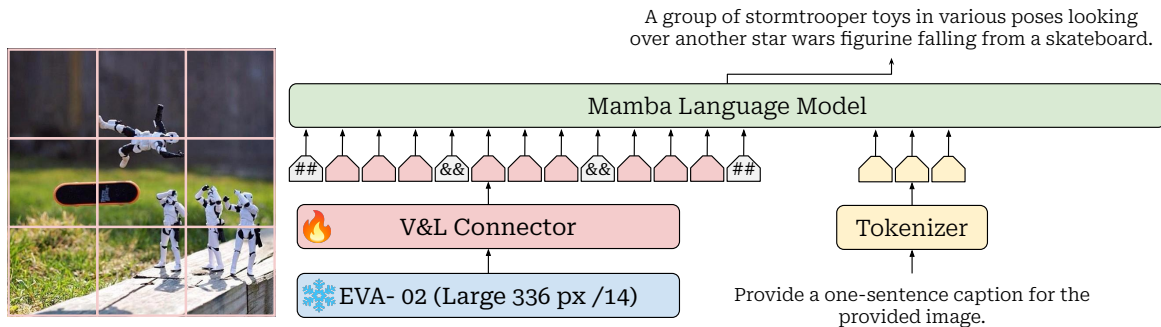
Figure 1: Overview of Mamba-VL. We embed images using EVA-02 and use an MLP as V&L connector to align the image with text embeddings before the Mamba backbone. Because Mamba does not encode positional information, we introduce custom tokens that delineate the beginning and the end position of the image in the sequence. We also use custom tokens that act as row separators within the image. The vision encoder is kept frozen during training.

pabilities of selective SSMs, Gu and Dao (2023) show that Mamba is capable of performing associative recall, as formulated by the Induction Heads (Olsson et al., 2022) task. However, follow-up work (Jelassi et al., 2024; Wen et al., 2024) provides evidence that SSMs fall behind Transformers when the copying task requires precise retrieval from the context. We leverage these insights from previous work to draw parallels with VL tasks. In particular, we formulate a synthetic task for multimodal in-context retrieval to explain the limitation of Mamba in visual grounding.

3 VLM Approach

3.1 Preliminaries: The Mamba model

S4 models (Gu et al., 2022) take inspiration from Linear Time-Invariant (LTI) models that map a sequence $x(t) \in \mathbb{R} \rightarrowtail y(t) \in \mathbb{R}$ through a hidden state $h(t) \in \mathbb{R}^N$. The output of an LTI model is computed in a two-stage format:

$$h'(t) = Ah(t) + Bx(t), \tag{1a}$$
$$y(t) = Ch(t) \tag{1b}$$

S4 models first transform the continuous parameters $(\mathbf{A}, \mathbf{B})$ with a discretization step with $\boldsymbol{\Delta}$ parameters, into discrete parameters $(\bar{\mathbf{A}}, \bar{\mathbf{B}})$. Given the discrete parameters $\bar{\mathbf{A}}, \bar{\mathbf{B}}$ the discrete update is defined in recurrent form Equation (2a), or via the convolution form Equation (3a):

$$h_t = \bar{\mathbf{A}}h_{t-1} + \bar{\mathbf{B}}x_t \tag{2a}$$
$$y_t = \mathbf{C}h_t \tag{2b}$$

$$\bar{\mathbf{K}} = (\mathbf{C}\bar{\mathbf{B}}, \mathbf{C}\bar{\mathbf{A}}\bar{\mathbf{B}}, \dots, \mathbf{C}\bar{\mathbf{A}}^k\bar{\mathbf{B}}) \tag{3a}$$
$$y = x * \bar{\mathbf{K}} \tag{3b}$$

However, for language modeling S4 models underperform attention-based models (Arora et al., 2023). Gu and Dao (2023) empirically show that the time-independent parameters of an S4 model are not sufficient to select the correct information from their context as it is not straightforward how to reset the hidden state at each timestep. For this purpose, certain parameters of the Mamba model $(\Delta, \mathbf{B}, \mathbf{C})$ are allowed to be functions of the input. With this change, hidden states can be updated in a selective fashion over the input – though due to violation of the convolution view (Equation (3a)), this requires a hardware-aware implementation to compute the hidden states efficiently. For additional implementation details of Mamba please see the original paper (Gu and Dao, 2023).

3.2 Model Architecture

Figure 1 shows an overview of our model. We built our approach using the standard paradigm for VLMs that combine unimodal experts (Liu et al., 2024a; Alayrac et al., 2022; Dai et al., 2024). More specifically, our model consists of three individual components, a vision encoder, the Vision & Language connector, and the language backbone.

Vision Encoder We use EVA-02-L336px/14 (Fang et al., 2023) to obtain high-quality visual representations. While previous work usually adopts CLIP models (Bai et al., 2023b; Liu et al., 2024a), the EVA series outperforms the existing open CLIP models. We also provide results in Appendix C, showcasing a comparison between the two vision encoders using preliminary checkpoints. Furthermore, based on previous work (Karamcheti et al., 2024), we opted for higher resolution images, as it has been shown that it leads to performance gains.

Figure 2: Overview of task categorization and format. We leverage a collection of datasets for coarse-grained (e.g., image captioning, visual question answering) and fine-grained (e.g., visual grounding, reading comprehension) multimodal tasks. Text in purple indicates the outputs of a model for each task. * denotes held-out datasets.

Vision & Language Connector We follow LLaVA-1.5 (Liu et al., 2023b) and use a two-layer MLP that projects the visual tokens to the dimensionality expected by the LLM, leaving more sophisticated architectural choices (Dai et al., 2024; Bai et al., 2023b; You et al., 2023) for future work.

Language Backbone We use Mamba or Pythia (Biderman et al., 2023) as the language backbone that accepts the visual features from the connector module, and the tokenized text containing the task instruction and any sample text. We select Pythia as the baseline Transformer-based language model because it enables direct comparison as it 1) follows the state-of-the-art Transformer recipe (Su et al., 2024; Dao, 2023), 2) is trained on the same dataset as Mamba (Gao et al., 2020), 3) provides model variants with a similar number of parameters.

A key difference between the two models is that Mamba does not allocate parameters to model positional information. This inductive bias has been identified by concurrent work (Liu et al., 2024c; Zhu et al., 2024a), applying Mamba to computer vision tasks, since positional embeddings capture the structure of the image. Inspired by Fuyu (Rohan et al., 2023), we overcome this issue by introducing a separator token ("##") that signals the beginning and the end of the image sequence, as well as an image-newline character ("&&") that depicts the end of a row of patches.

4 Datasets

We use a collection of open-source datasets to allow a fully reproducible comparison. For pretraining, we leverage the dataset from Liu et al. (2024a), a subset of 595K captions from Conceptual Captions 3M (Sharma et al., 2018). For instruction tuning, we use a collection of established coarse and fine-grained vision-language tasks (e.g., captioning, visual question answering, and referring expression). Figure 2 shows examples for all tasks in our training and evaluation. We provide details for our dataset, filtering approach, and task instructions in Appendix A. Notably, we pack the examples from the same image and task into one sequence.

5 Experiments

5.1 Experimental Setup

Similar to previous work (Liu et al., 2024a; Li et al., 2024) we employ a two-step training regime. First, we perform a warmup stage where we train only the connector component on the pretraining dataset. Next, we unfreeze the language model parameters and train on the instruction-tuning dataset. All models are trained using the same data, in the same order, and with identical training hyperparameters (see Appendix B for further details). Unless stated otherwise, we report the evaluation performance without task-specific fine-tuning.

Model	LLM Param	**Image Captioning**			**General VQA**				**Misc**	
		COCO test	NoCaps* val	Sum	VQAv2 val	GQA test-dev	V7W test-T	Sum	VSR test	POPE* test
Pythia-VL	1B	132.89	97.61	230.50	72.26	53.79	81.96	208.81	72.43	86.77
Mamba-VL	790M	133.81	99.00	232.81 (+2.31)	71.67	54.95	81.82	208.44 (-0.37)	75.39	86.77
Pythia-VL	1.4B	134.06	100.72	234.78	73.57	57.05	83.06	213.68	77.72	86.40
Mamba-VL	1.4B	134.76	100.56	235.32 (+0.54)	74.46	58.44	83.78	216.67 (+2.99)	80.18	85.32
Pythia-VL	2.8B	134.97	101.27	236.24	75.08	59.76	84.34	219.18	80.86	86.87
Mamba-VL	2.8B	**135.53**	**102.00**	**237.53** (+1.29)	**76.08**	**60.41**	**85.31**	**221.80** (+2.62)	**81.45**	**87.33**

Table 1: Results on image captioning, general VQA, and misc benchmarks. * denotes zero-shot performance.

Model	LLM Param	**Visual Grounding**							**Reading Comprehension**			
		RefCOCO test-A	test-B	RefCOCO+ test-A	test-B	RefCOCOg test	V7W test-P	Sum	TextCaps val	TextVQA† val	AI2D test	Sum
Pythia-VL	1B	76.00	62.48	45.36	47.44	67.58	83.78	382.64	92.73	35.22	77.62	205.57
Mamba-VL	790M	67.84	56.35	57.97	41.43	59.16	74.01	356.76 (-25.88)	94.30	40.72	79.27	214.29 (+8.72)
Pythia-VL	1.4B	82.43	68.39	72.35	55.16	72.56	86.13	437.02	94.60	37.54	79.27	211.41
Mamba-VL	1.4B	76.60	63.48	68.40	52.11	68.82	80.18	409.59 (-27.43)	98.68	41.30	80.86	220.84 (+9.43)
Pythia-VL	2.8B	**85.39**	**70.82**	**75.39**	**58.62**	**76.24**	**86.61**	**453.07**	99.74	39.14	81.57	220.45
Mamba-VL	2.8B	79.29	64.97	71.64	53.94	71.27	82.50	423.61 (-29.45)	**100.47**	**42.14**	**83.71**	**226.32** (+5.87)

Table 2: Results on visual grounding, and text-oriented, benchmarks. † denotes a task not in the training mixture.
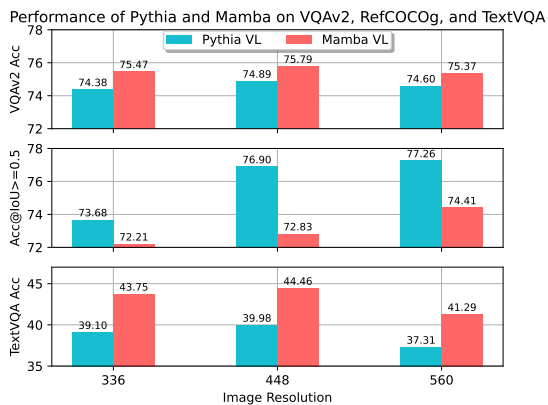

Figure 3: Results of finetuned 1.4B models with increased resolution on VQAv2 (top), RefCOCOg (middle), and TextVQA (bottom). Increasing the resolution to 480×480 pixels results better performance for both models, however, Pythia benefits significantly more than Mamba in the grounding task.

5.2 Results

Pythia vs Mamba Table 1 and Table 2 illustrate the comparison between Pythia-VL and Mamba-VL across three model sizes. We provide results for each benchmark individually, along with a summation score as an indication of overall performance for a task group. We observe that Mamba variants match or surpass the performance of models with Pythia as an LLM across all three sizes in most tasks. Specifically, the smallest Mamba-VL

achieves competitive performance with Pythia-VL even though it has approximately 200M fewer parameters but also outperforms Pythia-VL on zero-shot image captioning (NoCaps) and on spatial understanding (VSR). However, the performance gap decreases proportionally to the size of the compared models. The largest performance difference is observed in the reading comprehension tasks. We hypothesize that textual information within an image provides a strong signal for Mamba to maintain this information in the hidden state. Surprisingly, Pythia-VL models consistently outperform Mamba-VL on grounding tasks across all scales, but also this gap is further widened in larger models.

Finetuning with Higher Resolution It is widely known that increasing the image resolution yields benefits in Transformer-based VLMs (Karamcheti et al., 2024; Laurençon et al., 2024b). We explore whether the benefits of higher image resolution translate to Mamba given its strong long sequence modeling capabilities (Gu and Dao, 2023). Figure 3 shows the performance of 1.4B models on VQAv2, RefCOCOg, and TextVQA after finetuning on each task with higher-resolution images. As expected, both models benefit from higher-resolution images, and the differences are more evident in RefCOCOg, possibly due to the granularity of the task. Comparing Pythia-VL and Mamba-VL, both models exhibit a similarly small improvement in VQAv2

Model	LLM	NoCaps* val	VQA test-dev	GQA test-dev	RefCOCOg test	V7W (P) test	TextVQA val	AI2D val	POPE* test
LLaVA-1.5 (2024a)	Vicuna-7B	-	78.5	62.0	-	-	58.2	-	85.8
InstructBLIP (2024)	Vicuna-7B	123.1	-	49.2	-	-	50.1	-	83.7
Shikra (2023a)	Vicuna-7B	-	77.4	-	82.19	85.33	-	-	83.9
Ferret-v2-7B (2024)	Vicuna-7B	-	81.5	64.7	89.27	-	61.7	-	87.8
Qwen-VL-Chat (2023b)	Qwen-7B	120.2	78.2	57.5	86.32	-	61.5	62.3	-
IDEFICS2 (2024b)	Mistral-7B-v0.1	-	81.2	-	-	-	73.0	-	-
LLaVA-Phi (2024b)	Phi2-2.7B	-	71.4	-	-	-	48.6	-	85.0
TinyLLaVA (2024)	Phi2-2.7B	-	79.9	62.0	-	-	59.1	-	86.4
Cobra (2024a)	Mamba-2.8B	-	75.9	58.5	-	-	46.0	-	88.0
VL-Mamba (2024c)	Mamba-2.8B	-	76.6	56.2	-	-	48.9	-	84.4
Pythia-VL	Pythia-2.8B	100.72	77.0	59.8	76.24	86.61	39.1	81.6	86.9
Mamba-VL	Mamba-2.8B	100.56	78.0	60.4	71.27	82.50	42.1	87.3	87.3

Table 3: Results against state-of-the-art models. * denotes zero-shot performance.

and TextVQA, but Pythia-VL benefits substantially more than Mamba-VL in RefCOCOg. This provides further evidence regarding the limitations of Mamba on grounding tasks, on which we further elaborate in Section 5.3.

Comparison with SOTA models For completeness, we provide a comparison against state-of-the-art 3B and 7B parameter models (Table 3). We observe that our largest models are competitive even against the largest VLM models. Our model performs on par with other Mamba-based VLMs (Cobra and VL-Mamba) with a small advantage in general VQA benchmarks (VQA, GQA). Importantly, we want to note that our base LLMs have not been instruction-tuned, which could have a major impact, particularly in multimodal language modeling tasks (Laurençon et al., 2024b). Furthermore, it is hard to draw definite conclusions between different models as they have been trained using different datasets and training regimes.

5.3 Why is Grounding Difficult for Mamba?

We observed that Mamba models are quite effective in multimodal language modeling tasks (e.g., captioning, visual question answering). However, they underperform compared to Transformers of equal capacity in visual grounding tasks. *What is the underlying reason for this weakness?* We explore two possible explanations using the 1.4B parameter models by 1) examining the effect of task-agnostic visual encoding, and 2) framing visual grounding as an in-context multimodal retrieval task.

5.3.1 Task-agnostic Visual Encoding

Both Transformer causal models and SSMs operate unidirectionally, i.e. the representation at a given timestep is a function of only the previous and
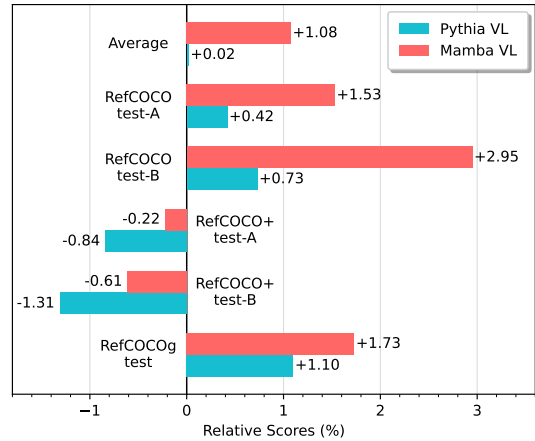

Figure 4: Relative performance difference on visual grounding benchmarks between task-aware and task-agnostic visual encoding. On average, task-aware encoding yields a marginal performance boost on Mamba-VL while it has almost no effect on Pythia-VL.

current tokens. However, SSMs enforce a stricter update rule, where the hidden state can only be updated with information from the previous hidden state and the current input (Equation (2a)). Consequently, when the image precedes the instruction, patch representations are encoded in a *task-agnostic* manner. Intuitively, this might lead the model to store "generic" information in its hidden state, which is useful for multimodal language modeling tasks but ineffective in explicit visual grounding, where the model has to remember the spatial positions of any entity in the image. On the other hand, in Transformer models, the hidden state of each timestep has direct access to all previous timesteps and, therefore, can retrieve relevant information in later hidden states.

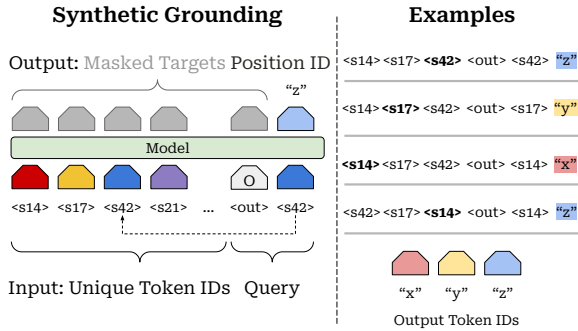We investigate the impact of *task-aware* visual

Figure 5: Overview of the synthetic visual grounding task. The model accepts as input a sequence of unique special tokens, followed by an output token and a special token id that appears in the context as a query. The model needs to predict the token id corresponding to the position of the queried token.

encoding by placing the task instruction before the image during the instruction-tuning stage. In principle, this simple modification favors Mamba as the model may choose to store or ignore inputs that are not relevant to the task. Figure 4 shows the gain of both models using task-aware encoding on visual grounding benchmarks. We observe that on average the task-aware encoding leads to a small relative improvement for Mamba-VL, but that even in this setup, Pythia-VL achieves higher performance (see Table 11 for full results). Furthermore, the results vary across different grounding benchmarks, but also across other tasks suggesting that the task-aware encoding is not always beneficial.

Perhaps the performance of Mamba-based VLMs on grounding, as well as on other tasks, can be further improved by incorporating the task instruction, and the query (e.g., question, referring expression) before the image. This is in line with recent work (Jelassi et al., 2024) showing that, when the query is available at the beginning of the input, SSMs can perform on par with Transformers on toy associative recall tasks. However, this is at odds with the common practice of data packing in current VLM training (Bai et al., 2023a; Li et al., 2024). We anticipate that naively separating the queries and outputs with image tokens can negatively affect the capabilities of a model.

5.3.2 Grounding as Multimodal Retrieval

We can view visual grounding as an in-context multimodal retrieval task. In a standard in-context retrieval task, the model is provided with a context (a text paragraph) and a query (a relevant question), and it needs to extract and copy the part of the

input corresponding to the question. Similarly, in a visual grounding task, the model is provided with a series of patch tokens as context and a prompt and needs to reference the area that matches the prompt. The core difference is that the space of the token embeddings is different. In the standard retrieval task, the inputs and outputs of the model are both in text form, whereas in visual grounding the VLM performs a two-hop step by matching the text prompt to the visual modality and then providing a textual response.

For this purpose, motivated by concurrent work (Jelassi et al., 2024; Merrill et al., 2024), we introduce a synthetic task (Figure 5) that frames visual grounding as a retrieval objective and facilitates an interpretable model comparison. We provide a pretrained model with a context of unique special tokens (<s14><s17><s42>...), followed by a query (<out><s42>). To incorporate the two-hop step between two modalities, we ask the model to return the token id from the vocabulary that matches the position of the special token in the sequence ("z"). This setup resembles how the language model of a VLM adapts to two modalities. We resize the embedding layer of the pretrained models to accommodate the new special tokens (the patch tokens in the VLMs[2]), and task the model to learn a mapping between the textual and the new embedding space. Finally, visual grounding is an instantiation of this synthetic task, where the input sequence is composed of the patch representations, the query token is the prompt, and the outputs are the pixels that match the prompt in the image.

We experiment with varying the sequence length (50/100/200, see Appendix C.4 for details). For each sequence, we use three different learning rates and train each model with three initializations of the embeddings of the special tokens (9 runs in total per sequence length). We track the performance on a held-out set and terminate training whenever the model achieves $\geq 95\%$ accuracy.

Figure 6 shows the results of both models. We observe that Pythia learns to solve the task consistently using approximately 10% of the training data. On the other hand, Mamba is less sample-efficient requiring nearly double the amount of training when increasing the sequence length, and even fails to reach the accuracy threshold for some runs with longer sequences. These results show

---

[2]In practice, VLMs do not resize the embedding layer of the LLM, but accept the embeddings from the visual encoder.

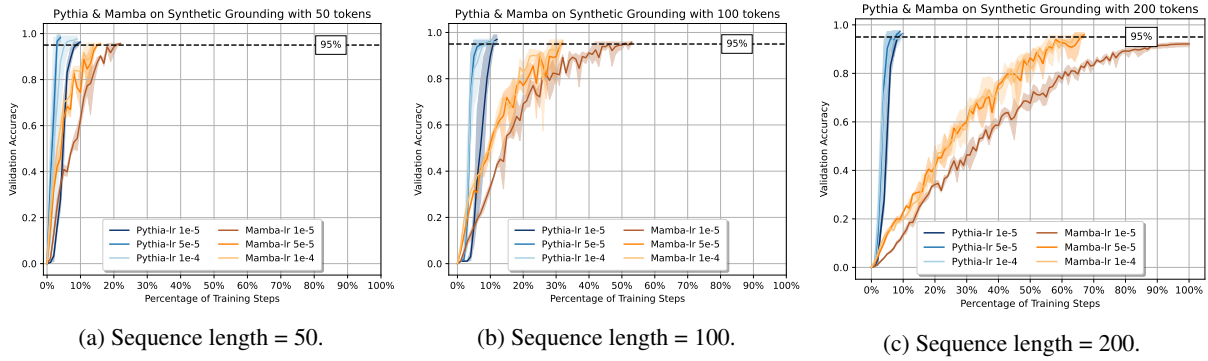| (a) Sequence length = 50. | (b) Sequence length = 100. | (c) Sequence length = 200. |

Figure 6: Performance curves for Pythia-1.4B and Mamba-1.4B variants on the synthetic grounding task with varying sequence length. Pythia learns the task significantly faster compared to Mamba.
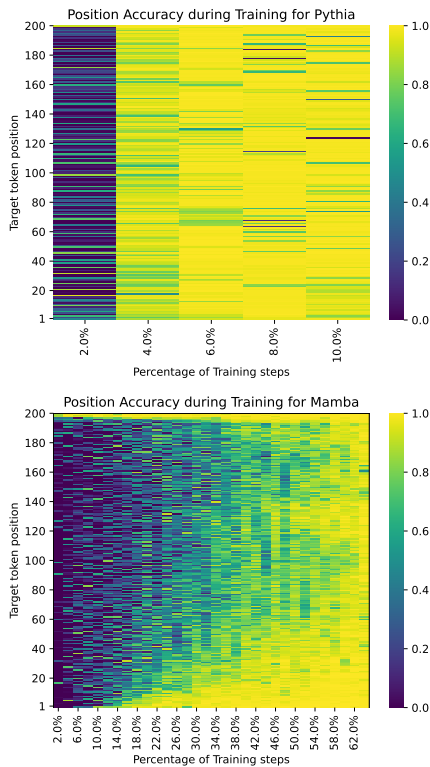

Figure 7: Accuracy per position on the held-out set during training on sequences of 200 tokens.

that in tasks requiring access to the whole context, Mamba struggles to retrieve information from its fixed-size hidden state. Transformers do not encounter this challenge as the representation of a token is informed by all preceding tokens.

Finally, we discuss how Transformers and Mamba learn to perform in-context retrieval. Figure 7 illustrates the performance per target token of both models on the synthetic grounding task with sequences of 200 tokens. Pythia learns the correct target position uniformly. On the other hand, Mamba exhibits a different pattern: at the

early stages of training it performs adequately in sequences where the target token is located at the end, gradually learns to retrieve the correct token in sequences where the target is at the beginning, and finally, at the end of training learns the task with a target token in between the sequence.

6 Conclusion

Implications of Findings In this work, we compare Transformer and SSM-based language model backbones for VLMs. We show that Mamba consistently outperforms Transformers in tasks where the output depends on a summary of the visual information. Transformers, on the other hand, maintain the lead in visual grounding tasks, which we link to their ability to learn more accurately and efficiently to retrieve dense information from the context. Regardless, Mamba and SSMs, in general, have memory and computational advantages that could be especially critical for tasks that require modeling long sequences, such as high-resolution images, videos, or multimodal documents. Developing hybrid architectures that integrate an attention-like mechanism into SSMs (Dao and Gu, 2024; Waleffe et al., 2024) is therefore an exciting avenue for future work. Such architectures could lead to efficient VLMs that are also able to effectively retrieve relevant information from the context.

Feature or Bug? Additionally, we experiment with the effect of placing the instruction before and after the visual input. While task-aware image encoding provides a marginal performance boost for Mamba on visual grounding, the results fluctuate across other tasks. Ultimately, we want multimodal models that can seamlessly encode different modalities without forcing a strict order on how they are presented to the model. From this perspective, per-

formance differences due to the input structure are a strong signal that the current iteration of VLMs is only partially addressing this issue.

Limitations

Data Ablations We have not investigated any impact of the data and task distribution. We have not covered any ablations regarding how the examples are packed into sequences. Recent work has shown that this might affect downstream performance in LLMs (Zhao et al., 2024b). Based on our analysis in Section 5.3, and the conclusions from concurrent work (Jelassi et al., 2024; Merrill et al., 2024), we expect that Transformer and Mamba models might behave differently with different packing strategies. However, we want to emphasize that both models are trained on the same data thereby ensuring a fair comparison between them, and also the distribution of the data is heavily skewed towards grounding tasks due to the inclusion of the GRIT dataset.

Ethics statement

The Cost of Training Large Scale VLMs It has been increasingly transparent that the cost of training large-scale models, including VLMs, raises compute barriers (Strubell et al., 2019; Thompson et al., 2020; Luccioni et al., 2024). While patch representations have become the standard approach for encoding images, these representations substantially increase the context window and, consequently, the computational cost of training. To improve efficiency, we have employed sequence packing during training, which results to fewer padding tokens within the batch. Additionally, more sophisticated V&L connectors that downsample the visual sequence (Alayrac et al., 2022; Dai et al., 2024; Laurençon et al., 2024a) can, in principle, accelerate training and inference. We leave comparisons of more efficient V&L connectors in combination with SSMs as future work.

Hallucinations & Reliability A widely acknowledged limitation for LLMs and VLMs is the factuality of the generated content (Ji et al., 2023). The impact of this property can vary depending on the downstream task (e.g., answering a question accurately versus creating novel images with text prompts). Furthermore, prior work (Pantazopoulos et al., 2024), has shown that the visual instruction tuning stage imposes a forgetting effect on the safety guardrails of the backbone LLM leading to more vulnerable VLMs. In this work, we use POPE (Li et al., 2023), a benchmark specifically designed to evaluate object hallucinations in VLMs. However, further investigation is needed to evaluate model hallucinations and improve the reliability of VLMs.

Acknowledgements

References

Amro Abbas, Kushal Tirumala, Dániel Simig, Surya Ganguli, and Ari S Morcos. 2023. Semdedup: Data-efficient learning at web-scale through semantic deduplication. *arXiv preprint arXiv:2303.09540*.

Ekin Akyürek, Bailin Wang, Yoon Kim, and Jacob Andreas. 2024. In-context language learning: Arhitectures and algorithms. *arXiv preprint arXiv:2401.12973*.

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*, 35:23716–23736.

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. 2023. Zoology: Measuring and improving recall in efficient language models. *arXiv preprint arXiv:2312.04927*.

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Re. 2024. On input-dependence and recall in convolutional language models. In *The Twelfth International Conference on Learning Representations*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023a. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023b. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. 2023a. Shikra: Unleashing multimodal llm's referential dialogue magic. *arXiv preprint arXiv:2306.15195*.

Ting Chen, Saurabh Saxena, Lala Li, David J. Fleet, and Geoffrey Hinton. 2022. Pix2seq: A language modeling framework for object detection. In *International Conference on Learning Representations*.

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*.

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*, 36.

Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*.

Tri Dao and Albert Gu. 2024. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*.

Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. 2023. Eva-02: A visual representation for neon genesis. *arXiv preprint arXiv:2303.11331*.

Daniel Y Fu, Tri Dao, Khaled Kamal Saab, Armin W Thomas, Atri Rudra, and Christopher Re. 2023. Hungry hungry hippos: Towards language modeling with state space models. In *The Eleventh International Conference on Learning Representations*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913.

Riccardo Grazzi, Julien Siems, Simon Schrodi, Thomas Brox, and Frank Hutter. 2024. Is mamba capable of in-context learning? *arXiv preprint arXiv:2402.03170*.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Albert Gu, Karan Goel, and Christopher Re. 2022. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*.

Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré. 2021. Combining recurrent, convolutional, and continuous-time models with linear state space layers. *Advances in neural information processing systems*, 34:572–585.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. 2021. Clipscore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7514–7528.

Tao Huang, Xiaohuan Pei, Shan You, Fei Wang, Chen Qian, and Chang Xu. 2024. Localmamba: Visual state space model with windowed selective scan. *arXiv preprint arXiv:2403.09338*.

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/CVF conference on computer vision and pattern recognition*, pages 6700–6709.

Samy Jelassi, David Brandfonbrener, Sham M Kakade, and Eran Malach. 2024. Repeat after me: Transformers are better than state space models at copying. *arXiv preprint arXiv:2402.01032*.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. 2024. Prismatic vlms: Investigating the design space of visually-conditioned language models. *arXiv preprint arXiv:2402.07865*.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR.

14327

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. 2014. Referitgame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 787–798.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. 2016. A diagram is worth a dozen images. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 235–251. Springer.

Mario Michael Krell, Matej Kosec, Sergio P Perez, and Andrew Fitzgibbon. 2021. Efficient sequence packing without cross-contamination: Accelerating large language models without impacting performance. *arXiv preprint arXiv:2107.02027*.

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*, 123:32–73.

Hugo Laurençon, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander Rush, Douwe Kiela, et al. 2024a. Obelics: An open web-scale filtered dataset of interleaved image-text documents. *Advances in Neural Information Processing Systems*, 36.

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

Chunyuan Li, Cliff Wong, Sheng Zhang, Naoto Usuyama, Haotian Liu, Jianwei Yang, Tristan Naumann, Hoifung Poon, and Jianfeng Gao. 2024. Llavamed: Training a large language-and-vision assistant for biomedicine in one day. *Advances in Neural Information Processing Systems*, 36.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. 2023. Evaluating object hallucination in large vision-language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, et al. 2024. Jamba: A hybrid transformer-mamba language model. *arXiv preprint arXiv:2403.19887*.

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 *Computer Vision–ECCV 2014: 13th European Conference, Zurich,* Switzerland, September 6-12, 2014, Proceedings, Part V 13, pages 740–755. Springer.

Fangyu Liu, Guy Emerson, and Nigel Collier. 2023a. Visual spatial reasoning. *Transactions of the Association for Computational Linguistics*, 11:635–651.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2023b. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024a. Visual instruction tuning. *Advances in neural information processing systems*, 36.

Jiarun Liu, Hao Yang, Hong-Yu Zhou, Yan Xi, Lequan Yu, Yizhou Yu, Yong Liang, Guangming Shi, Shaoting Zhang, Hairong Zheng, et al. 2024b. Swin-umamba: Mamba-based unet with imagenet-based pretraining. *arXiv preprint arXiv:2402.03302*.

Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, and Yunfan Liu. 2024c. Vmamba: Visual state space model. *arXiv preprint arXiv:2401.10166*.

Sasha Luccioni, Yacine Jernite, and Emma Strubell. 2024. Power hungry processing: Watts driving the cost of ai deployment? In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '24, page 85–99, New York, NY, USA. Association for Computing Machinery.

Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. 2022. Infographicvqa. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1697–1706.

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

Brandon McKinzie, Zhe Gan, Jean-Philippe Fauconnier, Sam Dodge, Bowen Zhang, Philipp Dufter, Dhruti Shah, Xianzhi Du, Futang Peng, Floris Weers, et al. 2024. Mm1: Methods, analysis & insights from multimodal llm pre-training. *arXiv preprint arXiv:2403.09611*.

William Merrill, Jackson Petty, and Ashish Sabharwal. 2024. The illusion of state in state-space models. *arXiv preprint arXiv:2404.08819*.

Anand Mishra, Shashank Shekhar, Ajeet Kumar Singh, and Anirban Chakraborty. 2019. Ocr-vqa: Visual question answering by reading text in images. In *2019 international conference on document analysis and recognition (ICDAR)*, pages 947–952. IEEE.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. 2022. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*.

Georgios Pantazopoulos, Malvina Nikandrou, Amit Parekh, Bhathiya Hemanthage, Arash Eshghi, Ioannis Konstas, Verena Rieser, Oliver Lemon, and Alessandro Suglia. 2023. Multitask multimodal prompted training for interactive embodied task completion. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 768–789.

Georgios Pantazopoulos, Amit Parekh, Malvina Nikandrou, and Alessandro Suglia. 2024. Learning to see but forgetting to follow: Visual instruction tuning makes llms more prone to jailbreak attacks. In *Proceedings of Safety4ConvAI: The Third Workshop on Safety for Conversational AI@ LREC-COLING 2024*, pages 40–51.

Jongho Park, Jaeseung Park, Zheyang Xiong, Nayoung Lee, Jaewoong Cho, Samet Oymak, Kangwook Lee, and Dimitris Papailiopoulos. 2024. Can mamba learn how to learn? a comparative study on in-context learning tasks. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Leon Derczynski, et al. 2023a. Rwkv: Reinventing rnns for the transformer era. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14048–14077.

Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. 2023b. Kosmos-2: Grounding multimodal large language models to the world. *arXiv preprint arXiv:2306.14824*.

Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. 2023. Hyena hierarchy: Towards larger convolutional language models. In *International Conference on Machine Learning*, pages 28043–28078. PMLR.

Yanyuan Qiao, Zheng Yu, Longteng Guo, Sihan Chen, Zijia Zhao, Mingzhen Sun, Qi Wu, and Jing Liu. 2024. Vl-mamba: Exploring state space models for multimodal learning. *arXiv preprint arXiv:2403.13600*.

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 *International conference on machine learning*, pages 8748–8763. PMLR.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Bavishi Rohan, Elsen Erich, Hawthorne Curtis, Nye Maxwell, Odena Augustus, Somani Arushi, and Taşırlar Sağnak. 2023. Fuyu-8b: A multimodal architecture for ai agents.

Jiacheng Ruan and Suncheng Xiang. 2024. Vm-unet: Vision mamba unet for medical image segmentation. *arXiv preprint arXiv:2402.02491*.

Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. 2022. A-okvqa: A benchmark for visual question answering using world knowledge. In *European Conference on Computer Vision*, pages 146–162. Springer.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565.

Oleksii Sidorov, Ronghang Hu, Marcus Rohrbach, and Amanpreet Singh. 2020. Textcaps: a dataset for image captioning with reading comprehension. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 742–758. Springer.

Jimmy T.H. Smith, Andrew Warrington, and Scott Linderman. 2023. Simplified state space layers for sequence modeling. In *The Eleventh International Conference on Learning Representations*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2021. Long range arena : A benchmark for efficient transformers. In *International Conference on Learning Representations*.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Neil C Thompson, Kristjan Greenewald, Keeheon Lee, and Gabriel F Manso. 2020. The computational limits of deep learning. *arXiv preprint arXiv:2007.05558*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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

Ahmet Üstün, Viraat Aryabumi, Zheng-Xin Yong, Wei-Yin Ko, Daniel D'souza, Gbemileke Onilude, Neel Bhandari, Shivalika Singh, Hui-Lee Ooi, Amr Kayid, et al. 2024. Aya model: An instruction finetuned open-access multilingual language model. *arXiv preprint arXiv:2402.07827*.

Roger Waleffe, Wonmin Byeon, Duncan Riach, Brandon Norick, Vijay Korthikanti, Tri Dao, Albert Gu, Ali Hatamizadeh, Sudhakar Singh, Deepak Narayanan, Garvit Kulshreshtha, Vartika Singh, Jared Casper, Jan Kautz, Mohammad Shoeybi, and Bryan Catanzaro. 2024. An empirical study of mamba-based language models.

Junxiong Wang, Tushaar Gangavarapu, Jing Nathan Yan, and Alexander M Rush. 2024. Mambabyte: Token-free selective state space model. *arXiv preprint arXiv:2401.13660*.

Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In *International Conference on Machine Learning*, pages 23318–23340. PMLR.

Kaiyue Wen, Xingyu Dang, and Kaifeng Lyu. 2024. Rnns are not transformers (yet): The key bottleneck on in-context retrieval. *arXiv preprint arXiv:2402.18510*.

Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Faisal Ahmed, Zicheng Liu, Yumao Lu, and Lijuan Wang. 2022. Unitab: Unifying text and box outputs for grounded vision-language modeling. In *European Conference on Computer Vision*, pages 521–539. Springer.

Haoxuan You, Haotian Zhang, Zhe Gan, Xianzhi Du, Bowen Zhang, Zirui Wang, Liangliang Cao, Shih-Fu Chang, and Yinfei Yang. 2023. Ferret: Refer and ground anything anywhere at any granularity. In *The Twelfth International Conference on Learning Representations*.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. 2023. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11975–11986.

Haotian Zhang, Haoxuan You, Philipp Dufter, Bowen Zhang, Chen Chen, Hong-You Chen, Tsu-Jui Fu, William Yang Wang, Shih-Fu Chang, Zhe Gan, et al. 2024. Ferret-v2: An improved baseline for referring and grounding with large language models. *arXiv preprint arXiv:2404.07973*.

Han Zhao, Min Zhang, Wei Zhao, Pengxiang Ding, Siteng Huang, and Donglin Wang. 2024a. Cobra: Extending mamba to multi-modal large language model for efficient inference. *arXiv preprint arXiv:2403.14520*.

Yu Zhao, Yuanbin Qu, Konrad Staniszewski, Szymon Tworkowski, Wei Liu, Piotr Miłoś, Yuxiang Wu, and Pasquale Minervini. 2024b. Analysing the impact of sequence composition on language model pre-training. *arXiv preprint arXiv:2402.13991*.

Baichuan Zhou, Ying Hu, Xi Weng, Junlong Jia, Jie Luo, Xien Liu, Ji Wu, and Lei Huang. 2024. Tinyllava: A framework of small-scale large multimodal models. *arXiv preprint arXiv:2402.14289*.

Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. 2024a. Vision mamba: Efficient visual representation learning with bidirectional state space model. *arXiv preprint arXiv:2401.09417*.

Yichen Zhu, Minjie Zhu, Ning Liu, Zhicai Ou, Xiaofeng Mou, and Jian Tang. 2024b. Llava-phi: Efficient multi-modal assistant with small language model. *arXiv preprint arXiv:2401.02330*.

Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. 2016. Visual7w: Grounded question answering in images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4995–5004.
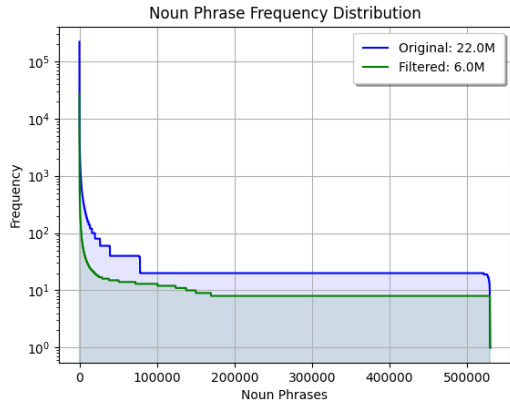
Figure 8: Filtered and unfiltered distribution of noun phrases in GRIT. By applying minimal filtering, we can reduce the dataset size while at the same time maintain object coverage.

A Datasets

A.1 Data Mixture

Table 4 shows the datasets used for instruction tuning. Table 5 shows a detailed breakdown regarding the number of examples for each task.

Filtering Visual Genome We follow OFA (Wang et al., 2022) by preprocessing region descriptions. Specifically, we use only image-region pairs from Visual Genome where the area of the region is smaller than $16,384$ pixels to encourage more fine-grained alignments between vision and language.

Filtering GRIT The original version of GRIT (Peng et al., 2023b) contains 20.5M image-grounded caption pairs. Simply including this benchmark bears the risk of task imbalance, and therefore overfitting on a given task at very early stages of training (Raffel et al., 2020). Furthermore, previous work (Abbas et al., 2023) has shown that semantic deduplication of a large-scale corpus from the web, can significantly reduce the training cost while at the same time maintain performance.

Therefore, to accelerate training without sacrificing diversity we filter GRIT by trying to maximize the number of concepts in the corpus. An easy approach would be to rank the image-text pairs using CLIPScore (Hessel et al., 2021) and then select the top-N images as the filtered corpus. However, this approach may result in selecting images of the most frequent concepts and thus do not expose the model to a variety of examples. For this purpose, we filter the dataset using the noun phrases from each caption. First, we discard all images with

width or height less than 100 pixels. With regards to the text descriptions, we begin by removing any articles from the noun phrase and then counting all phrases for each image. Next, starting from the rarest noun phrases: 1) if the frequency is between a min and a max threshold we add all images to our filtered corpus that contain the phrase in their caption, 2) else if the frequency is higher than the maximum threshold we randomly select max images. As shown in Figure 8, by setting min = 3, and max = 8, we can obtain a smaller corpus that covers all noun phrases.

Filtering OCRVQA We filter out images with a width or height of less than 350 pixels. Additionally, we have observed that some image URLs contain blank images (i.e., images with only a single color). We performed rudimentary filtering by removing all images that have more than 85% pixels from the same color. Finally, we removed all questions associated with the category of the book (e.g, "Is this a sociopolitical book?") as we identified from manual inspection that answering this question based solely from the cover of the book can be particularly challenging.

Multiple Choice VQA For the multiple choice VQA datasets used in instruction tuning (e.g, AI2D (Kembhavi et al., 2016), Visual7W (Zhu et al., 2016), and A-OKVQA (Schwenk et al., 2022)), we have augmented the training data by assigning the correct option to all possible character options. For example, if the question has four candidate answers (A, B, C, D) and the correct answer is A, we created four data points from this question alone by rotating the labels clockwise until the correct answer is in all positions.

A.2 Response Formatting

Table 6 shows the instructions used in our models. Across all experiments, including the first training stage, we mask the instruction prompts and predict only the response. The full sequence given to the model has the following format: $\#\# p_{11}, p_{12}, \ldots, p_{1N} \& \& \ldots, p_{N1}, p_{N2}, \ldots, p_{NN} \#\#$ <Task Instruction> <Prompt> <Response>, where the tokens p_{ij} are the embeddings for each patch.

Representing Coordinates in Images We follow previous VLMs that choose to represent coordinates in images using decimal values (Chen et al., 2023a; Bai et al., 2023b). Other works (Chen

Task	# Packed Samples	Dataset
Captioning	588K	COCO, TexCaps
Chat	157K	LLaVA-Instruct
Dense Captioning	467K	RefCOCO, RefCOCO+, RefCOCOg, Visual Genome
Grounded Captioning	4.2M	GRIT
Image-Text Matching	8k	VSR
Multiple-Choice VQA	127K	AI2D, Visual7W
VQA	352K	VQAv2, GQA, OCR-VQA, VG-QA, DocVQA, InfographicVQA
Visual Grounding	467K	RefCOCO, RefCOCO+, RefCOCOg, Visual Genome
Total	6.2M	

Table 4: Dataset statistics for instruction-tuning. We pack examples from the image into the sequences.

et al., 2022; Wang et al., 2022; Yang et al., 2022; Peng et al., 2023b) introduce special tokens that represent image coordinates in a discrete format. This approach increases the size of the model by adding extra rows to the embedding matrix corresponding to the new special tokens. Furthermore, Shikra (Chen et al., 2023a) has shown preliminary results on the benefits of decimal representation. While there is yet a comprehensive comparison, we believe that the advantage of the decimal representation is due to the fact that the LLM has often already trained embeddings for the decimal tokens, i.e, the model roughly knows what "0.5" refers to and therefore starts from an advantageous point during the visual instruction tuning stage. However, decimal representation introduces longer sequences which prolongs training and inference. Future work could further explore this trade-off.

A.3   Dataset Packing

A significant component during our model development is how we pack the examples into sequences in a meaningful way. The benefits of this dataset packing are two-fold: 1) we ensure efficiency in training by minimizing unnecessary computations due to the padding tokens in a batch (Krell et al., 2021), and 2) by packing examples we facilitate chat capabilities of our models to some degree. In this work, we pack examples from the same image into a sequence of input-output pairs. As already mentioned, we apply packing for all (multiple-choice) VQA, Visual Grounding, and Dense Captioning examples. We refrain from packing captioning examples because the target captions can be repetitive, therefore the model may rely on previous captions without paying attention to the image. We aimed for a maximum sequence length of
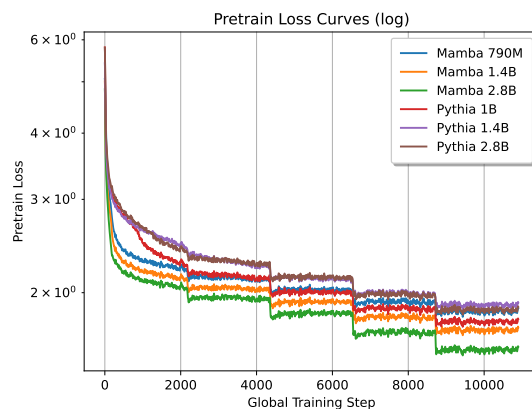

Figure 9: Loss curves for all models during pretraining.

1024 tokens including the patch embeddings and the special image tokens. For this purpose packed examples from VQAv2 (Goyal et al., 2017) are limited to 20 qa pairs. Similarly, we limit the number of qa-pairs to 10 and 5 for the telling, and pointing task in Visual 7W (Zhu et al., 2016). Finally, for all tasks in Visual Genome (Krishna et al., 2017), all packed examples are limited to 10 input-output responses.

B   Training Details

Training Hyperparameters   We use the same hyperparameters as LLaVA-1.5 for pretraining / instruction tuning. We decided to increase the number of epochs in the pretraining stage as we observed significant performance differences after zero-shot evaluation on COCO captioning with preliminary experiments. Additionally, we obtain visual features from the last layer of the EVA model. We have not conducted any ablations considering the layer from which to obtain visual representations. All experiments were conducted using 4x

Dataset	Tasks	# Images	# Packed Samples
AI2D (Kembhavi et al., 2016)	Multiple Choice VQA	3K	44K
A-OKVQA (Schwenk et al., 2022)	Multiple Choice VQA	16K	68K
COCO (Lin et al., 2014)	Captioning	113K	566K
DocVQA (Mathew et al., 2021)	VQA	10K	20K
GQA (Hudson and Manning, 2019)	VQA	87K	72K
GRIT (Peng et al., 2023b)	Grounded Captioning	4M	4M
InfographicVQA (Mathew et al., 2022)	VQA	4K	12K
LLaVA-Instruct (Liu et al., 2024a)	Chat	81K	157K
OCR-VQA (Mishra et al., 2019)	VQA	66K	66K
RefCOCO (Kazemzadeh et al., 2014)	Dense Captioning	16K	16K
	Visual Grounding	16K	16K
RefCOCOg (Kazemzadeh et al., 2014)	Dense Captioning	21K	21K
	Visual Grounding	21K	21K
RefCOCO+ (Kazemzadeh et al., 2014)	Dense Captioning	16K	16K
	Visual Grounding	16K	16K
TextCaps (Sidorov et al., 2020)	Captioning	109K	21K
VQAv2 (Goyal et al., 2017)	VQA	84K	82K
VSR (Liu et al., 2023a)	Image Text Matching	5k	8K
Visual Genome (Krishna et al., 2017)	Dense Captioning	411K	105K
	Visual Grounding	411K	105K
	VQA	184K	97K
Visual7W (Zhu et al., 2016)	Multiple Choice VQA	27K	255K
Total			5.7M

Table 5: Dataset statistics for instruction-tuning. We pack image-text examples from the same dataset into the same sequence.

A100 (40GB / 80GB) or 2x H100 GPUs. For the small models ($\leq$ 1.4B) we set the maximum sequence length to 1024. For the larger models (2.8B) we set the maximum sequence length to 800 to maintain a large batch size during training. Note that this results in a small loss of within-sequence examples. We pretrain each model for 10k steps. We train each model for 100k during the instruction tuning phase, where we evaluate each checkpoint after 10k steps. However, we found that the latest checkpoint resulted in greater performance across both models, despite the higher validation loss.

Model Training Strategy We employ a 'mixed batches' approach, where a batch contains examples from any instruction tuning task. However, we have not used any form of custom sampling e.g adjusting the sampling weight based on the size of the dataset (Raffel et al., 2020). Additionally, we note that the target length can vary significantly per task, for example the correct response to multiple choice VQA is a single token (e.g, the character from the given options), while for captioning examples the target sequence is longer. Therefore, similar to previous work (Üstün et al., 2024; Pantazopoulos et al., 2023), we normalize the cross-entropy loss over the target tokens per sequence first and then average over all the sequences in the batch to weigh all samples equally during finetuning.

Training Logs All training logs regarding pretraining and instruction turing are available here. We also provide here the training curves for the pretraining (Figure 9) and instruction tuning for all

Task	Instruction
Captioning	Provide a one-sentence caption for the provided image
Dense Captioning	Provide a short description of the region
Grounded Captioning	Provide a one-sentence caption for the image and mention each entity.
Image Text Match	Determine if the image matches the description
Multiple Choice VQA	Answer with the option's letter from the given choices directly
Visual Grounding	Locate the region that is described by
Visual Question Answering	Answer the question using a single word or phrase

Table 6: Instructions for all tasks.

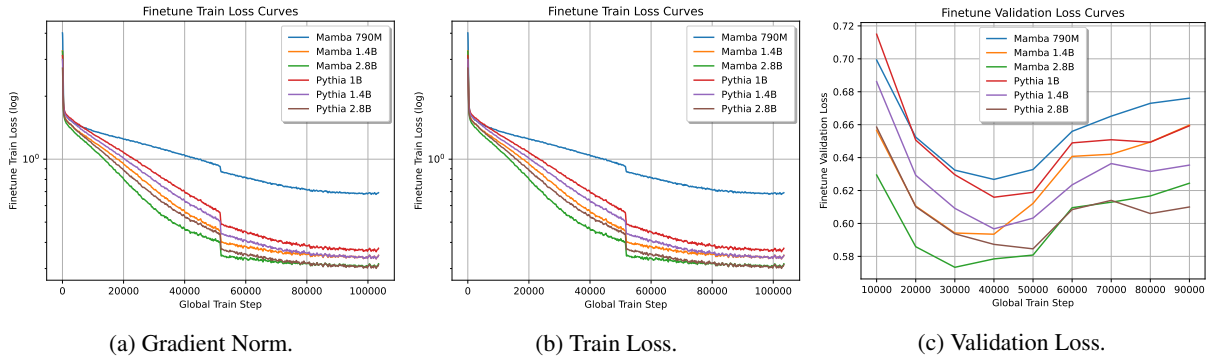| (a) Gradient Norm. | (b) Train Loss. | (c) Validation Loss. |

Figure 10: Performance curves for all models during finetuning.

models (Figure 10). In all of our cases the latest model achieved the best performance despite the trend in the validation loss.

Finetuning on Downstream Tasks In Section 5.2 we also report the results of Pythia-VL and Mamba-VL with 1.4B parameters on VQAv2 and RefCOCOg. For this purpose, we apply a small grid search for each task by using three values for the learning rate ($1e^{-5}$, $5e^{-5}$, $1e^{-4}$) and a batch size of 64. We finetune each checkpoint from the instruction tuning stage for 1 and 3 epochs on VQAv2 and RefCOCOg, respectively, by keeping the examples packed into larger sequences. To increase the resolution of images we simply interpolated the positional embeddings of the vision encoder. We did not scale the rotary embeddings of Pythia, as even in the highest resolution images (560) the sequence length does not exceed the maximum sequence length of the pretrained language model. The models with higher resolution (560) are not using the checkpoint from the finetuning of the previous lower resolution (448). We report the best performing model on the validation split of VQAv2 and test split of RefCOCOg.

C Experiments

C.1 Benchmarks & Metrics

Table 9 shows the benchmarks used for our evaluation with their respective metrics.

C.2 Comparison between EVA-02 and CLIP

We evaluate a Mamba-790M checkpoint after the first training stage using EVA-02 Large 336px/14 (Fang et al., 2023) and CLIP-Large 336px/14 (Radford et al., 2021). We use the same training parameters across both runs. Table 10 illustrates the results on COCO without any fine-tuning. We observe that using visual representations from EVA leads to greater performance.

C.3 Task-agnostic Visual Encoding

We provide the full results showcasing a comparison between task-agnostic and task-aware visual encoding, where the task identity is known to the model before encoding images. Table 11 illustrates the performance for each model with and without task-agnostic visual encoding for all held-in benchmarks. We would like to highlight that a similar comparison has been conducted for Transformer-based VLMs in InstructBLIP (Dai et al., 2024), showcasing that the task-aware visual encoding

14334

DC		Provide a short description of the region [0.50, 0.72, 0.87, 0.89] A rusted junk car with a white R painted on the door [0.24, 0.69, 0.51, 0.87] A rusted truck with '13' spray painted on it. [0.23, 0.68, 0.50, 0.88] The pick-up marked 13
M-VQA		Answer with the option's letter from the given choices directly Question: What color is the closest tent? A: Orange and blue. B: White. C: Black. D: Purple. Answer: A Question: Why is the sand darker at the edge of the ocean? A: It is dirty. B: It is wet. C: It's dark out. D: There's a shadow on it. Answer: B Question: When was this picture taken? A: During the night. B: In daytime. C: At dawn. D: At dusk. Answer: B
VG		Locate the region that is described by Dog's eye is black [0.39, 0.24, 0.41 , 0.27] Black collar on dog [0.45 , 0.24, 0.52 , 0.42] Dog's tail pointing upwards [0.48 , 0.00, 0.56, 0.25] Dog's paw off the ground [0.42 , 0.54, 0.50 , 0.67]
VQA		Answer the question using a single word or phrase Question: Are all the items in the bowl fruits? Answer: Yes Question: What is the light green item? Answer: Apple Question: What is the biggest fruit here Answer: Cantaloupe

Table 7: Illustration of packing examples for each task. Text are the targets for the model for each example.

is beneficial in held-in as well as held-out benchmarks. However, InstructBLIP opts for a specific architectural choice, where the task-aware encoding is conducted at the connector module between the LLM and the vision encoder. The connector (i.e the QFormer), is creating a multimodal prompt that is then prepended to the instruction at the input of the LLM. This means that in practice the LLM sees first the visual prompt and then the instruction. This architectural choice might justify the need for more suitable and versatile multimodal fusion architectures.

C.4 Synthetic Grounding

For the task of synthetic grounding, we create sequences of varying lengths (50/100/200). For each sequence, we created in total 1M training examples and evaluated each model on 100k held-out samples. To eliminate any biases regarding the distribution of the targets, we equally distributed the target token evenly within the sequence. For example, for sequences with 100 tokens, 1% of the training examples (1000) have the 1st token as target. All models are trained using a global batch size of 64 for 78K steps. We evaluated every model after 1% of training steps to capture precisely the timestep where each model learns the task.

14335

Hyperparameter	Pretraining	Instruction Tuning
global batch size	256	128
lr	1e-3	2e-5
lr schedule	cosine decay	
lr warmup	0.03	
number of epochs	5	2
optimizer	AdamW	
DeepSpeed Stage	2	3

Table 8: Hyperparameters during both training stages. The same hyperparameters are used for Pythia-VL, and Mamba-VL across all three different scales.

Benchmark	Zero-shot	Metrics
COCO	✗	CIDEr (C), BLEU-4 (B4), METEOR (M), ROUGE (S), Spice (S)
NoCaps	✓	CIDEr
VQAv2	✗	VQAv2 score
GQA	✗	Accuracy
Visual7W (T)	✗	Accuracy (Multiple Choice)
VSR	✗	Accuracy
POPE	✓	Accuracy
RefCOCO /g/+	✓	Accuracy@IoU≥ 0.5
TextCaps	✓	CIDEr
TextVQA	✓	Accuracy
AI2D	✓	Accuracy (Multiple Choice)

Table 9: Evaluation metrics for each benchmark.

Model	C	B4	M	R	S
Mamba-CLIP	79.9	21.8	22.2	47.9	16.5
Mamba-EVA02	**87.1**	**23.7**	**23.2**	**48.8**	**17.9**

Table 10: Performance of Mamba-790M on COCO test after the first training stage using similar sized CLIP and EVA02 models. Across the board, Mamba achieves greater performance when paired with EVA.

Relation to Induction Heads Our task is closely related to the Induction Heads (Olsson et al., 2022), which requires models to perform associative recall by retrieving relevant information from the memory. More specifically, if the model has already observed the pattern AB in a sequence of tokens, then it should be able to infer that A is followed by B some time within the same sequence.

The results of Mamba (Gu and Dao, 2023) on Induction Heads show that a two-layer model trained on short sequences maintains high performance across varying sequence lengths compared to other SSMs and Transformer recipes. A key difference between this setup and how we framed our syn-

thetic grounding task is that in the Induction Heads benchmark there exists a single special token in a sequence and the model always needs to predict the follow-up token (e.g, Input: a b c d e ⊢ f g h i ... x y z ⊢, Output: f (Fu et al., 2023)). On the other hand, in our task every token in the sequence is a "special token", and the model needs to be able to recall every element in order but also perform a two-hop reasoning between two embedding spaces.

C.4.1 Prefix Variation

Additionally, motivated by the improvements of the task-aware encoding on visual grounding, we experiment with a prefix variant of our synthetic task. The key difference is that the query precedes the input sequence, and therefore, Mamba has direct access to the required information from the beginning. We experiment with the same sequence lengths for Pythia and Mamba. Figure 12 illustrates the performance of both models. Compared to the suffix variant (Figure 7) we can see that Mamba learns the task significantly faster. For example, in the suffix version of the task and for sequences of 200 tokens, Mamba is not able to reach 95% accuracy in the training window. On the other hand, in the prefix setting and for the same sequence length, we observe that Mamba learns the task within the first half of the training. Nevertheless, even on this setup, Pythia is more efficient as it learns the task within only 10% of the training steps.

Model	Task Agnostic	Image Captioning	General VQA			Misc	Visual Grounding						Reading Comprehension	
		COCO	VQAv2	GQA	V7W	VSR	RefCOCO		RefCOCO+		RefCOCOg	V7W (P)	TextCaps	AI2D
		test	val	test-dev	test-T	test	testA	test B	testA	testB	test-P	test	val	test
Pythia-VL	✓	134.06	73.57	57.05	83.06	77.72	82.43	68.39	72.35	55.16	72.56	86.13	94.60	79.27
Pythia-VL	✗	133.87	**73.15**	58.12	79.30	76.94	82.78	68.89	71.74	54.44	73.76	85.41	95.03	79.83
Mamba-VL	✓	134.76	74.46	58.44	83.78	80.18	76.60	63.48	68.40	52.11	68.82	80.18	98.68	80.20
Mamba-VL	✗	135.45	**74.58**	58.32	83.19	79.54	77.77	65.35	68.25	51.79	70.01	77.04	100.2	80.86
Relative Performance Gain Per Task														
Pythia-VL	-	-0.14	-0.57	+1.88	-4.53	-1.00	+0.42	+0.73	-0.84	-1.31	+1.10	-0.84	-0.45	-0.70
Mamba-VL	-	+0.51	+0.16	-0.21	-0.70	-0.80	+1.53	+2.95	-0.22	-0.61	+1.73	+3.92	+1.54	-0.82

Table 11: Comparison of Pythia-VL & Mamba-VL with task-agnostic and task-aware visual encoding.

(a) Sequence length = 50.

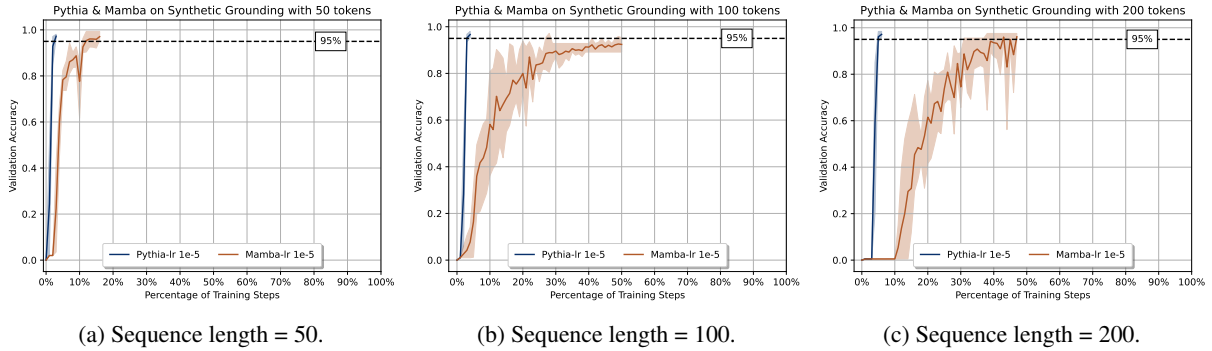(b) Sequence length = 100.

(c) Sequence length = 200.

Figure 11: Performance curves for Pythia-1.4B and Mamba-1.4B variants on the synthetic grounding task with varying sequence length and the prefix modification.
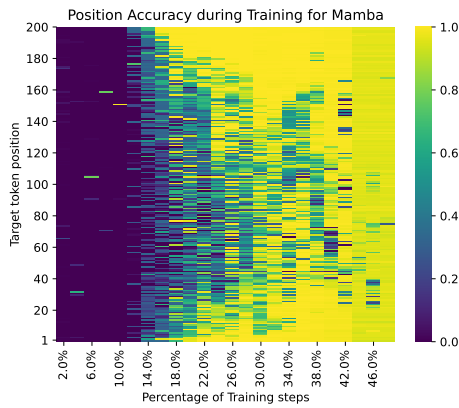

Figure 12: Accuracy per position on the held-out set during training on sequences of 200 tokens the prefix synthetic grounding task with the prefix modification.