Mamba-Based Approaches to the Hateful Memes Challenge

Michael Liu, Prajwal Mohan Kumar, Aryan Mittal, Eric Yeon
Georgia Institute of Technology
Atlanta, Georgia, USA
mliu611,pkumar352,aryan.mittal,syeon31@gatech.edu

Abstract

Our work explores several modeling challenges presented by the application of Mamba, or selective state space models, to the multimodal classification problem of identifying hateful memes. The existing baselines and best performing models on the problem all leverage transformer backbones, which have computational and memory requirements that are quadratic with respect to sequence length. In contrast, Mamba architectures offer linear scaling of computation and memory during sequence modeling. We integrate a foundational Mamba model pretrained on text and a new Mamba variant, VMamba, pretrained on images, in a variety of architectures aimed at the hateful memes classification problem. In our experiments, we focus exclusively on accuracy and AUC as our metrics of choice, excluding computational and memory comparisons to benchmark models due to compute constraints. Setting aside model ensembles that achieve SoTA performance, we find that our text-only and multimodal models perform slightly worse than their transformer counterparts, while our image-only model performs better than its ResNet-based counterparts.

1. Introduction/Background/Motivation

1.1. The Hateful Memes Challenge

Our goal is to classify multimodal hateful memes using the novel architecture, selective state space models, also known as Mamba. Teasing out the meaning of a multimodal meme requires subtle reasoning and integration of semantic clues from both the image and text, making it a strong data source to test out the multimodal fine-tuning capabilities of text and image models. Classifying multimodal content (hate speech, LLM-generated, etc.) also grows more important each day as the ability to generate undesirable content is made easier by constantly improving deep learning tools.

The Mamba architecture is designed to model sequences much longer than traditional transformers can handle due to their quadratic time and memory complexity with respect to sequence length. On a variety of language modeling and long-range benchmarks, Mamba has performed comparable to or better than strong transformer recipes of similar size [9]. Given that transformer models have come to dominate almost all deep learning tasks, we explore the broad applicability of a model framework, Mamba, that operates in linear time complexity. The specification of the hateful memes classification problem doesn't necessarily play to all of the studied advantages of Mamba (e.g., long-range dependencies), but we nonetheless are interested in the model's performance given a variety of modeling challenges that aren't yet well studied: (a) applying Mamba to discriminative tasks, (b) fine-tuning Mamba, and (c) adapting Mamba to the multimodal context.

Our goal is compare the accuracy of models using Mamba backbones to similar unimodal and multimodal baselines highlighted in [16]. Due to compute constraints, we won't dive into time complexity comparisons between benchmarks and our models as that would involve rerunning the benchmarks. We also don't expect to compete with the best performance achieved on the dataset, given that SoTA approaches tend to ensemble results from a variety of models for stability [4]. We did not perform ensemble experiments given the same compute constraints previously mentioned.

The Hateful Memes dataset was curated by the FAIR team at Meta from various internet forums and social media sites [16]. To overcome licensing issues, FAIR had human annotators study the source material and create derivative memes. The dataset contains over 10,000 data samples of these derivative memes, open-sourced for further research against hate speech identification. Each sample contains the image file and its extracted text caption. Note that we further masked out the text captions in each image and inpainted the results using OpenCV (more sophisticated deep learning inpainters are also available, but weren't used) as suggested by the Hateful Memes Challenge winners. Finally, each meme has a binary label indicating whether its content is hateful or not. See examples in Figure 1 for the semantic nuance across modalities that makes a meme hate-

ful.

1.2. Current approaches

Traditional methods of hate speech identification often rely on text-based data and in recent years, transformer-based encoders like BERT [3] have become the SoTA NLP models known for their ability to comprehend context and semantics within textual data. Furthermore, the semantic relationships that can exist between text and image in multimodal data has led to the development of multimodal models such as BERT+ResNet [3, 13] or VisualBERT [17] where features from images and text interact through a cross-attention mechanism. The best-in-class performances of the dataset are all derived from foundational models with transformer backbones. Multiple foundational models are usually ensembled to drive the best performance.

Existing limitations of transformers largely tie to the quadratic time and memory complexity of the attention mechanism with respect to sequence length. Pairwise attention must be computed between each input token in the sequence. As a result, transformers struggle with long-range time dependencies, but also suffer from computationally intensive training and inference more broadly. In addition, the positional encoding of tokens can inadvertently affect the accuracy of information traveling through the model and this becomes an evident issue for domain-specific language tasks.

1.3. Applying Selective State Space Models (Mamba)

Mamba presents an alternative approach to modeling sequences that provides computational benefits and maintains longer-range time dependencies in the data. It innovates on its predecessor, S4, a sequence-to-sequence model [12]. S4 projects a 1-dimensional signal, $x(t)$ to a 1-dimensional output $y(t)$ via an n-dimensional hidden state, $h(t)$. $h(t)$ is formulated as a polynomial approximation of the entire signal history at a given time step t. The hidden state and output are defined via the differential equations below (1a,1b), where A, B, C, D are learnable time-invariant matrices. Further structure is imposed on A as described in [11]. D is a trivial skip connection.

$$h(t) = Ah(t) + Bx(t) \tag{1a}$$
$$y(t) = Ch(t) + Dx(t) \tag{1b}$$

While not covered in detail here, an additional time-invariant parameter Δ helps transform continuous parameters A, B to discrete parameters $\overline{A}, \overline{B}$. This results in equation 2a and 2b:

(a) Hateful.

(b) Not hateful.

Figure 1: Examples of memes from the dataset.

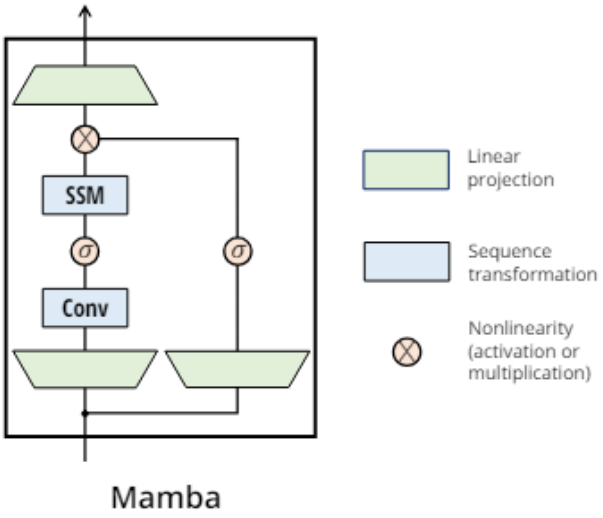

Figure 2: Baseline Mamba module

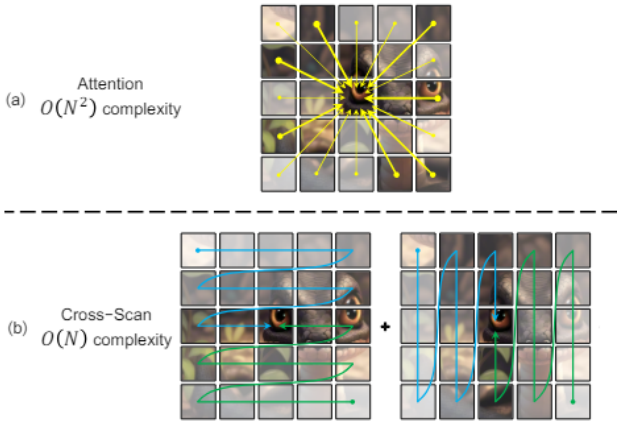

Figure 3: Comparing Attention and Cross-Scan

$$h_t = \overline{A}h_{t-1} + \overline{B}x_t \tag{2a}$$
$$y_t = Ch_t + Dx_t \tag{2b}$$

Through some algebraic conveniences, this model structure allows the output $y(t)$ to be computed as a global convolution during training, bypassing the time-complexity of computing the hidden state $h(t)$, while still allowing recurrent computation during inference. However, whereas an attention mechanism doesn't compress memory at all and results in in a quadratic time complexity with respect to sequence length, S4 sacrifices any form of selective memory to achieve its linear time complexity. S4 instead compresses the memory with respect to the same measure throughout the sequence.

To give S4 more flexibility, Mamba implements a selective scan mechanism [9]. Matrices B, C and parameter Δ are re-parameterized as functions of the input x_t, rather than as time-invariant. Though $y(t)$ can no longer be computed via global convolution, further hardware-aware algorithm improvements are made to recover the linear time-complexity benefits of S4. The final Mamba module is visualized in Figure 2. The module is a merger between an H3 module [7] and a standard MLP. The language-based Mamba model in our experiments is pretrained on the Pile dataset [8] and available publicly on Github [10].

Notably, Mamba is designed for unidirectional sequences, making text data a strong fit for its modeling capabilities. However, image data has no inherent direction in its 2D input, meaning that adaptations to Mamba are needed to process images for modeling tasks. Research on applying Mamba to image data is still in its early stages, but we test a publicly available image encoder, VMamba, which is pretrained on ImageNet [5]. VMamba adapts the Mamba module to image data through its 2D-selective-scan mechanism, Cross-Scan [18]. Cross-Scan traverses the image patches in 4 different ways (see figure 2): (1) starting from the top left corner, traversing left to right, (2) top left corner, top to bottom, (3) bottom right corner, right to left, and (4) bottom right corner, bottom to top. Whereas attention computes pair-wise attention between each patch, VMamba breaks down the patches into four sequences which are then passed into separate Mamba modules, and finally reconstructed into an output map. This maintains the linear time complexity of processing sequences while still allowing each pixel to learn from all others in the image. In addition, given the sequence length of images, we should benefit from Mamba's ability to model long-range dependencies.

Armed with pretrained Mamba modules trained on text and image, we can now specify the custom Mamba architectures we've designed to tackle the hateful memes classification problem.

2. Approach

2.1. Architecture

We tested 4 total architecture backbones inspired by the benchmarks in the original paper. First, we fine-tuned a single-directional Mamba pre-trained on language generation tasks named **TextMamba**. Second, based on the standardized practice of using bi-directional sequential models (such as with RNNs and LSTMs) on classification tasks, we fine-tuned a bi-directional Mamba model named **BiDTextMamba**. Third we leveraged a pre-trained vison-based Mamba architecture, VMamba, as an image encoder, labeled **ImageMamba**. Finally, we concatenated hidden states from TextMamba and ImageMamba to form **Concat-**

Mamba. See Section 1.3 (or the readme in our attached code) for more details on the pretrained Mamba modules used in our architectures and other code samples we used.

While there hasn't been rigorous testing on the best way to design a classifier head for state space models to our knowledge, we can leverage successful methods applied to prior popularized sequential models (RNNs, LSTMs). For all four models, we mean-pooled output states to pass to the classifier head. The classifier head for our four models consisted of two linear layers connected by a ReLU activation, and a final sigmoid activation to generate class probabilities. While transformer-based foundational models typically do not include non-linear activation layers in their classification head, we found it mildly improved performance while testing our models. However, adding greater depth and complexity to the MLP tended to overfit the model so we kept the structure fairly simple, reducing the pooled output state to 64 dimensions before passing through a ReLU and running the data through a classifier layer.

2.2. Training design

Fine-tuning Mamba models also lacks a robust literature. Without the compute to fully re-train our foundational models, we attempted to implement the Parameter Efficient Fine Tuning (PEFT) framework, Low-Rank Adaptation (LoRA) [15], where we freeze model parameters while introducing a trainable low-rank approximation AB of each linear projection matrix $W_0 \in \mathbb{R}^{d\times k}$ in our model. The output of each projection operation becomes $h = W_0x + BAx$ where we tune the matrices $B \in \mathbb{R}^{d\times r}$ and $A \in \mathbb{R}^{r\times k}$. BA is scaled by $\frac{\alpha}{r}$ where α is a constant in r. We use default hyperparameters as specified in the LoRA configuration on Hugging Face [6], such as $r = 8$ and $\alpha = 8$. However, we increase dropout to 0.1. Note that the default configuration leads to a greater percentage of parameters trainable, although that is partially driven by the difference in the rank hyperparameter r (see Table 1). Bert uses values of $r = 1$, but we found lowering our target rank to be detrimental. Given LoRA was developed with Transformers in mind, a new tuning method may be needed to work better with Mamba.

Model	Parameters(M)		
	Trainable	Total	%
BERT	0.04	100	0.04%
TextMamba	4	375	1.06%
BiDTextMamba	8	751	1.06%
ImageMamba	1	89	1.51%
BiDConcatMamba	13	844	1.55%

Table 1: Trainable parameters under LoRA framework

We leveraged the hyperparameters in Table 2 for training. We specified a batch size of 32 based on the GPU ram available to us. We also only needed a short sequence

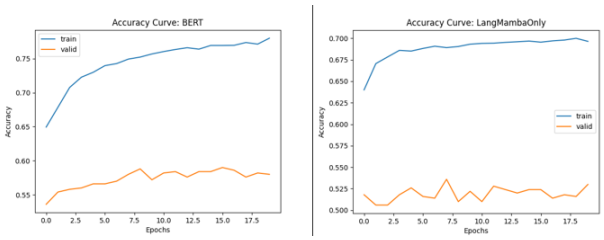

Figure 4: Comparison of BERT and TextMamba training runs

max length given the text in memes is typically a short sequence of words. Specifying the optimizer parameters was difficult and ultimately we do not think our application of LoRA may be the best way to fine-tune Mamba architectures. Heuristically, what seemed to work best was choosing a relatively high learning rate of 1e-3 that then is exponentially decayed by a gamma parameter of 0.8. However, no matter the configuration we tested, we found that while training loss and accuracy steadily improved, the validation loss and accuracy were extremely choppy across Mamba architectures and hyperparameters. Even after lowering the learning rate significantly and adding weight decay, we didn't see much improvement. A sample chart for one of TextMamba training runs is specified in Figure 3, which we can compare to the much smoother BERT training run.

Our models still reached reasonable validation accuracy, but given our focus is on applying Mamba models rather than explicitly testing the procedures involved in fine-tuning the model (optimization, architecture design for classification, etc.), it is difficult to diagnose exactly what is causing what appears to be overfitting of the data. Our best hypothesis is that a better designed PEFT framework for Mamba may limit the overfitting when fine-tuning Mamba and lead to smoother training.

Batch size	32
Sequence max length	50 tokens
Epochs	20
Learning rate	1e-3
Gamma	0.8
Weight decay	1e-4
Sequence max length	50 tokens

Table 2: Training Hyperparameters

2.3. Uncertainties and challenges

There were a number of uncertainties in testing these architectures. First, there have not been many reported results of fine-tuning Mamba models. A Kaggle challenge

around detecting long-form LLM-created text saw a participant ensemble a text Mamba model with more traditional transformer-based text encoders for the classification task, but ultimately suffer in testing performance when including the Mamba backbone [2]. As mentioned prior, given compute constraints, we tried to implement the Parameter Efficient Fine Tuning (PEFT) framework, LoRA, but there hasn't been empirical evidence that LoRA is the right method to fine-tune Mamba architectures.

There also is not a prescribed design for the classification head of Mamba models so we rely on mean-pooling methods designed for other prior popularized models like RNNs, LSTMs, and Transformers. The Mamba author suggested a CLS token may also be productive [14], but this is not something we tested.

Finally, there are only a few explorations of how Mamba architectures are best set up in multimodal settings. One paper suggests concatenating an image embedding from a transformer encoder onto each text token embedding to be fed into the Mamba backbone [22], but we did not ultimately get to this. For our purposes, we tried a simple concatenation of output states from separate text and image Mamba models before feeding it into our classifier head. However, it is unclear if concatenation offers the best integration of learning from multiple data modes.

3. Experiments and Results

As mentioned, we focused primarily on accuracy in our application of Mamba and comparison against benchmarks. Our final models were selected on the basis of best validation accuracy and then run on the test dataset. Note that the hateful memes dataset has two different sets of validation and test data: "seen" and "unseen". We leverage the "seen" set which is what is used in the original paper [16]. The "unseen" dataset is leveraged for the competition run by Meta alongside the paper publication. Table 3 compares our models in bold with the original benchmarks in italics.

Model		**Val.**		**Test**	
		Acc.	AUC	Acc.	AUC
Unimodal (text)	*BERT*	0.58	0.65	0.63	0.69
	TextM.	0.58	0.59	0.59	0.62
	BiDTextM.	0.56	0.61	0.58	0.64
Unimodal (image)	*Img_Grid*	0.51	0.52	0.53	0.54
	Img_Region	0.53	0.57	0.52	0.58
	ImgM.	0.55	0.54	0.53	0.53
Multimodal (unimodal training)	*Late Fusion*	0.59	0.65	0.63	0.69
	CC_BERT	0.59	0.66	0.62	0.68
	CC_M.	0.59	0.63	0.60	0.59
(multimodal training)	*VilBERT*	0.66	0.73	0.66	0.75
	Vis.BERT	0.66	0.74	0.69	0.75

Table 3: Model performance

Looking at our results, the first broad observation we can make is that the testing set seems be a lot more similar to the training set than our validation set. While there is not necessarily a better metric to select our best model than validation accuracy (or loss), it is a possibility that there is a better model for our testing set in our training history. We did not have the storage capabilities to test this.

Looking at the unimodal text-only results, we see that Mamba performs as well as BERT on the validation set, but does not translate as well to the test set. As mentioned, the brevity of meme text does not necessarily emphasize the long-range-dependency strength of the mamba model. In addition, we are not sure of the empirical validity of the LoRA PEFT framework we adapted. Finally, BERT and our text Mamba are pre-trained on different text datasets. To get a better comparison, we may want to pretrain both architectures on the same text data. However, given all this uncertainty, it is encouraging to see similar results in the text modality, especially given the subtle semantic nuances in hateful meme text.

One interesting result we see is that BiD_TextMamba didn't necessarily perform as well on the dataset. While the bidirectional model approach hasn't been formally tested to our knowledge, one possibility why it is not as productive here is that the sequences are relativey short. Perhaps for a paragraph or even larger body of text, the bidirectional information may provide more expressivity. But in our simple use case, that expressivity may make it too easy to overfit our data. A possible area for further exploration may be to use lighter weight Mamba variants in the bidirectional model.

It is also important to note that the text Mamba is designed directly as a sequence-to-sequence model and does not have the same encoder-decoder division that other popular deep learning models have. Hence, while we can use Mamba as a way to generate text sequence embeddings, it is likely going going to be much a much larger model and more expressive by design than a text encoder like BERT. Perhaps this difference is driving the overfitting that Mamba seems to demonstrate off the bat, although, as discussed, there are many other facets of our model design that still leave room for improvement.

In the unimodal image-only setting, the Mamba results are rather encouraging. The VMamba backbone outperforms the ResNet backbone baselines referenced in the paper, likely due to the global receptive field for each pixel and flexible Cross-Scan mechanism to facilitate deeper integration of features and longer-range dependencies in each image. It would have been interesting to compare the results of VMamba against a ViTransformer backbone to compare the relative effectiveness of the selective scan against attention, but overall, the results are encouraging for Vmamba.

In the multimodal setting with unimodally trained modules, our models perform as well as benchmarks on the validation set, but again do not translate as well to the test set. Given the text and image modalities are trained independently, it may be the worse inferential capabilities of our text Mamba module (whose results similarly does not translate as well to the test set) that weaken our results.

Note that we did not test any multimodally trained Mamba benchmarks. While there are papers starting to emerge that test the paradigm [22], we did not find a usable pretrained model for our tests and were beginning to run past our set budget for compute. The concept of multimodally trained Mamba is still in question as there isn't a clean analogy for the cross-attention mechanism that is leveraged in models like VilBERT and VisualBERT [19, 17]. The primary suggestion to date has been to use an image encoder (e.g., ViT, VMamba) to produce image embeddings which are then concatenated onto each text token that is fed into the Mamba module.

4. Conclusion

Our research aimed to explore the application of Mamba-based models in the classification of hateful memes, specifically leveraging the computational efficiently of linear complexity and potential for handling long-range dependencies of Mamba architectures (in contrast to the quadratic complexity of existing traditional transformer-based models).

In conclusion, we observe promising results from our Mamba-based approaches to the Hateful Memes challenge, though we do not obtain any particularly impressive results with respect to the given baselines.

In particular, we observe that although Mamba-based models can perform comparably to the transformer models given unimodal settings (especially regarding text-data), there is not a consistent outperformance in comparison to the existing benchmarks given multimodel contexts. Moreover examining the text-specific and image-specific Mamba models, we saw a similar validation accuracy to the BERT models with a slight dip in accuracy performance on the testing dataset for the text-specific Mamba model, and in terms of the image-only VMamba model, there was an observed outperformance in comparison to the ResNet-based models. Discrepancies to our inital inferences were seen in the bidirectional Mamba model which did not yield accuracy improvement as initially anticipated (however as previously menttioned, this may be due to the lack of longevity in sequence length in mem text which consequently does not take advantage of the gains in bidirectional processing). It was also seen that our attempt at conglomerating the text and image features did not result in our model surpassing the accuracy performance of the traditional transformer-based models (which may simply be akin to hurdles in multimodel integration in existing Mamba architectures).

As for improvements to this work, we emphasize that future work solve the issue of overfitting on the dataset by either freezing more parameters or developing a PEFT framework specifically for Mamba, the latter being more broadly applicable to more general research thrusts. Moreover, in the absence of a compute bottleneck, we encourage the finetuning of a multimodally-pretrained visual question answering Mamba model, such as VL-Mamba or Cobra [20, 22], to detect whether memes are hateful. While both of these models are designed to give text responses to a given prompt, we suspect that the LLM final layer can be replaced with a Linear (or similar) layer whose weights are finetuned to classify a meme as hateful or not. to a multimodal prompt of a meme and the text "Is this meme mean?".

Moreover, we feel it may be an interesting task to visualize both the salient regions of input meme images and salient words in input meme text to analyze where the models we evaluate in this paper place their "focus" when classifying a meme as hateful or not. For this reason, we suggest the application of Grad-CAM [21] or the methods proposed in [1] to the models we propose in this paper to further assess the performance, from an explainability standpoint, of Mamba-based methods versus the previous benchmark models.

References

[1] Ameen Ali, Itamar Zimerman, and Lior Wolf. The hidden attention of mamba models, 2024. 6

[2] James Day. Llm - detect ai generated text, 5th place solution: 1.7 million training examples + domain adaptation. https://www.kaggle.com/competitions/llm-detect-ai-generated-text/discussion/470093, 2024. 5

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, May 2019. arXiv:1810.04805 [cs]. 2

[4] DrivenData. Hateful memes challenge winning submissions. https://github.com/drivendataorg/hateful-memes, 2020. 1

[5] Liu et al. Vmamba: Visual state space model. https://github.com/MzeroMiko/VMamba, 2024. 3

[6] Hugging Face. Lora conceptual guide. https://huggingface.co/docs/peft/main/en/conceptual_guides/lora, 2024. 4

[7] Daniel Y. Fu, Tri Dao, Khaled K. Saab, Armin W. Thomas, Atri Rudra, and Christopher Ré. Hungry hungry hippos: Towards language modeling with state space models, 2023. 3

[8] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling, 2020. 3

[9] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2023. 1, 3

[10] Albert Gu and Tri Dao. Mamba. https://github.com/state-spaces/mamba/tree/main, 2024. 3

[11] Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Re. Hippo: Recurrent memory with optimal polynomial projections, 2020. 2

[12] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces, 2022. 2

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition, Dec. 2015. arXiv:1512.03385 [cs]. 2

[14] Albert Hu. Training a text classifier on mamba. https://github.com/state-spaces/mamba/issues/163, 2024. 5

[15] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. 4

[16] Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. The Hateful Memes Challenge: Detecting Hate Speech in Multimodal Memes, Apr. 2021. arXiv:2005.04790 [cs]. 1, 5

[17] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. VisualBERT: A Simple and Performant Baseline for Vision and Language, Aug. 2019. arXiv:1908.03557 [cs]. 2, 6

[18] Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, and Yunfan Liu. Vmamba: Visual state space model, 2024. 3

[19] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks, Aug. 2019. arXiv:1908.02265 [cs]. 6

[20] Yanyuan Qiao, Zheng Yu, Longteng Guo, Sihan Chen, Zijia Zhao, Mingzhen Sun, Qi Wu, and Jing Liu. Vl-mamba: Exploring state space models for multimodal learning, 2024. 6

[21] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization, 2019. 6

[22] Han Zhao, Min Zhang, Wei Zhao, Pengxiang Ding, Siteng Huang, and Donglin Wang. Cobra: Extending mamba to multi-modal large language model for efficient inference, 2024. 5, 6