CS 7643 Project Report: Assessing the performance and robustness of newer Vision-Language Models on the Hateful Memes Challenge

Ke Xin Chong Michelle Lim Wei Hong Low Qin Wayne Toh
Georgia Institute of Technology
{kchong43, mlim92, wlow7, qtoh3}@gatech.edu

Abstract

Older vision-language models that utilize pre-trained convolution based object detectors are still considered state of the art approaches in Facebook AI's The Hateful Memes Challenge [3] despite the shift towards transformer-based architectures. Motivated by this shift and the significant computational expense incurred by object detectors in the older models [5], we aim to explore newer and faster transformer-based vision-language models and assess their base performance on The Hateful Memes Challenge. In this study, we highlight CLIP's superiority over BLIP and ViLT in meme classification tasks, attributing this to CLIP's larger and more diverse training dataset and its ability to learn representations tailored specifically for multimodal classification. CLIP's implicit integration of information from both modalities through joint training proves advantageous, resulting in more robust representations of image-text pairs. Conversely, BLIP's reliance on pre-trained weights may limit its effectiveness in capturing nuanced multimodal relationships. ViLT's use of a shared transformer to conduct image and text encoding contributes towards its weaker performance. These findings underscore the significance of CLIP's multimodal capabilities in real-world applications. Additionally, we also highlight the vulnerability of these models to different adversarial attacks, and note that performing adversarial training can be beneficial to both model robustness and improved model generalizability.

1. Introduction/Background/Motivation

Memes wield immense influence on social media, but their potential for harm necessitates proactive screening. While humans excel at discerning hateful content, AI-based approaches face challenges due to the multimodal nature of memes as well as the contextual complexity required to understand the intent and content of memes.

The Hateful Memes Challenge, launched by Facebook AI [4], aims to determine whether machine learning models, devoid of preconceptions and cultural biases, can match human proficiency in identifying memes containing hateful content. Affirmative results could enhance online safety for marginalized communities and mitigate the influence of hateful content on susceptible youth.

In the context of the Hateful Memes Challenge, our project aims to explore recent advancements in Deep Learning, particularly in multi-modal modeling, and assess their efficacy in addressing the challenge. Notably, we observe that leading teams in the Hateful Memes Challenge utilize multi-modal deep learning techniques that integrate pre-trained convolutional object detectors to extract plausibly relevant crops of the image before feeding these crops into cross-modal Transformer [3]. Architectures such as Ernie-ViL, VisualBERT, and UNITER follow this approach, employing object detection methods like Faster-RCNN to extract regions of interest[3].

Our investigation shifts focus to newer architectures like CLIP[9], BLIP[8], and ViLT[5], which forego pre-trained object detectors in favor of patch projection schemes such as Vision Transformers (ViTs)[5], thereby streamlining model complexity and inference speed.

We aim to reproduce and establish performance baselines for these models, with slight architectural modifications for potential performance enhancements. Despite the challenge posed by achieving state-of-the-art results[3], which are often accomplished through *ensembling* methods, we aspire to surpass previous benchmarks.

Additionally, we intend to assess the robustness of these architectures against adversarial attacks[11], recognizing the potential vulnerability of models to malicious manipulation, which could undermine their effectiveness in fostering online safety over time.

For this work, we utilize the dataset created by Facebook AI for the Hateful Memes Challenge, which consists of 12,140 hateful and non-hateful memes, where each meme consists of an image-caption pair [4]. An example of a non-hateful meme from the dataset can be found in Figure 1.

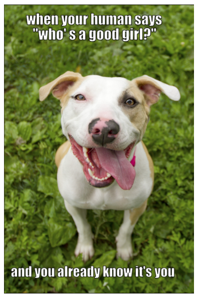

Figure 1. An example of a non-hateful meme present inside the Hateful Memes Challenge Dataset

The memes were created from scratch, with annotations overlaid over pre-existing images from the internet by trained annotators from a third-party annotation company hired by Facebook [4]. The dataset is already split in a train-dev-test format, and we utilize the train and dev sets for training and validation, and use the test set for assessment of our trained models.

2. Approach

Our approach involves leveraging pre-trained CLIP, BLIP, and ViLT models from HuggingFace and fine-tuning them using the Hateful Memes Challenge dataset. Additionally, we experiment with various architectural modifications to enhance their performance on the challenge's test dataset. Following the training process, we subject our models to adversarial attacks to generate adversarial examples on the development and test sets of the Hateful Memes Challenge dataset. Subsequently, we evaluate the robustness of our best-performing model on these generated datasets to gain insights into its resilience.

2.1. Models

In this study, we note that the problem is of a multimodal nature, hence necessitating a model that is able to properly combine both the image and caption of the memes together to produce appropriate feature outputs that are effective for meme classification. CLIP and BLIP accomplish this via the use of separate image and text encoders, whilst ViLT tries to combine the encoding of the image and text in a single transformer encoder. The below subsections go to further explain the individual models we used for the study and their unique features that lend them suitability for this work.

2.1.1 CLIP

CLIP is a multimodal model developed by OpenAI, and it stands for 'Contrastive Language-Image Pre-Training'[9]. CLIP is trained on image to text pairs, and is pre-trained in a way whereby real image to text pairs have the cosine similarity between their encodings maximized, while incorrect image to text pairings have their cosine similarity minimized [9].

The Transformer architecture proposed by Vaswani et al [10] is used for CLIP's text encoder, while the Visual Transformer architecture [2] is used for CLIP's image encoder. Cosine similarity between the image and text encodings are used to obtain the model's output logits. The CLIP model has been demonstrated to be highly adept at many tasks even with zero-shot learning, such as ImageNet class prediction[9].

2.1.2 BLIP and BLIP-2

BLIP (Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation) is a vision-language model that leverages on three pre-training objectives, namely image-text contrastive loss, image-text matching loss and language modelling loss to train three different parts of its architecture, namely its unimodal encoder which separately encodes text and images, image-grounded text encoder which utilizes cross-attention to inject visual information from images, and image-grounded text decoder respectively [8].

The newer BLIP-2 model leverages frozen pre-trained image encoders and large language models to achieve state-of-the-art performance on various tasks with fewer trainable parameters. It uses a Querying Transformer to bridge the gap between the text and image modalities, enabling high performance in zero-shot image-to-text generation following natural language instructions [7].

2.1.3 ViLT

ViLT (Vision-and-Language Transformer) is a vision-language model that eliminates the usage of expensive image feature extraction via the use of object detectors and convolution, and instead utilizes the Visual Transformer's method [2] of processing an input image as a series of patches to speed up the efficiency of the model [5]. The architecture of ViLT is straightforward: it consists of a word embedding layer that processes tokenized text, a patch projection layer that projects an image grid to an embedding, and a transformer encoder layer that takes in both the processed word embeddings and patch embeddings as input. ViLT is then pre-trained using image text matching and masked language modelling as its objectives.

2.2. Modelling Assessments

To assess the above models, we utilize a largely shared assessment framework based on the model's architecture. Specifically, each of the above 3 model architectures are subject to zero shot capability testing, finetuning without architectural modifications, and finetuning with architectural modifications, if applicable.

2.2.1 Zero Shot Capability Testing

One of the ways we assess the models is to assess their *zero-shot* capabilities on the Hateful Memes Challenge. This will give us a rough idea of the model's generalization abilities onto unknown datasets. To do this, we directly feed the intended classes 'hateful' and 'non-hateful' as the text input to the model alongside the meme's image as the image input. We apply softmax on the model's output logits to obtain the probability of the meme being 'hateful' and 'non-hateful'. We note that this approach does not consider the **caption** of the meme during inference. It wholly relies on matching the image's encoded embeddings with the text embeddings for 'hateful' and 'non-hateful', with the caption being discarded completely.

2.2.2 Finetuning without Architectural Modifications

The next logical step to take would be to finetune the base model in the context of the Hateful Memes Challenge without any form of architectural modifications. We utilize the training dataset from the Hateful Memes Challenge to finetune the base models. Only the final Residual Attention Block layers of the image encoder and text encoders, the layer normalization layer / pooling layers following the image and text encoders, and the final projection layers of the models are finetuned. The remaining layers are frozen and not modified during training.

This approach is done to assess the ability of the model's base architectures when applied specifically on the Hateful Memes Challenge. We note that this approach was not done for ViLT due to its base architecture not containing any heads for text-specific logit predictions.

2.2.3 Finetuning with Architectural Modifications

Our final approach is to conduct architectural modifications on the base models and conduct finetuning on the modified models. This approach actively utilizes both the meme's image as well as the provided caption with the image.

A simple method of architectural modification is to simply add on a series of ReLU, Linear and Dropout layers on top of the base models without any manipulation of the outputs from the earlier layers of the model.

Another method is to utilize the outputs from the image and text encoders of the base models and further combine them before adding additional layers on top of the encoders' outputs to conduct classification. This method is inspired by the Hate-CLIPper paper's own implementation of CLIP in the context of the Hateful Memes Challenge [6]. The methods used to combine the image and text encodings are as follows:

1. Simple concatenation, where the image and text encodings are concatenated before being passed to the added layers to obtain the final probability of the meme being hateful,
2. Batch Matrix Multiplication between the image encodings and text encodings before passing it to our added layers,
3. Self-attention on the vector that is a result of concatenating the image and text encodings together as per 1. The vector we obtain from this self-attention is then passed to our final few layers.

The final few layers added on top of the encoders that are used to process the output matrices from the above 3 methods are a series of Linear, ReLU, Dropout and Linear layers.

2.3. Adversarial Robustness

Apart from accuracy, it is also crucial to assess and address adversarial vulnerabilities for such content filter use cases, to defend against malicious attackers trying to bypass it.

From an attack environment perspective, adversarial attacks can be categorized into white-box, black-box and grey-box attacks. In white-box attacks, the attacker has all prior knowledge of the victim model, including full information on model parameters and gradients. In contrast, for black-box attacks, the attacker has no knowledge of the underlying structure of the victim model, while in grey-box attacks the attacker only has partial knowledge.

In this study, we focus on algorithms that can be applied to the grey-box and black-box settings. While white-box attacks are typically the most effective, they are also unrealistic for real-world systems, as full information on a productionized model is rarely made public.

2.3.1 Grey-box Attacks

In the grey-box setting, we assume the stance of an attacker with access to only the publicly available pre-trained CLIP, but not our fine-tuned model nor any knowledge of the architectural modifications performed. Here, we focus on attacking the pre-trained CLIP model to generate adversarial images. Since only the image modality is involved in this process, to simplify the attack, we adopt the same approach of directly feeding the classes of 'hateful' and 'non-hateful' to the text as described above. We then adapted the Projected Gradient Descent (PGD) implementation from the *cleverhans* library to work with the CLIP schema, maximizing the cross-entropy

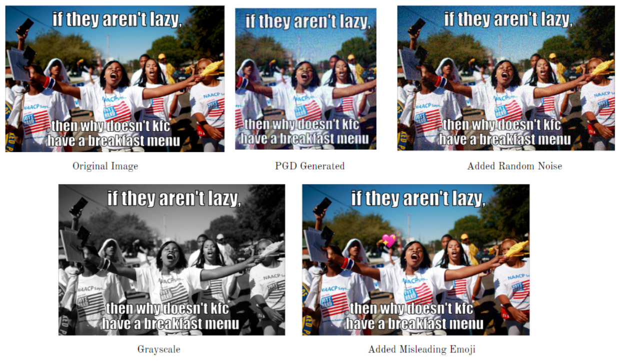

Figure 2. An example of adversarial transformations for a hateful meme from the test-unseen dataset

Method	Text
Original	if they aren’t lazy, then why doesn’t kfc have a breakfast menu
Unicode Replacement	if they aren’t lazy, then why Ďoesn’ kf have a brakḟast menu
Simulate Typos	tf they raen ’t lafy, then why doesn’ t kf have a breakfast menu

Table 1. Text Modality Attacks

loss in order to generate pixel values that result in high probabilities to the incorrect label.

2.3.2 Black-box Attacks

In the black-box setting, we assume the attacker has no information on the model at all, and attempts to fool the model using simple data augmentation techniques. The intention is to mimic the actions of how someone without any knowledge on machine learning might realistically attempt to bypass the content filter. Specifically, we leverage Meta AI’s *AugLy* data augmentation library to explore image and text augmentations to the original dataset, such as adding random noise and changing to greyscale. For the captions, we performance character-level attacks, such as introducing typos and replacing some parts with similar unicode characters. We also explore a simple example of adversarial patching, adding small emojis on the original memes with the aim of misleading the model. To do so, we extracted 30 positive and 30 negative emoji images from the internet, and overlay a randomly selected emoji on top of the original meme. If the meme is labeled to be hateful, a positive emoji is selected, and vice versa.

Figure 2 and Table 1 respectively show some examples of adversarial images and text generated from the process.

Effectiveness of attacks are evaluated on the following metrics: (1) Attack Success Rate, defined as the number of output predictions successfully altered, over the number of initial correct predictions from the original dataset; (2) difference in AUC on the adversarial dataset against original dataset and (3) difference in Accuracy.

2.3.3 Adversarial Training

In an attempt to defend against adversarial attacks, we first select the best performing attacks on both modalities, and apply them to a random 10% subset of the training dataset respectively. 10% was chosen as we felt that it was a reasonable augmentation proportion to add training signal while avoiding potential over-fitting. These adversarial examples are then shuffled into the original training dataset for model re-training and re-evaluation.

3. Experiments and Results

3.1. Zero Shot Performance

We assessed the models’ zero-shot performance on the dev-unseen and test-unseen dataset, and the results are as shown in Table 6. We note that all 3 models have rather similar results of around 0.5 for AUC and 50+% accuracy on the dev-unseen and test-unseen sets, albeit with CLIP still slightly outperforming the other two models. Given that we are obtaining such figures without any form of finetuning on the Hateful Memes Challenge dataset, this shows that the models’ zero-shot ability is decent and has a performance above random chance.

3.2. Model Performances after Simple Finetuning without Modifications

We then conduct finetuning for the CLIP and BLIP models on the training dataset using a learning rate of 1e-4 alongside the Adam optimizer for 10 epochs. We obtain much better results from this finetuning, with CLIP achieving AUC scores of **0.743** and **0.774** and accuracies of **70.5%** and **70.7%** on the dev-unseen and test-unseen sets respectively. BLIP does not perform as well as CLIP and achieves AUC scores of roughly 0.68 and accuracies of 67% on the dev-unseen and test-unseen sets. We note that the finetuning done was rather basic, but the results achieved were better than expected. The full results can be found in Table 7.

Specifically, although the above results lag behind the winners of the Hateful Memes Challenge in terms of AUC and accuracy, we note that the results are competitive with the results obtained by Kiela et al. [4] in their initial evaluation of the VILBERT and Visual BERT models re-trained in a multimodal fashion. Specifically, our finetuned CLIP model beats both VILBERT and Visual BERT in terms of AUC and accuracy on both the validation and test sets, as per Table 1 of [4]. This is impressive and showcases the promise that these newer architectures hold,

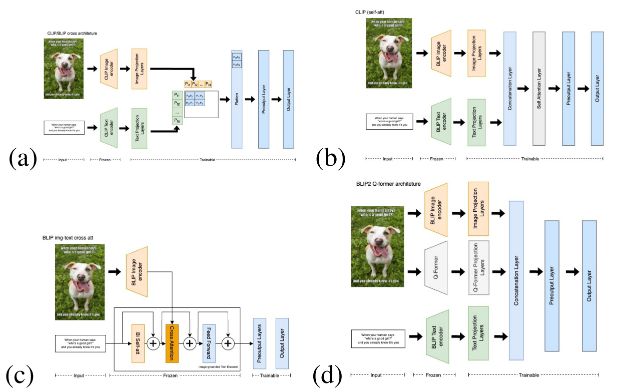
Figure 3. (a) Benchmark Architecture (b) CLIP with self attention layer (c) BLIP img-text cross attention (d) BLIP2 with Q-former

hence lending credence to our exploration of these newer architectures on the Hateful Memes Challenge.

3.3. Model Performances after Finetuning and Architectural Modifications

Our best performance was achieved using the architecture specified in Hate-CLIPper[6], as illustrated in 3a. Details regarding the parameter settings for it can be found in Table 13 in the appendix. Building upon this foundation, our team proposed various methods to replace the Feature Interaction Matrix (FIM) with alternative layers for learning cross-modal interactions between images and text. Specifically, we explored the utilization of a self-attention layer (Figure 3b), a cross attention between image-text layer (Figure 3c), and a q-former layer from BLIP2 (Figure 3d).

3.3.1 Key Findings

The top three performing models among the various iterations of CLIP, BLIP and ViLT models trained are documented in Table 2. Despite our efforts, our best performing model failed to surpass the benchmark architecture established by Hate-CLIPper [6]. Our analysis has identified several potential reasons for this outcome, which we will elaborate in the following subsections. Beyond the top three models highlighted in the table, a comprehensive list of performance metrics for all models that we have studied can be referenced in the appendix, specifically Table 8, 9, and 10 in the appendix.

Dataset: One of the reasons we find that CLIP outperforms BLIP and ViLT in the meme challenge possibly because of its broader training data, which enhances its ability to understand complex image-text relationships. Moreover, CLIP's training on unannotated natural data may better equip it for real-world tasks with noisy or limited labels.

Model	Dev-unseen AUC	Dev-unseen ACC	Test-unseen AUC	Test-unseen ACC
Hate-CLIPper	**0.811**	**76.7%**	**0.824**	**75.6%**
CLIP concat	0.761	70.2%	0.785	73.3%
BLIP-2 ViT-L FlanT5$_{XL}$	0.733	70.9%	0.764	71.6%

Table 2. Top 3 Best Performing Models

Training Strategy: CLIP trains its image encoder from scratch, tailoring representations for multimodal classification, potentially improving image-text associations. In contrast, BLIP utilizes pre-trained weights for both encoders, which may not capture multimodal nuances as effectively as training from scratch. ViLT utilizes pre-trained ViTb/32 weights on ImageNet to initialize its encoder and hence faces the same problem.

Implicit Modality Integration: BLIP incorporates visual information into the text encoder using cross-attention layers, while CLIP integrates text and image modalities implicitly through joint training. This implicit approach may lead to more robust representations, as CLIP learns to associate relevant text with images without relying on modality-specific architectures. When applying the batch-matrix-multiplication (BMM) operation, essentially we are aligning the attention between image and text, thus further boosting CLIP's performance. BLIP on the other hand, may not get this boost. Finally, ViLT's architecture makes it not possible to obtain and combine the encodings from its image and text specific encoders due to its use of a single shared transformer encoder, which limits its ability to generate more effective feature representations.

3.4. Performance on Adversarial Dataset

The victim model selected for this section was the variation of CLIP with cross layer, which was our best performing model at the time these experiments were performed. The most effective attacks are summarized in Table 3, while the full list of attacks and their metrics can be found in Table 11 in the appendix.

3.4.1 Robustness to Image Modality Attacks

The best attack to the image modality PGD, with the adversarial generated dev and test unseen datasets achieving a **18.89%** and **17.02%** success rate respectively. This finding is unsurprising as gradient-based approach tend to be the most effective at misleading models. Still, these pale significantly in comparison to the 100% success rate when applied to the pre-trained CLIP, proving that the attack had poor transferability to the downstream model. This could be in part due to the architectural modifications performed,

Dataset	Success Rate	AUC (%Δ)	Accuracy (%Δ)
Dev-unseen	**24.48%**	-4.75%	-8.87%
Test-unseen	**18.44%**	-7.69%	-5.78%

Table 3. Summary of robustness evaluation of combined attacks on image and text modality.

suggesting that the act of withholding complete information on the model architecture could in itself be an effective first line of defence for real-world use cases.

It is also noteworthy that adding emojis was surprisingly effective, attaining a **11.29%** and **9.94%** success rate respectively. Although not as effective as PGD, the comparative ease of attack and lower barriers to entry makes this finding a concerning one. Overlaying an image can be done easily by anyone without requiring any knowledge on machine learning. These were also markedly faster and easier to generate on a large scale, as compared to PGD which both required an understanding of the CLIP model and was more computationally intensive to perform.

3.4.2 Robustness to Text Modality Attacks

Text-based attacks proved to be less effective, with the best attack being replacing characters in the caption with similar unicode characters. This attack attained a success rate of 5.51% and 6.5% for dev-unseen and test-unseen respectively; and only decreased AUC and accuracy by less than 2%. The stark contrast in effectiveness reinforces the hypothesis that in a multi-modal architecture, the image modality is far more important than the text modality in determining the outputs.

3.4.3 Robustness to Combined Modality Attacks

Finally, we found that combining attacks to both modalities together resulted in the most effective model degradation. Specifically, we combined PGD and unicode character replacement, to attain a success rate of **24.48%** and **18.44%** respectively. Notably, the AUC of the test-unseen dataset fell by a significant **10.2%** to 0.695, highlighting the added vulnerability of multi-modal models due to the presence of multiple modalities.

3.5. Model Performance after Adversarial Training

When evaluated against the adversarial datasets, we found that training the model with a small subset of the adversarial examples can provide a significant improvement in defending against them. Specifically, success rates of the combined attack fell by 6.85% and 5.94% for dev and test specifically. Furthermore, model performance remained relatively unaffected by adversarial attacks, with only a **1.74%** and **2.11%** drop in AUC, indicating that the model was 5 times more robust after being provided with adversarial training examples.

Dev AUC (%Δ)	Dev Accuracy (%Δ)	Dev AUC (%Δ)	Dev Accuracy (%Δ)
0.768 (+1.28%)	72.2% (+2.0%)	0.804 (+0.71%)	75.2% (+1.9%)

Table 4. CLIP (Concat) Model Performance on Unseen Data after Adversarial Retraining

Furthermore, Table 4 highlights how adversarial training had further improved the model performance to achieve an AUC of **0.768** and **0.804** on the original dev and test unseen datasets respectively. Similarly, accuracy improved by 2.0% on both, suggesting that the augmentation of adversarial examples could help to provide additional training signal to the model and enhance performance further.

4. Challenges and Limitations

One of the main challenges we faced over the study was navigating the navigating through the hardware requirements to train these complex models. We note that unlike the winners of the Hateful Memes Challenge, we did not have the luxury of using powerful hardware to conduct model ensembling [3] to improve our results. Specifically, none of the model configurations we explored utilized ensembling, although we would like to note that our best performing model configuration which is motivated by Hate-CLIPper [6] is still able to achieve an AUC and accuracy that is only slightly worse than the previous first place solution [1]. Hence, a possible avenue for future exploration would be to conduct ensembling of a variety of modern visual-language model architectures together to see if such configurations are able to beat the best benchmark.

Another limitation we faced was the computational complexity of the adversarial PGD attack, in part due to the complexity of our proposed models. Specifically, as adversarial images are generated from the gradients of a model, in light of time and resource constraints, we had to choose a specific model to perform the adversarial evaluation and retraining on. This decision was informed by the hypothesis that adversarial attacks would have similar effects across all vision-language models, and we chose to prioritize CLIP-based models over BLIP and ViLT due to its superior performance. Nevertheless, future model-specific robustness studies to adversarial datasets will definitely prove fruitful for the study of multimodal models in general.

Student Name	Contributed Aspects	Details
Ke Xin Chong	CLIP implementation	Reproduce the model architecture and result from Hate-CLIPper. Perform finetuning and various modification on the architecture on CLIP, compare the results with BLIP & BLIP2.
Michelle Lim	Adversarial Dataset Generation, Evaluation and Training	Implemented PGD for CLIP and generated augmented imafe and text datasets. Performed adversarial evaluation and training and conducted assessments on model robustness.
Wei Hong Low	BLIP & BLIP2 implementation	Prepare the dataloader pipeline. Finetuned on various BLIP & BLIP2 architecture and compare the results with CLIP.
Qin Wayne Toh	Zero-Shot and Simple Finetuning analysis for CLIP, BLIP, ViLT.	Conducted zero-shot performance analysis for CLIP, BLIP and ViLT. Did simple finetuning for CLIP and BLIP without architectural modifications. Finetuned ViLT via the use of additional layers.

Table 5. Contributions of team members.

5. Appendix

The Appendix contains the full performance tables of the CLIP, BLIP and ViLT models which are referenced inside the main text. It also contains the specifications and full performance of the best CLIP model's results on adversarial unseen data before and after adversarial retraining.

Model	Dev-unseen AUC	Dev-unseen Accuracy	Test-unseen AUC	Test-unseen Accuracy
clip-vit-large-patch14	**0.539**	**60%**	0.537	**60.6%**
blip-image-captioning-large	0.525	52.78%	**0.547**	57.05%
vilt-b32-finetuned-coco	0.524	55.74%	0.467	51.4%

Table 6. Zero-Shot Performance of Models

Models	Dev-unseen AUC	Dev-unseen Accuracy	Test-unseen AUC	Test-unseen Accuracy
clip-vit-large-patch14	**0.743**	**70.6%**	**0.775**	**70.7%**
blip-image-captioning-large	0.670	66.5%	0.697	67.9%

Table 7. Finetuning Performance of CLIP and BLIP Models without modifications

CLIP Pretrained Model Used	#Proj. layer	#P.O layer	Fusion	#t. param.*	Dev AUC	Dev ACC	Test AUC	Test ACC
clip-vit-large-patch14	1	1	concat	3,936,257	**0.761**	**70.2%**	**0.785**	**73.3%**
	5	1	self-att	24,922,113	0.529	63.0%	0.510	62.5%
	10	1	cross	1,094,473,729	0.765	73.9%	0.766	73.3%
laion/CLIP-ViT-B-32-laion2B-s34B-b79K	1	5	cross	1,075,056,641	0.738	69.1%	0.766	73.3%

Table 8. CLIP Models' Performance on Unseen Data after architectural modifications, *#t.params - total number of trainable parameters

BLIP variations	BLIP Pretrained Model Used	#Layers	Fusion	#t. param.*	Dev AUC	Dev ACC	Test AUC	Test ACC
BLIP ITM	Salesforce/blip-itm-large-coco	P.O: 6	Cross att. between img. & text	1,488,642	0.690	64.6%	0.679	65.2%
BLIP Base	Salesforce/blip-image-captioning-large	Proj.: 5 P.O: 1	cross	43,393	0.638	64.1%	0.678	66.1%
BLIP2	Salesforce/blip2-opt-2.7b	Proj.: 5 P.O: 1	concat	5,903,361	0.726	69.1%	0.745	69.8%
	Salesforce/blip2-flan-t5-xl	Proj.: 5 P.O: 1	concat	46,674,945	**0.733**	**70.9%**	**0.764**	**71.6%**

Table 9. BLIP Models' Performance on Unseen Data after architectural modifications, #t.params - total number of trainable parameters

Pretrained Model Used	Number of ReLU, Linear and Dropout layers	#trainable. param.	Dev AUC	Dev Accuracy	Test AUC	Test Accuracy
vilt-b32-mlm	1	7,709,246	**0.663**	**65.19%**	**0.701**	**68.45%**
vilt-b32-mlm	2	7,710,728	0.656	67.59%	0.684	68.2%
vilt-b32-mlm	3	7,712,210	0.651	65.92%	0.668	68.15%

Table 10. ViLT models' Performance on Unseen Data after architectural modifications

Modality Attacked	Attack	Dev Success Rate	Dev AUC (Δ%)	Dev Accuracy (Δ%)	Test Success Rate	Test AUC (Δ%)	Test Accuracy (Δ%)
Image	PGD	18.89%	0.714 (-4.14%)	63.7% (-6.52%)	17.02%	0.722 (-7.46%)	66.2% (-7.13%)
Image	Add emoji	11.29%	0.747 (-0.80%)	65.7% (-4.46%)	9.94%	0.752 (-4.51%)	67.6% (-5.70%)
Image	Grayscale	7.61%	0.726 (-2.9%)	68.7% (-1.50%)	5.49%	0.765 (-3.19%)	70.9% (-2.45%)
Image	Add noise	7.09%	0.737 (-1.76%)	69.1% (-1.13%)	8.40%	0.770 (-2.75%)	69.6% (-3.75%)
Text	Replace text with similar unicode characters	5.51%	0.741 (-1.43%)	68.9% (-1.31%)	6.5%	0.763(-3.36%)	70.1% (-3.20%)
Text	Simulate typos	4.46%	0.749 (-0.57%)	69.6% (-0.57%)	5.32%	0.769 (-2.80%)	70.7% (-2.65%)
Combined	PGD + Replace text with similar unicode characters	**24.48%**	**0.704 (-5.06%)**	**61.7% (-8.51%)**	**18.44%**	**0.695 (-10.2%)**	**65.7% (-7.63%)**

Table 11. Evaluation of CLIP (Concat) model on Adversarial Unseen Data

Modality Attacked	Attack	Dev Success Rate	Dev AUC (Δ%)	Dev Accuracy (Δ%)	Test Success Rate	Test AUC (Δ%)	Test Accuracy (Δ%)
Image	PGD	14.47%	0.763 (-0.53%)	72.1% (-0.10%)	14.47%	0.787 (-1.71%)	74.6% (-0.60%)
Text	Replace text with similar unicode characters	4.36%	0.764 (-0.40%)	71.7% (-0.50%)	3.19%	0.795(-0.90%)	75.6% (+0.30%)
Combined	PGD + Replace text with similar unicode characters	**17.63%**	**0.750 (-1.74%)**	**69.2% (-3.1%)**	**12.5%**	**0.783 (-2.11%)**	**77.1% (+1.9%)**

Table 12. Evaluation of CLIP (Concat) model on Adversarial Unseen Data After Adversarial Retraining

Parameters	Values
Pretrained CLIP model	clip-vit-large-patch14
Image Projection dimension	1024
Text Projection dimension	1024
Pre-Output layer	1
Projection layer	5
Fusion	CROSS
Optimizer	AdamW
Maximum epochs	20
Batch size	64
Learning rate	0.0001
Gradient clip value	0.1

Table 13. Parameters for our benchmark model

References

[1] Facebook AI. Hateful memes challenge leaderboard, 2021. 6

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[3] Casey Fitzpatrick. Meet the winners of the hateful memes challenge, 2021. 1, 6

[4] 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. *Advances in neural information processing systems*, 33:2611–2624, 2020. 1, 2, 4

[5] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *International conference on machine learning*, pages 5583–5594. PMLR, 2021. 1, 2

[6] Gokul Karthik Kumar and Karthik Nandakumar. Hate-clipper: Multimodal hateful meme classification based on cross-modal interaction of clip features. *arXiv preprint arXiv:2210.05916*, 2022. 3, 5, 6

[7] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023. 2

[8] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pages 12888–12900. PMLR, 2022. 1, 2

[9] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1, 2

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2

[11] Ziqi Zhou, Shengshan Hu, Minghui Li, Hangtao Zhang, Yechao Zhang, and Hai Jin. Advclip: Downstream-agnostic adversarial examples in multimodal contrastive learning. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 6311–6320, 2023. 1