Visual Recognition (AIM 825) - Mini Project 2

Gutla Gayathri, MT2024053
Sai Shruti Vadrev PH2024504
Sai Shruti Prakhya MS2024018

May 18, 2025

Introduction

This project explores the application of parameter-efficient fine-tuning using LoRA on vision-language models—ViLT and BLIP—for the Visual Question Answering (VQA) task. We compare baseline and fine-tuned performance across metrics like Accuracy, F1 Score, BERT Score, and BART Score. Innovative strategies such as label adaptation, cosine similarity-based answer replacement, and beam search decoding are integrated to improve model performance.

Methodology

0.1 Dataset Curation

We used the small variant of the Amazon Berkeley Objects (ABO) dataset that consisted of product images and corresponding metadata [1]. A subset of the ABO dataset consisting of approximately 20000 unique images and 5000-8000 unique products was used to generate the final Question and Answer (QnA) dataset. Each image and its associated metadata was passed as an input to a Gemini vision-language generative model through an API call along with a prompt. The model output was a question and one word answer to the question, grounded in the image and metadata passed to it. Different prompting strategies were used for same, a summary of which is given below [2].

1. Vanilla Instruction Prompting: Involves passing the input data to the model, along with specifying the type of output expected from it. It helps to be as descriptive as possible and clearly elucidate the requirements. In standard vanilla prompting, the query is just thrown at the LLM/VL model without any engineering of the query so as to obtain a better response. In our work, we tried to give a refined prompt in an instructive fashion, hence we call it "Vanilla Instruction Prompting".

2. Few Shot Prompting: The query passed to the model also includes some examples of the output expected by the user. This helps set context for the task at hand. The number of examples given can vary.

3. Chain of Thought Prompting: Tries to simulate human reasoning by making the model break down complex tasks into a series of logical steps. The model is expected to not just arrive at the correct/desired output but to also explain how it got there. Helps the model arrive at more logical answers and especially useful for tasks involving problem solving.

0.2 Model Choices

We conducted our experiments on 2 models - Vision Language Transformer (ViLT) and Bootstrapping Language-Image Pre-training (BLIP). The rationale behind choosing these 2 models was that we wanted to compare the performance of a larger model (BLIP has approximately 248 million parameters) and a smaller model (ViLT has approximately 124 million parameters). We also wanted to compare the performance of a generative model (BLIP) and an encoder-only discriminative model (ViLT). Table 1 does a comprehensive comparison between the two models. Figures 1 and 2 show both the model pipelines [3, 4].

Aspect	**BLIP**	**ViLT**
Model Type	Encoder–decoder (generative)	Single-stream encoder-only (discriminative)
Vision Backbone	ViT (e.g., ViT-B/16) with cross-attention in decoder	Linear patch projection within Transformer layers
Text Backbone	BERT-Base decoder for autoregressive generation	BERT-Base fused in the encoder for contextual embedding
Pre-training Objectives	Image-Text Contrastive, Image-Text Matching, Image-Conditioned Language Modeling	Masked Language Modeling (MLM), Image-Text Matching (ITM)
Parameter Count	224M total	197M total
Generative Capability	Yes (free-form captioning, QA)	No (classification, retrieval only)
Downstream Tasks	Open-ended captioning, VQA, retrieval, video transfer	VQA (classification), image-text retrieval, matching
Efficiency	Heavier decoder, slower inference	Up to 10× faster (no external detectors)

Table 1: Comparison of BLIP vs. ViLT architectures and capabilities.

0.3 LoRA Based Fine-Tuning

Low-Rank Adaptation (LoRA) is a parameter-efficient fine-tuning technique that introduces trainable rank-decomposition matrices into the attention layers of pre-trained models while

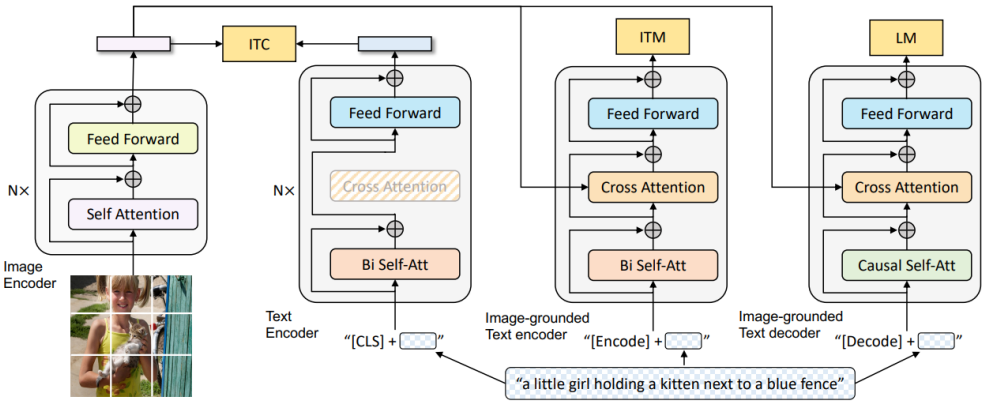

Figure 1: BLIP Model Pipeline

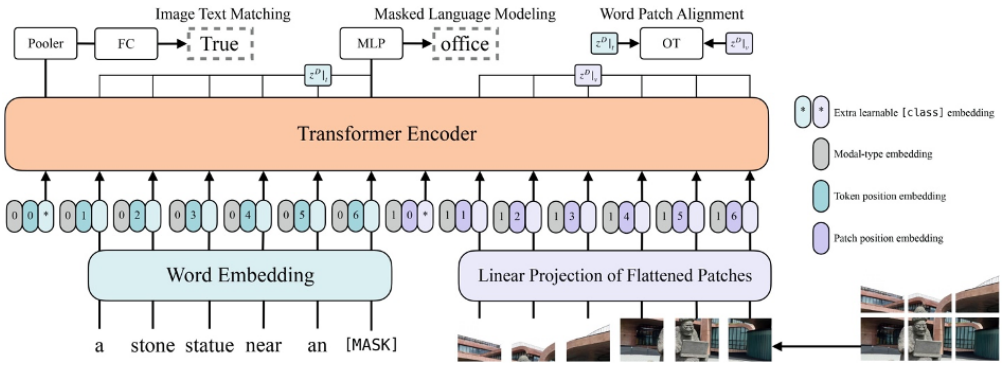

Figure 2: ViLT Model Pipeline

freezing the original weights. This approach significantly reduces the number of trainable parameters, enabling efficient adaptation to downstream tasks without compromising performance. LoRA proposes an efficient mechanism to adapt large pre-trained models by injecting low-rank trainable matrices into existing layers while keeping the original model weights frozen. Specifically, LoRA decomposes the weight update for a weight matrix $W \in \mathbb{R}^{d \times k}$ as:

$$\Delta W = BA$$

where $A \in \mathbb{R}^{r \times k}$ and $B \in \mathbb{R}^{d \times r}$ are trainable matrices and $r \ll \min(d, k)$ is the rank of the adaptation. The final modified layer becomes:

$$W' = W + \alpha \cdot BA$$

Here, α is a scaling factor (often set as $\alpha = \frac{\text{LoRA Alpha}}{r}$) that controls the magnitude of the adaptation and ensures stable training.

In the case of multi-head attention, LoRA is typically applied to the query and value projection matrices. For a self-attention layer with input $X \in \mathbb{R}^{n \times d}$, the LoRA-modified projection (e.g., for the query) becomes:

$$Q = X(W_q + \Delta W_q) = X(W_q + B_q A_q)$$

This setup allows for significant reduction in the number of trainable parameters: instead of fine-tuning a full W_q or W_v, we only train the much smaller A_q, B_q, A_v, and B_v matrices. For example, for a weight matrix of size 768×768 and a rank of $r = 8$, the LoRA approach introduces only $12,288$ trainable parameters compared to $589,824$ in full fine-tuning.

LoRA also allows for compatibility with mixed-precision training and other acceleration techniques. In our work, LoRA enabled us to fine-tune ViLT and BLIP efficiently under computational constraints, while preserving generalization and improving downstream performance on our VQA dataset. Please note, all these equations have been taken from the original LoRA paper [5].

In our experiments, we applied LoRA to the query and value projection layers of the transformer modules in both ViLT and BLIP models. The LoRA configuration parameters, such as rank (r), scaling factor (LoRA Alpha), and dropout, were carefully selected and are summarized in Table 2. The use of LoRA allowed us to fine-tune large vision-language models effectively on our VQA task with limited computational overhead

Setting	ViLT-1	ViLT-2	BLIP
LoRA Rank (r)	8	16	16
LoRA Alpha	32	32	32
LoRA Dropout	0.1	0.1	0.1
Target Modules	`query, value`	`query, value`	`query, value`
Training Epochs	20	20	20
Optimizer	AdamW	AdamW	AdamW

Table 2: LoRA Fine-Tuning Settings for ViLT-1, ViLT-2, and BLIP

0.4 Metrics

Evaluating Visual Question Answering (VQA) models is crucial for understanding their performance and guiding further development. Several metrics are used to assess how well these models predict the correct answers to questions about images. These metrics provide different perspectives on the model's accuracy, precision, and ability to understand the relationship between visual content and language. Below is a list of the metrics used, along with explanations and their respective formulas:

1. Accuracy: Accuracy measures the proportion of the model's predictions that exactly match the ground truth answers. It is a straightforward metric that gives an overall sense of how often the model is correct.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{1}$$

2. Precision: Precision measures the proportion of the model's predicted answers that are actually correct. It focuses on the quality of the model's positive predictions.
Precision = (Number of True Positives) / (Number of True Positives + Number of False Positives)

3. Recall: Recall measures the proportion of the ground truth answers that the model correctly predicts. It focuses on the model's ability to find all the correct answers.

$$\text{Recall} = \frac{\text{Number of True Positives}}{\text{Number of True Positives} + \text{Number of False Negatives}} \tag{2}$$

4. F1 Score:

   The F1 score is the harmonic mean of precision and recall. It provides a balanced measure of a model's accuracy, considering both the precision and recall. A high F1 score indicates that the model has both high precision and high recall.

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

5. BERTScore Precision: BERTScore Precision measures how much of the predicted answer is relevant to the ground truth answer, using contextual embeddings from the BERT model. It calculates the similarity between the embeddings of the words in the predicted and ground truth answers. A higher precision indicates that the predicted words are closely related to the words in the correct answer. Let c_i be the embedding of the i-th token in the candidate (predicted) sentence, and r_j be the embedding of the j-th token in the reference (ground truth) sentence. The similarity between c_i and r_j is given by their cosine similarity. For each token in the candidate sentence, we find the maximum similarity score with any token in the reference sentence. Precision is the average of these maximum similarity scores.

6. BERTScore Recall: BERTScore Recall measures how much of the ground truth answer is captured by the predicted answer. Like precision, it uses BERT embeddings and

$$Precision = \frac{1}{|C|} \sum_{i=1}^{|C|} \max_j s_{ij}$$

cosine similarity, but focuses on the perspective of the ground truth answer. A higher recall indicates that the predicted answer covers most of the important words in the correct answer.

Using the same notation as above, for each token in the reference sentence, we find the maximum similarity score with any token in the candidate sentence. Recall is the average of these maximum similarity scores.

$$Recall = \frac{1}{|R|} \sum_{j=1}^{|R|} \max_i s_{ij}$$

7. BERTScore F1: The BERTScore F1 is the harmonic mean of BERTScore precision and recall. It provides a balanced evaluation of the semantic similarity between the predicted and ground truth answers, taking into account both how relevant the prediction is and how much of the ground truth answer it covers.

$$\text{BERTScore}_{F1} = \frac{2 \times \text{BERTScore}_{\text{Precision}} \times \text{BERTScore}_{\text{Recall}}}{\text{BERTScore}_{\text{Precision}} + \text{BERTScore}_{\text{Recall}}} \tag{4}$$

8. BARTScore: BARTScore evaluates the quality of generated text by comparing it to a reference text using the pre-trained BART model. It measures how well the generated text matches the meaning of the reference text. BARTScore is calculated by feeding the generated and reference texts to the BART model and extracting the log probabilities of the reference text given the generated text. These log probabilities are then averaged to produce the final score.
While there isn't a single, simple formula, the core idea is:

$$\text{BARTScore} = \text{Average}\left[\log P(\text{Reference Text} \mid \text{Generated Text})\right] \tag{5}$$

Results

0.5 Dataset Curation

Our final dataset consists of around 25000+ unique image-question-answer touples. The below are some examples from our dataset. We noticed that there wasn't much of a difference between the outputs from the different prompting strategies. This might have occurred

because our instruction prompt was very well defined and thorough, coupled with the fact that the metadata was very comprehensive. Another point to note is that filtering the metadata to include only relevant field such as 'keywords', 'color' etc. before the API call, produced only a marginal improvement in the questions obtained.

One of the challenges faces during the data curation phase, was the generation of a huge number of questions with answers as yes/no (approximately 40%). The worry was that this would bias the data and hence an additional 4000 samples were generated, explicitly instructing the model to not generate such question-answer pairs.

Figure 3: Vanilla Instruction Prompt
Question: What is the gemstone type of the earrings?
Answer: Aquamarine

0.6 Model Baselines and Fine-Tuning

During training, we encountered challenges due to certain answers in our dataset not being present in the original `label2id` mapping of the ViLT model. To address this in ViLT-1, we extended the mapping by adding the missing answers. However, the results were suboptimal. For ViLT-2, we improved this approach by replacing the out-of-vocabulary answers with their most semantically similar alternatives using cosine similarity in the embedding space, which led to better performance. Additionally, we leveraged mixed-precision training using CUDA AMP to accelerate training and reduce memory consumption. For inference with BLIP, we incorporated beam search to generate more coherent and contextually accurate answers, enhancing the quality of VQA predictions. These innovations contributed to both training efficiency and improved evaluation metrics.

The results in Tables 3 and 4 reveal significant performance improvements after fine-tuning both ViLT and BLIP models. Initially, ViLT outperformed BLIP in terms of accuracy and F1 Score (M), but BLIP showed stronger BERT-based semantic alignment (BERT Precision, Recall, and F1). However, after fine-tuning, BLIP exhibited the most substantial overall gain, achieving the highest accuracy (0.7564) and outperforming ViLT-1 and

Figure 4: Chain of Thought Prompt
Question: What is the primary material of this phone case?
Answer: Silicon

Figure 5: Few Shot + Chain of Thought Prompt
Question: What is the base material of the lamp?
Answer: Marble

ViLT-2 across all metrics, especially in precision, recall, and F1 Score (M), suggesting better matching of model outputs to ground-truth answers. The BERTScore metrics remained relatively high across all models, indicating strong semantic overlap even when token-level matches were weak. Interestingly, BARTScore values remained negative for all fine-tuned models, likely due to the nature of the log-likelihood computation, though BLIP's baseline BARTScore was anomalously high (0.6892), possibly reflecting overfitting or domain mismatch during pretraining. Overall, the results suggest that fine-tuning meaningfully boosts both syntactic and semantic alignment, with BLIP emerging as the most effective model post-tuning.

Metric	ViLT	BLIP
Accuracy	0.5548	0.4871
Precision (M)	0.1008	0.0010
Recall (M)	0.1074	0.0010
F1 Score (M)	0.0895	0.0010
BERT Precision	0.8221	0.9732
BERT Recall	0.8187	0.9675
BERT F1	0.8203	0.9671
BARTScore	–4.0996	0.6892

Table 3: Baseline Metrics

Metric	ViLT-1	ViLT-2	BLIP
Accuracy	0.6367	0.6796	0.7564
Precision (M)	0.0604	0.1208	0.2813
Recall (M)	0.0810	0.1220	0.2880
F1 Score (M)	0.0643	0.1090	0.2546
BERT Precision	0.8009	0.8412	0.8245
BERT Recall	0.8006	0.8376	0.8147
BERT F1	0.7993	0.8382	0.8170
BARTScore	–3.5814	–3.3844	-3.618

Table 4: Fine-Tuned Metrics

Conclusion

Our experiments show that LoRA-based fine-tuning can significantly boost VQA performance while requiring fewer trainable parameters. BLIP, when fine-tuned, outperforms ViLT variants in accuracy and language-level metrics, demonstrating the effectiveness of generative models with prompt-aware tuning. Techniques like AMP training and answer-space refinement further contributed to these performance gains.

References

[1] J. Collins, S. Goel, K. Deng, A. Luthra, L. Xu, E. Gundogdu, X. Zhang, T. F. Yago Vicente, T. Dideriksen, H. Arora, M. Guillaumin, and J. Malik, "Abo: Dataset and benchmarks for real-world 3d object understanding," *CVPR*, 2022.

[2] S. Vatsal and H. Dubey, "A survey of prompt engineering methods in large language models for different nlp tasks," 2024. [Online]. Available: https://arxiv.org/abs/2407.12994

[3] W. Kim, B. Son, and I. Kim, "Vilt: Vision-and-language transformer without convolution or region supervision," in *International conference on machine learning*. PMLR, 2021, pp. 5583–5594.

[4] J. Li, D. Li, C. Xiong, and S. Hoi, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation," in *International conference on machine learning*. PMLR, 2022, pp. 12 888–12 900.

[5] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," 2021. [Online]. Available: https://arxiv.org/abs/2106.09685