King Fahd University of Petroleum and Minerals
College of Computing and Mathematics

ICS 504: Deep Learning

Examining the Potential of Multimodality for Vision-Language Models

Saad Alghamdi – 202404220
Mohammed Alharthi – 201840480
Abdulaziz Alshukri – 202403840

Supervisor: Dr. Muzammil Behzad

April 26, 2025

Abstract

General purpose vision language models are proven to be useful at identifying key components in images and generating a response based on a given prompt. They can be widely used in applications such as image captioning, visual question answering, visual search, etc. With the introduction of the Instruct BLIP model, vision language models have demonstrated improved zero-shot testing resulting in better generalization across various tasks. This is possible thanks to the addition of an instruction-based prompting that takes better advantage of the query transformer and enables better feature extraction from images based on the given instruction. This project aims to explore multiple configuration settings to Instruct BLIP architecture. Additionally, it introduces a new method of compressing the model's parameters by implementing Low-Rank Adaptation (LORA) method. The goal is to preserve at least 70% of the original model's performance to enable easier deployment on resource constrained devices.

Table of Contents

Table of Figures

Table of Tables

1 Introduction

A large paradigm of Artificial Intelligence is mainly concerned about mimicking the human ability to perceive and understand the world that surrounds it. Humans perceive things as intelligent if they exhibit some sort of behavior that's associated with our definition of intelligence. These behaviors can be simple to humans and range from visual perception and understanding, to verbal communication and reasoning. These behaviors might be trivial to a human but are very complicated to machines and require an immense amount of training data and computational power. One emerging type of AI models addressing these challenges is vision language models (VLMs), which combines both visual recognition and language generation to perform tasks such as VQA and image captioning.

The Instruct BLIP Vision language model is built on top of BLIP-2 model, it contains a query transformer layer that is used to extract important features from images. The idea of the instruction-based tuning is primarily used for LLM's in general, The Instruct BLIP model leverages this technique to better guide feature extraction from images in the Q-Former layer. This technique results in state-of-the-art performance in zero-shot testing, meaning that the instruction-based tuning helps the model to generalize better on unseen datasets.

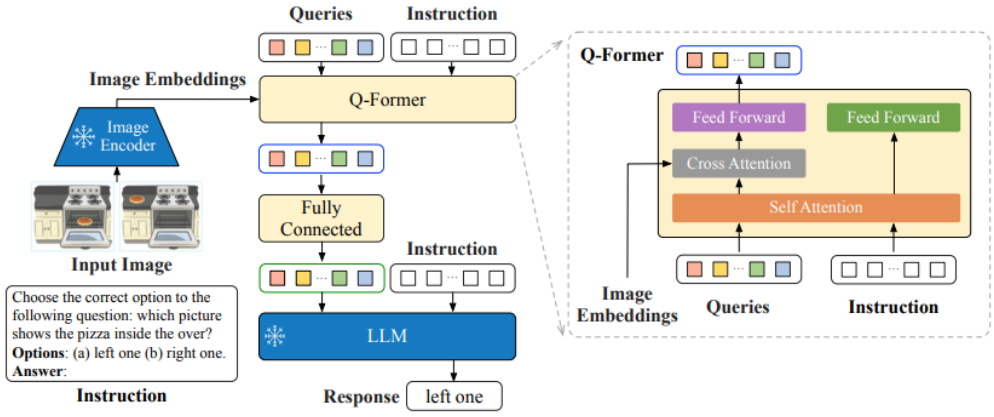

Figure 1 Instruct BLIP model architecture

The challenge that this paper tries to solve is how this model can be de-scaled to make it more efficient to run on devices with low computational power. Hence the objective of the paper is to introduce Low-Rank Adaptation (LORA) technique which is applied on layers of the language model to enable efficient fine tuning with fewer number of trainable parameters. Additionally, the different components of the architecture (Fig.1) such as the Image encoder, the Q-Former, and the last 4 layers of the LLM are unfrozen to allow for the experimentation of different configuration variables such as activation function, hidden size, number of layers, attention heads and intermediate size. Each variation will be tested with different values or LORA rank and aloha values. The paper then provides a comprehensive analysis and benchmark testing that captures the model's overall capability.

2 Literature Review

The need to create intelligent systems that can perceive, understand and react has led to the development of vision language models (VLMs). VisualBERT is a pioneering model that combines image and textual data using a transformer model. It uses a layer of image detection to spot parts of the image that are significant, it also tokenizes the text and process both parts through a transformer that is used to detect relationships between the important areas of the image and the textual data. The following model "ViLT" solved the problem of slow processing cause by the visual detector and introduced the ability to input fixed size image patches to account for the external visual detector, this approach resulted in faster computational speed and a lighter model. The real problem however is that both approaches lacked zero-shot generalization and were primarily task specific hence the BLIP model was developed.

The BLIP model focused on primarily fixing the problem of task specific models that do not perform well on tasks like image captioning. It solves this problem by using a combination of caption generation and filtering to improve the training data, so that the model teaches itself using its own generated clean captions which improves the quality of learning. The downside of BLIP model was that it required full training from scratch, it didn't utilize the power of large language models (LLMs), and it treated instructions the

same which was not useful for generalization. The BLIP-2, however, utilized the power of existing LLMs and introduced a query transformer layer that helps extracting only the most important parts of the image. The approach of incorporating frozen image encoder and LLM made the model cheaper and faster to train and improved its zero-shot capabilities.

The instruction-based fine-tuning is mainly used in LLMs to give the model the ability to understand what task it is being asked to do, which helps the model to follow the command and understand the intent. Following the BLIP-2, the Instruct BLIP was created to provide instruction-based fine-tuning to the query transformer layer, which allows the Q-Former to extract important visual features based on the given instruction. The training data uses triplets of instruction-image-response to allow the correct feature extraction based on the given instruction. This approach greatly enhanced the zero-shot testing as the model now can follow the given command and understands the intent of the question and what the appropriate answer should be. This ensures that the model not only improves on a wide range of tasks but also performs better in real world scenarios where instructions are very diverse.

With the increasing complexity of VLMs, fine tuning models require a significant amount of computing power. This requirement makes it hard to experiment with different configurations since time and resources are limited. To address this, parameter-efficient fine-tuning techniques such as LoRA are introduced. LoRA is a technique used to fine-tune large models efficiently by introducing small, trainable matrices into specific layers in a frozen model. This approach decreases the number of trainable parameters. Moreover, it reduces the overall model size allowing not only faster and more frequent fine-tuning, but also easier deployment in devices with low computing power. This, however, comes at the expense of losing some of the original model's accuracy. The trade-off in this case depends on the application the model is used in.

This paper focuses on experimenting with different configurations of the model with the implementation of LoRA in the LLM model. The vision encoder, Q-Former, and last 4 layers of the LLM are unfrozen to allow for different variations and testing including changing

the activation function, hidden size, number of layers, attention heads and intermediate size. These variations are also tested with multiple implementations of LoRA techniques that differ in alpha and rank values. Then the models are evaluated based on several metrics including the MME benchmark for instruction-based tasks, ROUGE scores for language generation, and CLIP for vision-language correspondence.

Recent vision-language models focused primarily on improving the performance of the models, the gap remains when it comes to enhancing the model's efficiency and complexity while maintaining significant portion of its reliability. The use of parameter-efficient tuning such as LoRA in vision-language models is underexplored. Also, experimenting with different configurations settings of other components such as the vision encoder, Q-Former, and LLM has not been fully investigated especially in the context of balancing model descale with performance retention.

3. Proposed Methodology

3.1 Existing Model and Challenges

InstructBlip is an advanced vision language model that is based on BLIP-2 (Bootstring Language-Image Pre-training). The model is an aggregation of different components, where each of them is a standalone model. Then InstructBlip works to glue each of these components and create the bootstrapping to have a powerful vision language model. The main three components of the model are the **Vision Encoder** which based on CLIP ViT (Vision Transformer); the vision encoder processes the input images and extracts the visual features by dividing the images into patches and passing them through several multiple self-attention layers. The **Q-Former** (Query Transformer) is the shared space that will align the visual features with inserted instructions to create unified embeddings to be passed to the language model. The last component is the **Language model** which generates the output based on the received multimodal input.

Despite being a powerful vision language model, InstructBlip has some limitations that might hinder its capabilities. The model hallucination is one of the issues due to the

tendency of the language models to generate sentences even with the absence of ground truth. Another challenge that the model is facing is down streaming the model for subtasks by fine-tuning due to the huge number of parameters. If the model components unfreeze, the model will have many trainable parameters, which require an extensive number of computational resources.

3.2 Proposed Enhancements

To address the challenge of fine-tuning the original InstructBlip model, we propose an optimized enhancement that focuses on preserving the model's performance while optimizing the computational efficiency. This enhancement is introduced by applying LoRa (Low Rank Adaptation). LoRa is a parameter-efficient fine-tuning technique (PEFT) that works by introducing low-rank matrices to be trained instead of training the total parameters of the model. In the proposed model, we applied LoRa on targeted modules that are related to the attention mechanism, as it is responsible for the general understanding of the relationship between the visual and textual representation.

3.3 Algorithm and Implementation

The implementation of the enhanced InstructBlip model was done by following a systematic approach to fine-tune the model by using the VL-RewardBench dataset. The dataset consists of 1250 samples curated for several vision language tasks with a focus on situations that would likely cause the model to start hallucinating; that approach in the dataset will lead the model to enhance its responses to a ground truth. We started the training by unfreezing specific model components, specifically the vision encoder, the Q-former, and a few layers of the language model, to be able to apply the experimental hyperparameter tuning on the model. Then LoRa is applied on selected attention modules to decrease the number of trainable parameters.

The fine-tuning process is a crucial step for successful model fine-tuning. We started by preprocessing the dataset to fit the model input requirements. The InstructBlip processor is utilized to transform the image and text at each input to create compatible inputs. The

preprocessing handled the image-text pairs and the target, which are the responses in the dataset.

The training arguments were selectively optimized to balance between the model performance and the computational constraints. The batch size was split into 2 per device with accumulation steps of 16, which resulted in a larger batch size of 32 to reach almost to the maximum capabilities of the utilized A100 GPU for the training and fine-tuning of our model.

3.4 Loss Function and Optimization

In our implementation, we have chosen cross-entropy to be used as the main training loss to handle the language complexity. The probabilistic nature of the cross-entropy loss makes it a good choice to check the generated embeddings against the ground truth or the target in the dataset. We have also introduced different metrics and benchmarks to test the model and decide the feasibility of the fine-tuned model, such as ROUGE, BLUE, and MME Benchmark for assessing vision language tasks.

4. Experimental Design and Evaluation

4.1 Datasets and Preprocessing

The fine-tunning was conducted using VL-RewardBench dataset, which a complete and carefully curated dataset to enhance vision-language models ability in instruction-based tasks and to reduce the hallucination by training on ground truth results on vague situations. This dataset consists of 1250 example we used 80% in (1000 example) in training and kept the remaining 20% (250 example) for testing purposes.

The preprocessing step started with discarding the unneeded features, because the dataset provided extra features for human feedback, which is unnecessary in our fine-tuning task , we only kept the input image, the query (instruction), and the target (response). Next step was to utilize the InstructBlip preprocessing pipeline which consist of the following:

- Image processing: resizing any image to 224*224 and converting to Tensor format to match the input requirements of the vison encoder.
- Text tokenization: the InstructBlip tokenizer will convert the queries and responses to tokens with fixed length of 512 tokens, the tokenizer will use padding and truncation to match the fixed length.
- Input formatting: the processed images and tokenized text combined and passed to be processed by the model in compatible format.
- Label preparation: the tokenized label is processed to be to create label tensors to calculate the cross-entropy loss during the training.

4.2 Performance Metrics

During the experimental process we used different metrics to assess the performance of the fine-tuned models. The main training loss metrics used is the cross-entropy loss to measure the predication accuracy in comparison with the label. ROUGE metrics were used to evaluate the text-generation quality.

- ROUGE-1: Measures world-level similarity.
- ROUGE-2: Measures local word order and phrasing.
- ROUGE-L: Measures the sentence structure and similarity.

The ROUGE score is ranging from 0 lowest to 1 highest possible score which indicates high level of fluency in the generated responses.

BLUE metric was used to measure the model ability to handle text in other languages and translations. BLEU sore ranges from 0 to 1, higher value indicates better translation quality.

for evaluating the alignment between text and visual content CLIP score were utilized which measure the cosine similarity between image and text embeddings. CLIP score ranges from -1 to 1, higher values indicate better alignments.

To evaluate the mode capabilities for multimodalities and vision languages tasks, we tested the model against MME benchmark to generate scores for different perception

and cognition abilities. The MME benchmark is standard framework to compare the different models and configurations in multimodalities task.

4.3 Experiment Setup

We have conducted a total of 10 experiments divided on three groups. Each group has a different approach, and it is carried out by different individual. All sets of the experiments followed a systematic approach to ensure fair and correct evaluation of the model.

- Group 1 (Experiments 1 - 4): Focused on implementing fine-tuning with reduced model architecture configurations and different LoRa parameters.
    - Experiments 1-2: decreased configurations with default LoRa parameters.
    - Experiments 3-4: decreased configurations with enhanced LoRa parameters.
- Group 2 (Experiments 5 - 8): Focused on increasing the model configurations and model size with different LoRa parameters.
    - Experiments 5 - 6: experiment 5 Large configurations with default LoRa parameters, experiment 6 large configurations with increased LoRa parameters.
    - Experiments 7 - 8: experiment 7 Extra-large configuration with default LoRa parameters, experiment 8 Extra-large configurations with increased LoRa parameters.
- Group 3 (Experiment 9 - 10): default model configurations with different LoRa setups and extended training steps.

All experiments followed the same training arguments with 3 epochs except experiments 9 and 10 which intentionally used 6 epochs of training to see the effect of increasing the training steps. the fine tuning on the same dataset with 80% for training and 20% for test purposes. The final evaluation done by comparing the results of the computed training and testing cross-entropy loss, the ROUGE score, BLUE score, CLIP score, and MME benchmark results. The model checkpoints saved every 100 steps, and the best performing checkpoint selected for final evaluation.

4.4 Results Comparative Analysis

The experimental evaluation of the InstructBlip fine-tuned models across the ten different configurations resulted in significant insights about the model size and LoRa parameters as well as the effect of training duration on performance.

Figure 2 shows the training loss of all different training setups to compare the model learning process, which helps in detecting potential overfitting or any issues during the training.

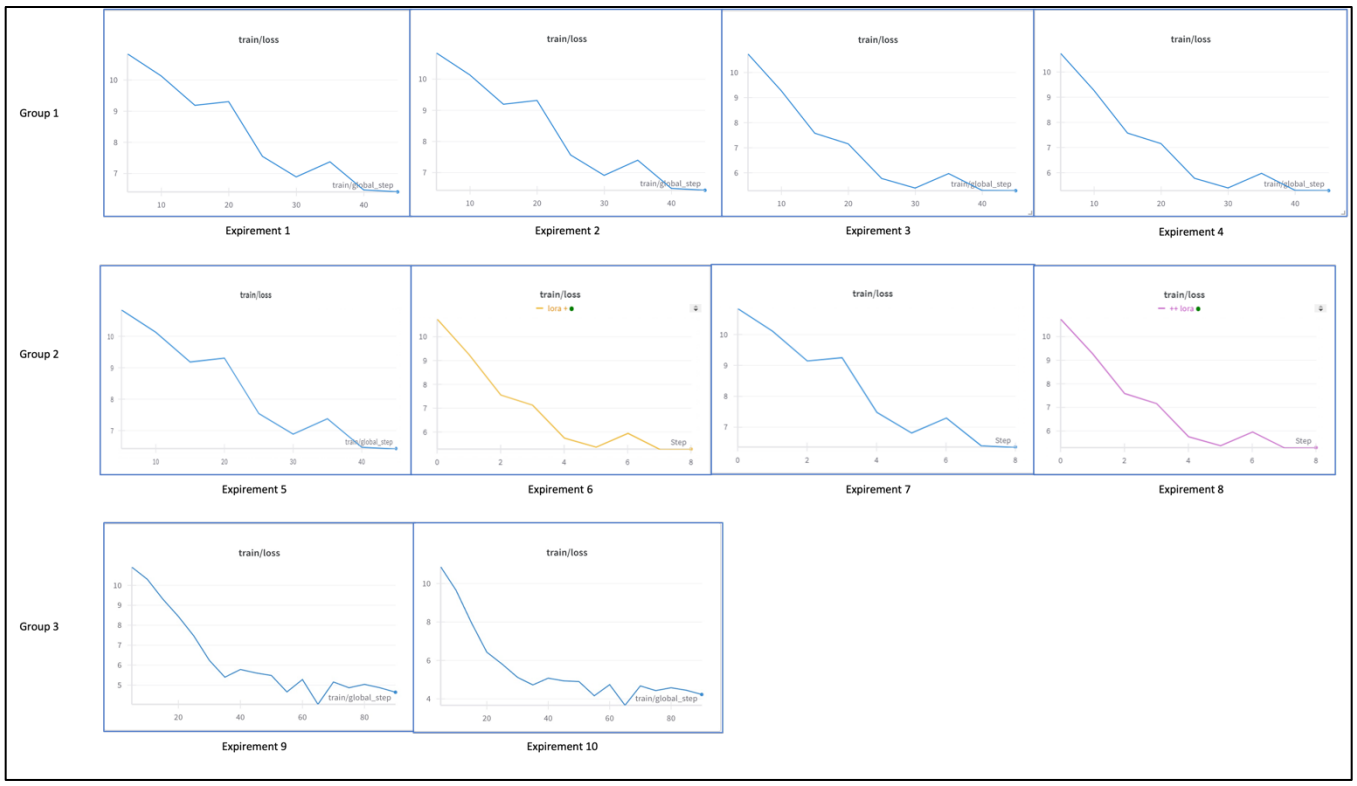

Figure 2 Train/loss

Figure 2 shows that the LoRa-adapted model with default configurations has achieved the best training results due to longer training, but in the test results, that model was the best performing, which indicates signs of overfitting with increasing the training epochs.

The following table lists the performance results of different model configurations to compare the results.

Experiment	Configuration Description	Test Loss	ROUGE-1	ROUGE-2	ROUGE-L	CLIP Score	MME Overall
1	Medium model | Default LoRA	5.674	0.2285	0.1039	0.1924	0.2657	1315.15
2	Small model | Default LoRA	5.7099	0.2282	0.1037	0.1928	0.2663	1339.66
3	Medium model | Enhanced LoRA	3.2433	0.2292	0.1057	0.1956	0.2647	1321.87
4	Small model | Enhanced LoRA	3.2548	0.2282	0.1044	0.1932	0.2638	1323.35
5	large model | Default LoRA	5.8116	0.2209	0.0971	0.1818	0.2644	1339.66
6	large model | Enhanced LoRA	3.2817	0.2198	0.0960	0.1813	0.2619	1322.58
7	Extra-Large model | Default LoRA	5.6749	0.2171	0.0955	0.1788	0.2652	1332.86
8	Extra-Large model | Enhanced LoRA	3.2839	0.2184	0.0966	0.1808	0.2616	1323.60
9	Default model | Default LoRA | Extended training	4.8339	0.1434	0.0463	0.0935	0.2671	1294.84
10	Default model | Enhanced LoRA | Extended training	4.5713	0.1193	0.0315	0.0794	0.2528	1262.80

Table 1 Performance Comparison of Different Model Configurations

The key findings based on the comprehensive analysis are the following:

- The optimal configuration was achieved with enhanced LoRa parameters (experiment 3), which gave the best balance between the performance metrics and the MME benchmark results.
- The impact of the enhanced LoRa configuration (r=64, alpha=128) achieved lower test loss among all experiments compared to the default LoRa configuration (r=32, alpha=64); the improvement in the test loss results by the enhanced LoRa models reached almost 43%.
- The extended training (experiment 9–10) shows mixed results where it improves in some specific tasks, but it shows a possibility of overfitting in the other test metrics.
- There was no one model that excelled on all metrics; different configurations revealed good results in specific tasks while decreasing on other tasks.

These findings give a valuable insight for balancing the trade-off between increasing the model's size and achieving good results in fine-tuning for specific tasks, specifically with applying Low Rank Adaptation.

4.5 Ablation Study

To gain more insights into the contribution of PEFT to the overall performance, we conducted an ablation study of specific LoRa modifications by isolating other configuration effects to have an insight for future model optimization.

The impact of different LoRa parameter settings while keeping others unchanged gives us the following results:

Model Size	Standard LoRA Test Loss	Enhanced LoRA Test Loss	Improvement (%)
Small (Exp 2 vs. 4)	5.7099	3.2548	43.0%
Medium (Exp 1 vs. 3)	5.6740	3.2433	42.8%
large (Exp 5 vs. 6)	5.8116	3.2817	43.5%
Extra-large (Exp 7 vs. 8)	5.6749	3.2839	42.1%
default Extended (Exp 9 vs. 10)	4.8339	4.5713	5.4%

Table 2 Impact of LoRA Parameter Variations

The improvement percentage is calculated as:

$$\text{Improvement}\ (\%) = \frac{\text{Standard Loss} - \text{Enhanced Loss}}{\text{Standard Loss}} * 100$$

The enhanced LoRa configurations (rank=64, alpha=128) outperformed default LoRa (rank=32, alpha=64), with improvements exceeding 42% in all experiments except the extended training configuration.

The extended training shows the lowest improvement in test loss with only 5.4%, which gives a strong sense of overfitting with the extended training epochs.

5. Conclusion

This project explored the Instruct BLIP vision-language model with a focus on improving its efficiency for use in environments with limited computational resources. By applying the Low-Rank Adaptation (LoRA) method, we were able to compress the model and reduce the number of trainable parameters while aiming to preserve at least 70% of its original performance. Several configuration settings of the model architecture were explored, including modifications to the Q-Former, image encoder, and selected layers of the language model. We tested different values for architectural parameters such as hidden size, attention heads, number of layers, and intermediate size, as well as LoRA-specific parameters like rank and alpha.

The enhanced model was fine-tuned using the VL-RewardBench dataset, which includes 1,250 samples across various vision-language tasks. Performance was evaluated using ROUGE metrics for text quality and CLIP scores for visual-textual alignment. Additionally, the model was benchmarked using the MME suite to assess its multimodal capabilities. Our results demonstrate that LoRA can be effectively used to fine-tune large models with minimal performance degradation, making them more practical for real-world deployment on devices with limited processing power.

6. References

Dai, W., Li, J., Li, D., Tiong, A. M. H., Zhao, J., Wang, W., Li, B., Fung, P., & Hoi, S. (2023). InstructBLIP: Towards general-purpose vision-language models with instruction tuning. arXiv. https://doi.org/10.48550/arXiv.2305.06500

Li, L., Wei, Y., Xie, Z., Yang, X., Song, Y., Wang, P., An, C., Liu, T., Li, S., Lin, B. Y., Kong, L., & Liu, Q. (2024). VL-RewardBench: A challenging benchmark for vision-language generative reward models. arXiv. https://huggingface.co/datasets/MMInstruction/VL-RewardBench