DAN2

An Image Search System with Natural Language Queries

by

Leung King Suen, Wang Zekai and Chen, Jia Wei

DAN2

Advised by

Prof. Dan XU

Submitted in partial fulfilment

of the requirements for COMP 4981

in the

Department of Computer Science

The Hong Kong University of Science and Technology

2023-2024

Date of submission: April 17, 2024

Abstract

Humans understand visual and language information through learning while growing up. An individual can answer what he/she sees. A machine learning model can go through the same multi-modal learning to establish vison-language alignment (VL-Alignment). However, the previous model at VL-Alignment is either course-grained, thus insufficient for answering complex queries or requires adaptation on specific tasks with structured input data. In this project, we developed a machine learning model that can answer natural language queries on images. Our vision-language model consists of a LLM for its language capability, vision encoder and an adaptor for mapping the visual features to language domain. Our model shows promising zero-shot performance on VQA tasks after training on only three million training examples of which less than 1% of examples tailored for the VQA task. Moreover, it shows understandings on the relationships between objects after we incorporated object relationships information into the fine-tuning. At the end, we provide a demonstration that allows users to search images and talk to the model with an image using natural language.

Table of Contents

1 Introduction

1.1 Overview

Natural Language Queries on image is the process of querying the content in an image using natural language through a system. To support this task, the system would have to understand natural language meaning and the content in the image through Vision-Language Alignment (VL-Alignment). However, current attempt at VL-Alignment is mostly coarse, while it is sufficient for simple query like generating short caption. It is not fine-grained for answering/reasoning on the more complex relationships for the objects in an image. Therefore, there is the need to improve VL-Alignment to enable support for more complex tasks.

Improving computer understanding in the vision or language domain alone is a heated topic. In Computer Vision, Vision Transformer (ViT) has been developed to extract image features for downstream tasks [1]. Yet, these tasks typically require specific fine-tuning and structured input rather than natural language, which limits its capability to various tasks. On the other hand, Large Language Models (LLM) has shown impressive understanding of natural language [2]. However, there is no feature embedding that maps image features into the language domain. If one could bridge both domains with an adaptor, it is possible to create a fine-grained VL-Alignment.

Therefore, in this project, we attempted to develop a machine learning model consisting of ViT, LLM and an adaptor for creating fine-grained VL-Alignment. In addition, we have demonstrated that such alignment is sufficient to support the LLM in answering queries on images. This would be demonstrated through the model answering queries of various

complexity of which can be summarized into machine learning tasks such as Image Captioning and Visual Question Answering (VQA) [3] using natural language prompts.

1.2 Objectives

The goal of this project is to create a machine learning model that achieves fine-grained VL-Alignment, in which the alignment would allow the model to answer various natural language queries on images. To reach this goal, we have been working on the following objectives.

1. Study on previous vision language models' architectures and alignment strategy on achieving fine-grained VL-Alignment.
2. Design the model architecture consisting of an image feature extractor, LLM and an adaptor. The adaptor acts as the core for aligning the image feature extractor and LLM.
3. Develop and train the adaptor that maps image feature extractor's output to the input of LLM for initial VL-Alignment.
4. Fine-tune the adaptor for advanced VL-Alignment that can answer natural language queries accurately.
5. Perform evaluation on the model's performance by quantitative benchmark and qualitative methods. Evaluate the effectiveness of the trained VL-Alignment.
6. Develop a project demo with the model inference pipeline integrated. It has a user interface for accepting user queries.

The biggest challenge we expect to face will be fine-tuning our model to improve its vision-language alignment for various image-related tasks with natural language as prompts. To address this, we will carefully study the structure of current vision-language models and related literatures.

1.3 Literature Survey

There are multiple approaches in establishing a VL-Alignment. The approaches can be categorized as encoder-based approaches and adaptation method on LLMs.

1.3.1 Encoder-based approaches

Encoder-based approaches normally involves training of encoders too align the feature embeddings output to the same feature space. For example, CLIP [4] presents a two-encoder structure, with one image and one text encoder. The two encoders were contrastively pre-trained on image-text pairs to learn to output same feature representation, thus establishing an alignment in between image and text features. CLIP demonstrated particularly good zero-shot ability when adapted to downstream tasks such as Image Classification or Image Retrieval (IR) [4]. There are projects that utilized CLIP VL-Alignment capability to conduct Image Retrieval (IR) by text in different scales of data and implementation [5] [6]. However, the zero-shot ability comes at a cost. Training CLIP is resource intensive. It was trained on 400M image-text pairs [4] while achieving, of course, image-level alignment. While this level of alignment is sufficient for IR tasks, it is not fine-grained for querying tasks that require detailed understanding of the image.

One surprise of CLIP is its image encoder was found to be efficient and powerful, thus it appeared in other projects. BLIP-2 is another model that improved the VL-Alignment [7]. Instead of training two encoders entirely, its image model incorporates frozen visual encoder from CLIP and train on a fusion transformer solely called ‘Q-Former’. It is a cross-attention layer trained on merely 129M image-text pairs and

achieved better alignments than CLIP. It has shown state-of-the-art performance on Image Captioning tasks [7].

The image model captioning capability of BLIP-2 then become foundations for other projects Another model, Caption Anything feeds BLIP2's image model with segmented image output segmentation model [8]. So, the captioner runs on different region of the image to generate multiple captions. The captions are then concatenated and fed into ChatGPT for embellishment. The generated caption is a detailed description on all parts of the image with the knowledge added by ChatGPT. However, a potential drawback is representing image information with English text. It uses solely text output from the captioner for its image information as opposed to the raw feature representation. We believe this could introduce a bottleneck if the text from the captioner is not detailed and the LLM may just be speculating without taking the actual image into consideration. Yet, this project inspired us on leveraging the natural language capability in the LLM for this project.

The takeaways from these encoder-based models displayed the possibility and idea of achieving VL-Alignment through aligning embedding feature spaces. They also showed the strategy to improve the alignment with the help of language models. In fact, we are planning to bring BLIP-2 into our design for preparing image embeddings.

One thing worth noting is that all the encoder-based alignments and their derivatives do not support query input at all. They can only generate Image Captions with no user query involved. To put it into the context of our project, this would mean our model

would only support one type of query (describing image only), which is undesired. Therefore, we continue to study other methods involving language models.

1.3.2 Adaptation Approaches with LLMs

Large-scale Language Models (LLMs) have shown impressive understanding, reasoning, and language capabilities. Therefore, there are works to align the output of visual encoder with a frozen LLM to take advantage of its capability. These works create a multi-modal extension to the LLMs and fine-grained understanding in the image context.

LLaMA-Adapter extended LLaMA with a lightweight adapter with zero-initialized attention and gating layer to a pre-trained ViT [9]. This enables LLaMA to accept input in multi-modalities (most importantly images) and have shown superior performance on ScienceQA [10] and Image Captioning benchmarks, at the same time introducing relatively little learnable parameters (1.2M).

LLaMA-Adapter extended LLaMA with a lightweight adapter with zero-initialized attention and gating layer to a pre-trained ViT [9]. This enables LLaMA to accept input in multi-modalities (most importantly images) and have shown superior performance on ScienceQA [10] and Image Captioning benchmarks, at the same time introducing relatively little learnable parameters (1.2M).

A model like MiniGPT-4 [11] is even simpler in the sense that apart from pre-trained ViT and Q-Former, it only has a single linear projection layer before the advanced Vicuna large language model [12]. So MiniGPT-4 only requires training a linear layer to align the visual

features with the Vicuna to support detailed image descriptions generations and generation capabilities with multi-modal image input.

A model like MiniGPT-4 [11] is even simpler in the sense that apart from pre-trained ViT and Q-Former, it only has a single linear projection layer before the advanced Vicuna large language model [12]. So MiniGPT-4 only requires training a linear layer to align the visual features with the Vicuna to support detailed image descriptions generations and generation capabilities with multi-modal image input.

The takeaways from these adaptation approaches show the potential to utilize a LLM as the unified prompting interface for accepting various natural language queries related to image content, which solves the querying capability issue left by previous discussed encoder-based approaches. These models also demonstrate the capability to enable various image related tasks such as VQA and achieve a more fine-grained VL-Alignment through different adaption designs.

2 Methodology

2.1 Design

2.1.1 Overall Inference Pipeline Design

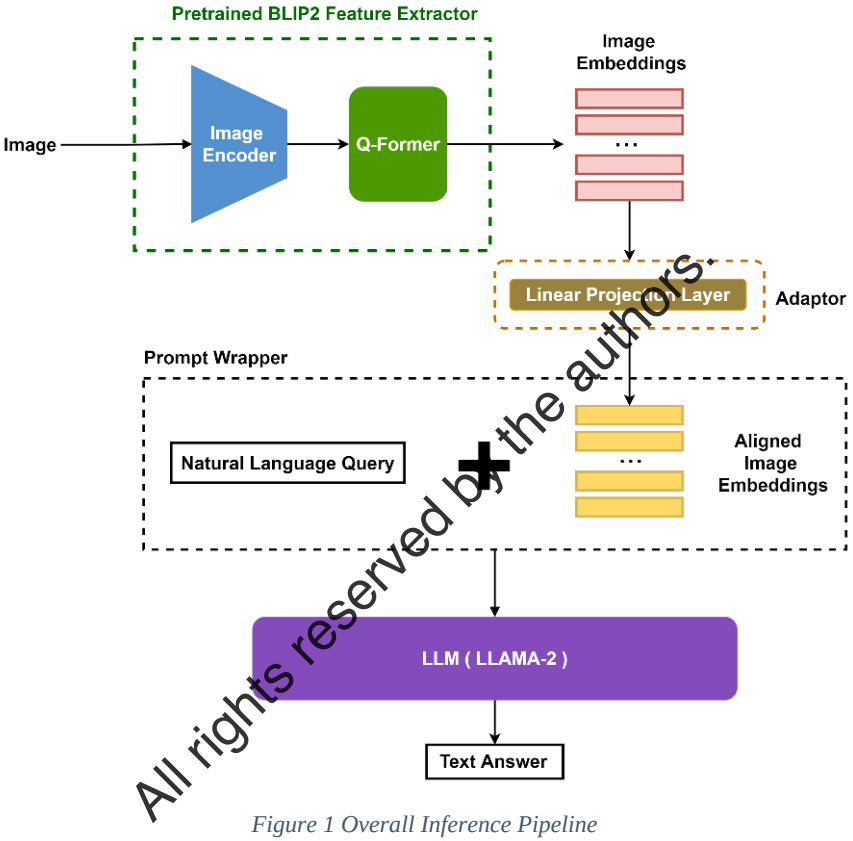

Figure 1 Overall Inference Pipeline

Our model takes an image together with a natural language query as inputs. The image is encoded to image embeddings by the Pretrained BLIP2 Feature Extractor [7]. The image embeddings are then aligned with a LLaMA-2 LLM through the adaptor [13]. The natural language query that may be a question or instruction is wrapped with the alignment image embeddings as prompt to the LLaMA-2 LLM to generate the answers. To keep the naming short, we will be referring 'LLaMA-2 LLM' as 'LLM' unless the full name is required for the context.

2.1.2 Pretrained BLIP-2 Feature Extractor

To extract visual features from an image into image embeddings, we utilize a pretrained BLIP2 Feature Extractor [7]. The extractor is made up of an Image Encoder (eva-clip-g) and a fusion transformer called ‘Q-Former’. The ‘Q-Former’ has undergone VL-Representation Learning, which allows it to extract language-informative visual representation. By using this extractor, only more informative features are fed into the adaptor, making it more effective during training. To keep the naming short, we will be referring 'Pretrained BLIP2 Feature Extractor' as ‘Feature Extractor’ unless the full name is required for the context.

2.1.3 Adaptor Design

This adaptor is a model that does feature mapping to help a LLM understand the visual features from upstream. It takes the image embeddings from the Feature Extractor, maps it to the feature space of the LLM, and finally output the aligned embedding. We designed and trained a linear projection layer to do the mapping.

2.1.4 Prompt Wrapper

We designed this module to combine the aligned image embeddings with natural language query as one input to prompt the LLM. It takes the aligned image embeddings as a prefix with the natural language query concatenated at the back. Finally outputting a prompt wrapped with the LLaMA-2 prompt template [14].

2.1.5 Dataset Design

We have chosen COCO Captions [15] and Conceptual Captions [16] datasets for training the adaptor. The two is concatenated into one big dataset of about three million in size. Dataset of

this scale is required since we want to learn as many visual features mapping as possible. The samples in the dataset are image-caption pairs. The two is concatenated into one big dataset of about three million in size. Dataset of this scale is required since we want to learn as many visual features mapping as possible. The samples in the dataset are image-caption pairs. For each pair, it has more than ten tokens per caption which is detailed enough for the adaptor to learn fine-grained feature mapping.

2.1.6 Adaptor Training Algorithm

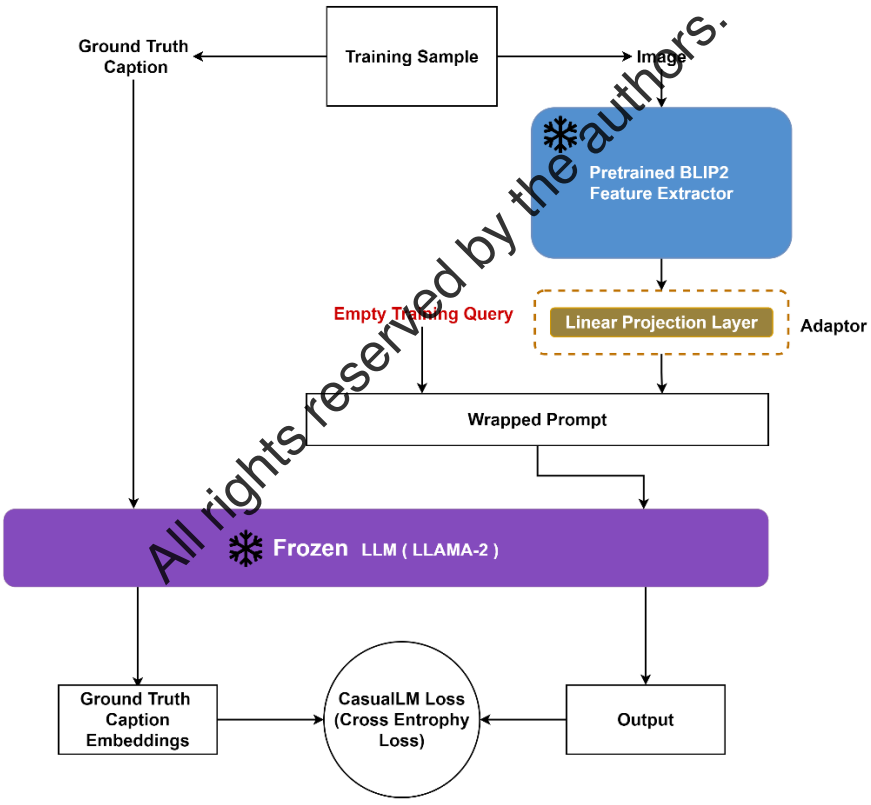

Figure 2 Adaptor Training Pipeline

The training objective is to minimize the Casual Language Modelling loss in the LLM (which is usually the Cross-entropy Loss between the sample and output logits) [13]. The parameters in the LLM and Image Encoder are frozen to save computing resources and training time. Weight updates only take place in the adaptor.

Image sample would follow the usual inference pipeline. However, the training query is kept empty in the training. The rationale behind this is that, if we added query like 'Describe this image' in the training query, the mapping learnt by the adaptor is bound to this specific query. The LLM is bounded to that query when inferring the meaning from the aligned features. This is undesired in the inference time since the query could be different. Essentially, the adaptor learns the image features with only image as input, the query to prompt the LLM generate different answers related to learnt image features.

2.1.7 Design Fine-tuning Strategy on Adaptor

Only training is not sufficient for our model to understand natural language query on image content. For the capability on answering queries to different vision-language tasks such as Image Captioning and VQA, we need to fine-tune the adaptor using the same algorithm described in the previous section **2.1.6**. Except that the prompt is now natural language instructions or questions related to image content from the datasets in section **2.1.8**.

2.1.8 Design the Dataset for Fine-tuning

We have combined the QA and Relationship sub-datasets from Visual Genome [17, 17], together with a filtered image captioning dataset prepared by MiniGPT-4 [11], we refer this dataset as MiniGPT-4 Caption in the following.

-Visual Genome QA. This dataset was used for VQA task, it covers various image-content related queries such as 'What is the man doing?', these queries were used to fill in the previously empty prompt part in the training query and the LLM was expected to learn to give answers corresponding to these questions. However, this dataset does not contain information

about relationships between objects in the images such as ‘a man IN FRONT OF the bus’ or ‘a catcher WEARING a helmet’.

-Visual Genome Relationship. This dataset was added to improve the VL-Alignment of the adaptor in terms of understanding object relationships in the image and answer queries related to object relationships. A list of customized object relationship related natural language instructions were prepared for wrapping the training query for each sample in the dataset.

-MiniGPT-4 Caption. This dataset was added for more detailed Image Captioning task. It has much larger tokens per caption (~60 tokens per caption) than the dataset we used for training. A list of customized image caption-related natural language instructions was wrapped into the training query and the LLM was expected to learn to generate longer captions for images.

2.1.9 Design a demo

When all the model developments are completed, we will have final demo showing the model’s conversation performance in responding various user queries. This demo allows the model to be used intuitively without touching development environment (Jupyter Notebook or command line interface (CLI)). It has a user interface for the user to upload or search images for the users and the model to chat on. The demo forwards the image and user query to the same backend inference pipeline in **Figure 1**.

2.2 Implementation

2.2.1 Dataset Preparation and Pre-processing

We first downloaded the COCO Captions dataset and the Conceptual Captions dataset separately in the form of webdatasets using image2dataset library [18]. Then the two datasets were merged. The unnecessary data fields were filtered out, only images and their associated captions were kept for our training. We were not able to download about 20% examples in the Conceptual Captions due to broken image link, thus the final example count was a little under the three million targets. There were 2,956,145 examples to be exact.

In the training, the images were first resized to 224 x 224 and then converted from PIL format into tensors. The captions were converted to lowercase, filtered to remove punctuations other than comma.

2.2.2 Implement the Model and Forward-Backward Passes

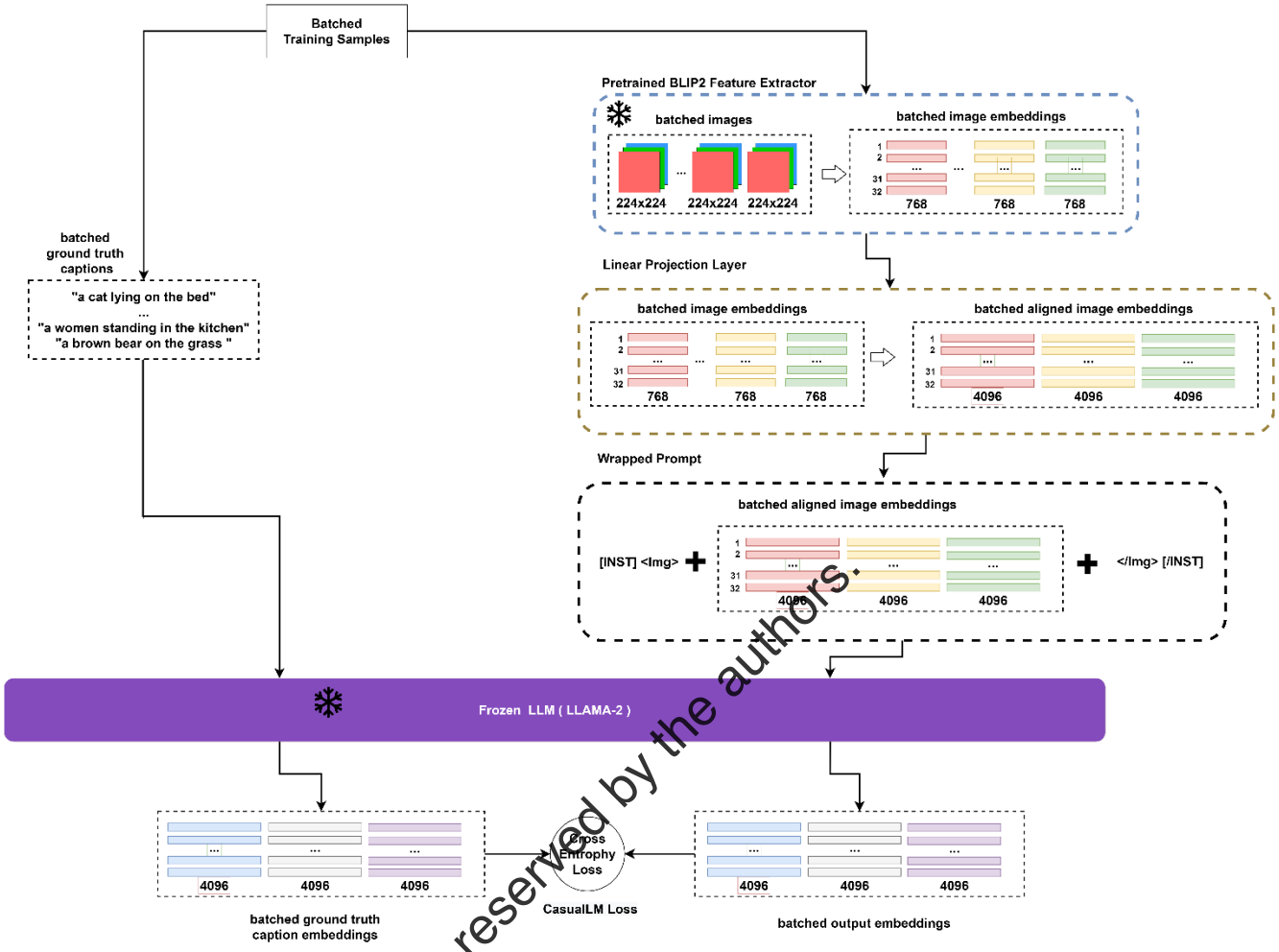

Figure 3 Model Training Flow

Each module is implemented and integrated using PyTorch. We used Lavis's implementation for the Feature Extractor [19] and Hugging Face's implementation for LLaMA-2 LLM [20] together with their pretrained weights. Then, we implemented the linear projection to map the Q-former's 32 × output tensor $\mathbf{Z} \in \mathbb{R}^{32\times768}$ to $\mathbf{Z}' \in \mathbb{R}^{32\times4096}$ for the correct dimension of LLaMA-2 input. This results in a total of 768 x 4096 + 4096 = 3,149,824 trainable parameters (with biases).

During the forward pass, the batched training samples are forward to the Feature Extractor and transformed to batch image embedding tensors of shape $\mathbf{Z}^{(\boldsymbol{batched})} \in \mathbb{R}^{batch\times32\times768}$. Then the linear projection layer (Adaptor) maps the batched image embedding tensors to the

feature space of LLM, which outputs a batched aligned embedding tensors of shape $\mathbf{Z}'^{(\boldsymbol{batched})} \in \mathbb{R}^{batch \times 32 \times 768}$. The aligned batched embeddings are wrapped into a prompt and used as inputs for the LLM. As elucidated in **section 2.1.6**, the prompt in training and fine-tuning is different, so the LLaMA-2 template wrapped prompt [21] as followed:

Prompt in Training:

<s>[INST] <Img> Aligned Embeddings </Img> [/INST]

Prompt in Fine-tuning:

<s>[INST] <Img> Aligned Embeddings </Img> {Natural Language Query} [/INST]

LLM takes the above inputs and generate the batched output embedding tensors. These embedding tensors are finally used to calculate the loss between batched ground truth caption tensors using cross-entropy loss implemented by the LLM's CausalLM Loss [20].

During the backward propagation, the gradients that needed to be updated for each module of the forward pass is calculated and scaled according to the loss obtained. Note that we have frozen both the Feature Extractor and LLM in the forward pass process, so only the linear projections layer (Adaptor) 's weights will be updated.

2.2.3 Training with Large Model and Dataset

There were adjustments we need to make to battle the resource constraint when training a large model on a large dataset.

First, we estimated that the GPU memory usage required was about 15.0 GB with all modules loaded in a forward pass with the LLM taking the most memory (~13.6 GB). This usage will

be higher when invoking the backward pass during the training. Therefore, it is not possible to run or train model this size on a normal home personal computer, thus we would need the **HPC3 cluste**r offered by HKUST to train our model. The node we used HPC3 has 10 GPUs in parallel [21]. We implemented **Distributed Model Parallel** which allows the training of larger models [22]. The technique allowed us to distribute LLM's parameters, gradient and states to 8 GPUs evenly. With such a technique, we were able to train the model with a larger batch size and increase the inference speed.

However, there is a 24-hour wall-time limit on HPC3 which prevents us to train on one node continuously. To tackle the time limit, we need a checkpoint mechanism which allows the model to save the weights in the middle of training and resume afterwards. To achieve this, we stored the last trained index together with the latest states of the scaler, optimizer, learning rate schedule and the adaptor's weights to a PyTorch state dictionary as a training checkpoint. There is a 24-hour timer set when the training starts, it will pause the training and save the training checkpoint right before it reaches the wall time limit.

At last, due to sheer size of dataset. It was not possible to complete one epoch within 24-hour time limit. In fact, we observed that 48 hours was needed on average for training one epoch, thus a similar checkpoint mechanism is required to be implemented for the data loader. We kept tracking the index of current sample. When the training resumes, we restarted the data loader from that index and continue to the end of the dataset. Moreover, since the training is broken into 2-halfs, it is hard to keep track of the samples' indexes while shuffling the samples from data loader. We have found an easier implementation, that is to shuffle the dataset using Hugging Face dataset library [23] before feeding it into PyTorch data loader and the seed to shuffle the data is different for each epoch.

To further improve the training efficiency, we have also utilized mix precision training [24] with gradient scalar implemented by PyTorch to cast the tensors to appropriate datatypes according to different operations during model training. For example, linear layers and convolutions, are much faster in float16 or bfloat16 while reductions often require the dynamic range of float32. This could reduce the training time and the memory usage during our training.

2.2.4 Training Details

After we overcame the challenges of training on limited resources, we trained on the dataset mentioned in section **2.2.1** for 4 epochs. The initial learning rate is 1e-6, then it warmups linearly to 1e-4 during the first epoch [25]. After that it is controlled by cosine annealing [26] scheduler with final learning rate of 8e-5. We used AdamW [27] as optimizer and a batch size of 192. All training runs on the HKUST HPC3 inode-145, which has 8 NVIDIA RTX 6000 GPUs [21]. The time used in training is around 150 hours.

2.2.5 Dataset Preparation for Fine-tuning

We first downloaded the QA, Relationship sub-datasets from Visual Genome using Hugging Face's dataset library [28] and the Caption dataset from MiniGPT-4's GitHub page [11]. After downloading, we read the above three datasets separately and converted each sample to a new dictionary containing three keys: *image*, *query*, and *answer*.

The value for each key was obtained differently for different datasets:

-Visual Genome QA. We could directly obtain the corresponding key-value pairs from the data fields in this dataset.

-Visual Genome Relationship. We could obtain the *image* field, but for *query* field, we created a list of queries such as "What is the relationship between *subject-A* and *object-B*?" and random selected one for each new sample, where *subject-A* and *object-B* are the data fields in this dataset. For the *answer* filed, we concatenated three fields *subject-A*, *predicate*, *object-B* together as the *answer* field for each new sample where *predicate* represents relationship between *subject-A* and *object-B*, it could be "in front of", "wearing", etc.

-MiniGPT-4 Caption We obtained the *image* field and for *query* field, we created a list of queries like "Describe this image in detail." and random selected one for each sample. And for the *answer* filed, we simply used the *caption* field for each sample.

After converting each sample following the above processing pipeline, we decided form a fine-tuning dataset of 10000 samples with 20% samples from Visual Genome Relationship, 20% samples from MiniGPT-4 Caption and 60% samples from Visual Genome QA. We included more samples from Visual Genome QA since it covers a wide range of image-related natural language queries while the other two datasets were added for improving the VL-Alignment on answering more specific image-related tasks.

During the finetuning, the images were first resized to 224 x 224 and then converted from PIL format into tensors while the queries and answers were converted to lowercase.

2.2.6 Finetuning Implementation

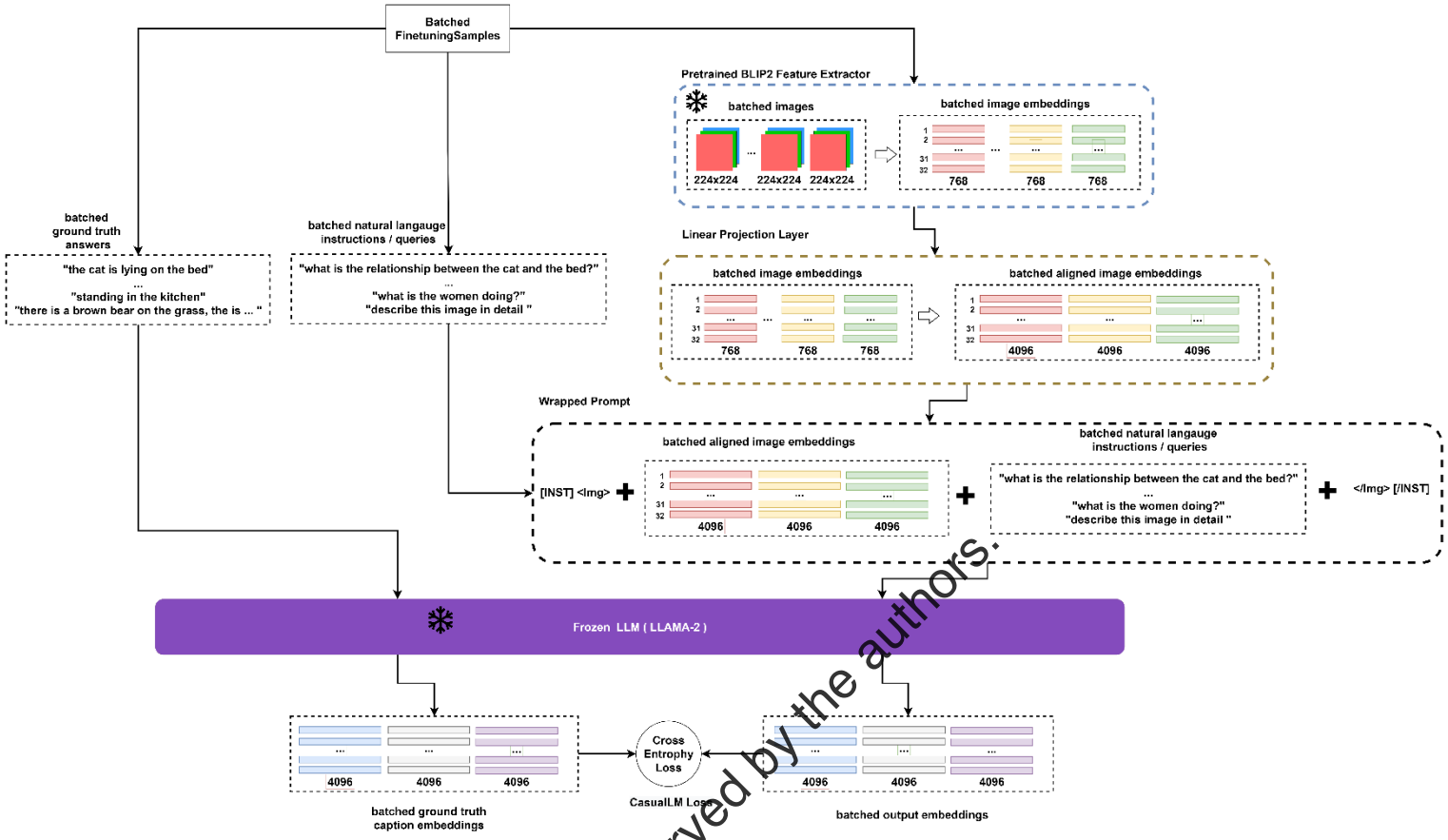

Figure 3 Model Fine-tuning Flow

The finetuning follows a similar implementation as the training of adaptor described in section **2.2.2 Implement the Model and Forwards-Backward Pass** and **2.2.3 Training with Large Model and Dataset**. The complete finetuning pipeline is illustrated as above. The main difference is that we concatenated the natural language instruction / query from the finetuning dataset to the wrapped prompt. For Feature Extractor and LLM, they were still frozen and only the linear projection layer (Adaptor)'s weights were updated.

2.2.7 Finetuning Details

We trained on the dataset mentioned in section **2.2.5** for 5 epochs. The initial learning rate is 1e-6, then warmup linearly to 3e-5 during the first epoch [25]. After that it is controlled by

cosine annealing [26] scheduler with final learning rate of 1e-5. We used AdamW [27] as optimizer and a batch size of 12. The time used in training is around 50 minutes.

2.2.8 Demo Implementation Details

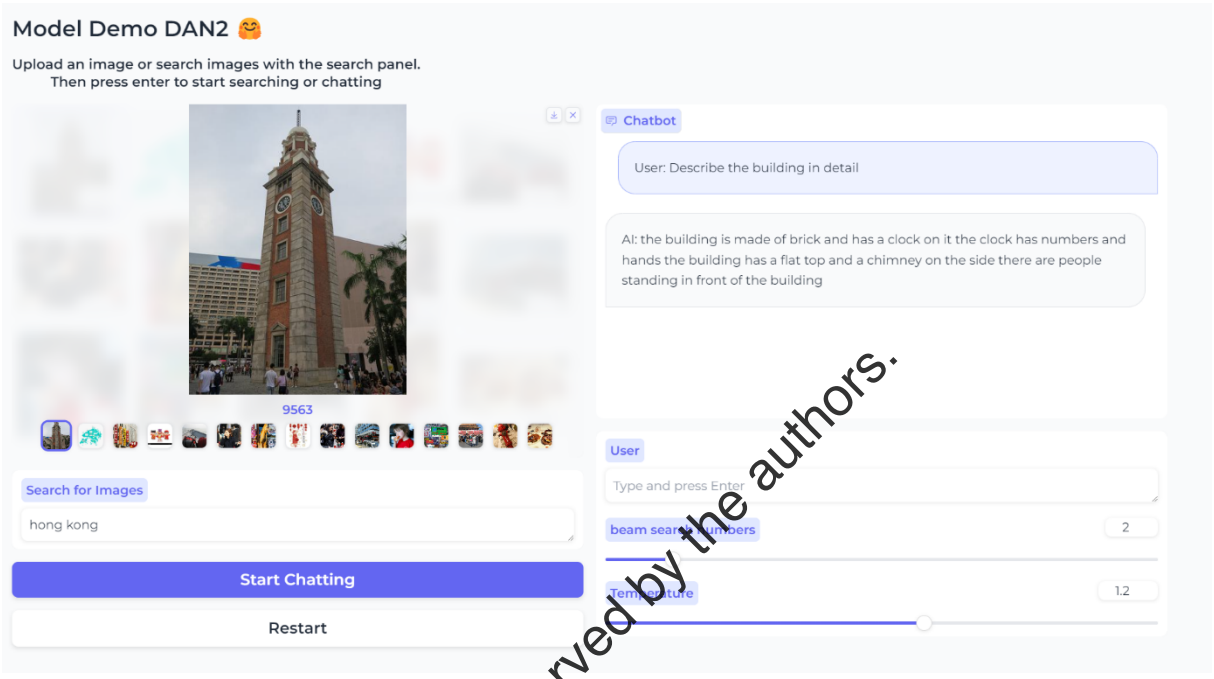

Figure 5: Demo built with Gradio

We utilized Gradio [29], an open-source Python package to build our demo and host it locally on a webpage through a PC. To recall, this demo can do image search and accept/answer queries on image content from the user.

For the image search, we first implemented an image database with a directory containing 14100 JPEG images. Each entry in the database is a path to the image and image embeddings of that image. The image embeddings were pre-computed by the Pretrained BLIP-2 Feature Extractor [7]. Having those image embeddings pre-computed could speed up image retrieval since we avoided computing them on the spot during the demo.

After we have built the database, we implemented an algorithm to retrieve images based on a given text query. There will be two types of text query in the following part, one is a text query for obtaining images from image database, another is a text query for conversating with the model. To avoid ambiguity, we will name the latter as **instruction query** instead.

We utilized the same BLIP-2 Feature Extractor mentioned above to obtain text feature embeddings from the user. The text embeddings $\mathbf{T} \in \mathbb{R}^{32\times768}$ are in the same dimension to image embeddings $\mathbf{Z} \in \mathbb{R}^{32\times768}$. We computed the cosine-similarities of the text embeddings with every image embedding in the database by dot-product. Afterwards, the cosine-similarities were sorted in descending order with Numpy's "argsort" and truncated to keep only the top 15 images that have the highest similarity scores with the text query. All computation for the image search is done on CPU to save video memory for hosting the LLM.

For the conversation function, once the image to conversate on is determined by the user. We forward its image embeddings and user's instruction query to the backend inference pipeline. We also provide to controllable parameters: *number of beams* and *temperature* in the UI for customization on the model's output. The model generated response is forwarded to Gradio for display.

2.3 Testing

We conducted testing the individual modules and the forward-backward passing with all modules combined. This is essential before starting the training on the model.

2.3.1 Feature Extractor

To ensure the Feature Extractor outputs correct dimensions, we randomly selected ten images from the COCO Captions dataset, format the images into tensors of shape [10, 3, 224, 224]. Then, feed the data into the encoder, and confirmed that the output tensors are in the shape of [10, 32, 768].

As per Design, the choice of using Feature Extractor for encoding image is made since it can extract language informative visual features. Therefore, besides the dimension checking, we have conducted Image Retrieval (IR) testing to verify its basic VL-Alignment ability. This is done by checking Recall@k metric on IR task. The metric is produced by validating whether the returned image and the query text are ground truth image-text pair. The variable k specifies the top k number of images that are used in the validation, of which each of the top k images is validated against the query text to find out if they were a legit pair. The model is said to have better performance if it produces a high Recall (close to 100%). We have conducted the evaluation with k=1, 5, 10 on the COCO Captions 2017 test set.

Model	MSCOCO (1 caption only)		
	R@1	R@5	R@10
Pretrained BLIP2 Feature Extractor	57.1%	82.5%	89.0%
CLIP (pre-trained, ViT-L-14) [4]	36.5%	60.8%	71.0%

Table 1 Image Retrieval benchmark results on MSCOCO test set. R@k refers to Recall@k. Only the first caption from MSCOCO were used.

The testing result was better than CLIP, another VL-Alignment model. This suggested our Feature Extractor implementation was correct and dependable in providing image embeddings for the adaptor training.

2.3.2 Adapter (Linear Projection Layer)

To check the correctness of the Adaptor implementation, we continued examples in the Feature Extractor testing to check for the correct input/output dimension of tensors, which is [batch size, 32, 768] and [batch size, 32, 4096] respectively.

2.3.3 LLM

The function of LLM is to process image embeddings with natural language query related tasks. To make sure the model could correctly process the prompt. We firstly test the model with trivial text prompts. For example, we used the prompts like “please introduce yourself” and “tell me a story of Hong Kong.” By analysing the response, we ensured the basic performance of the LLM.

2.3.4 Model Integration Testing

Each module was integrated together and conducted testing with random weights and input sample. This is the essential since finding errors after hundred hours of training is undesired and the correction is impossible besides re-training.

1. Checking Weight-loading and Forward Pass Behaviour:

This is to verify the weight has been loaded correctly and the dimensions are matched in forward pass. The testing on the weight loading is introduced specifically since training with **Distributed Model Parallel** introduces a differently styled PyTorch state dictionary compared to models saved on a single GPU. It ensures that whenever loading the weights, it should load perfectly fine in both formats reliably. For the forward pass, we

feed a random image sample to forward pass and check for dimension mismatch and ensured that the intermediate tensors are of their expected shapes.

2. Checking for Convergence:

This to verify the forward and backward passes are implemented correctly and the training can converge. We overfitted one image-caption pair from the MSCOCO dataset intentionally, then queried the LLM with the image embedding. Upon checking the generated answers on the image and that it did match with the ground truth caption, hinting at a convergence. Finally, we were confident to start the training process.

2.4 Evaluation

Our goal for the model is its ability to answer natural language queries. We have tested it against Visual Genome dataset and VQA benchmarks for quantitative testing. For the general answering performances, we have conducted User Acceptance Tests to collect qualitative data.

2.4.1 Evaluation on the Fine-tuning Result

Before evaluating on VQA benchmarks, we conducted evaluation on the Visual Genome dataset to check the effectiveness of fine-tuning. A split of 4000 examples were randomly selected, with 3000 samples from Visual Genome QA and 1000 samples from Visual Genome Relationship. Before fine-tuning, since our model has not been trained on any type of queries. The exact match rate was less than **1%**. After fine-tuning, it achieved **35%** exact match rate. This indicated the effectiveness of our fine-tuning strategy. The model successfully learnt to respond to various natural language queries and questions specifically ask about relationships between different image objects. Furthermore, the process of training and fine-tuning would turn out be useful on zero-shot testing on VQA benchmarks as well.

2.4.2 Quantitative evaluation on VQA Benchmarks

We have used OKVQA dataset for the testing of VQA benchmarks [30]. Each of the entry in this dataset consists of an input image, a question on the image and a text answer. An example could a puppy toy image, with question: ‘What is the type of the toy in the image?’ and an answer stating the toy is of type ‘puppy’. The following shows our result along with other models’ performance on this dataset.

Model	OKVQA Accuracy	
	Zero-shot	Fine-tuned
Our model	33.21	54.57
Flamingo80B [31]	-	50.6
Flamingo9B [31]	-	44.7
PaLI-X-VPD [32]		66.8
BLIP-2 ViT-G FlanT5 XXL [7]	45.9	-
BLIP-2 ViT-G OPT 2.7B [7]	31.7	

Table 2 Results on OKVQA benchmark. All figures are top-1 accuracy.

Table 8 presents our experimental results on QKVQA, our fine-tuned model outperforms BLIP-2 ViT-G OPT 2.7B by 1.51% for zero-shot on OKVQA. It also outperforms both Flamingo9B and Flamingo80B by 9.87% and 3.97% with extra fine-tuning on OKVQA. These results indicated the strong visual question answering capabilities of our model.

2.4.3 Qualitative evaluation with User Acceptance Tests

We believed that one of the shortcomings of VQA benchmarks evaluation is it is not a very faithful way to measure the actual conversation or the image understanding performance of the model. We observed that sometimes our model generates longer answers where the correct word lies in different part of it. Although the answer is semantically correct upon checking the ground truth answer, yet they were deemed incorrect. We believe this is because the specific requirement in the VQA benchmark [33] where it only checks exact match of answer texts and ignores answers with similar meaning or semantically correct answers with different words/phrases.

Apart from that, another shortcoming of the testing conducted in section **2.4.1** using VQA benchmarks is that the answer from the model is kept short to be evaluated with the ground truth answer. In fact, the OK-VQA benchmark labels have an average token of 1.3 [30], as with the case with the original VQA dataset concept [3]. The model is required to generate 1-2 words answer (an incomplete sentence) to be deemed correct. However, we believe that in the real world the model would be required to generate complete sentences/conversations that require more details reasoning and background information. Therefore, to facilitate evaluation of our model with more representative use-cases. We have conducted a survey to collect responses from the users asking about their subjective thoughts on the performance of our model.

The survey has 3 types of questions, totalling at 9 query-answers examples, they are as follows:

- Question 1 to 3: Visual Question Answering.
- Question 4 to 6: Answer the relationships between objects.
- Question 7 to 9: Generating detailed description.

At last, there is one question asking on the general comments from the user. The survey can be accessed from this link.

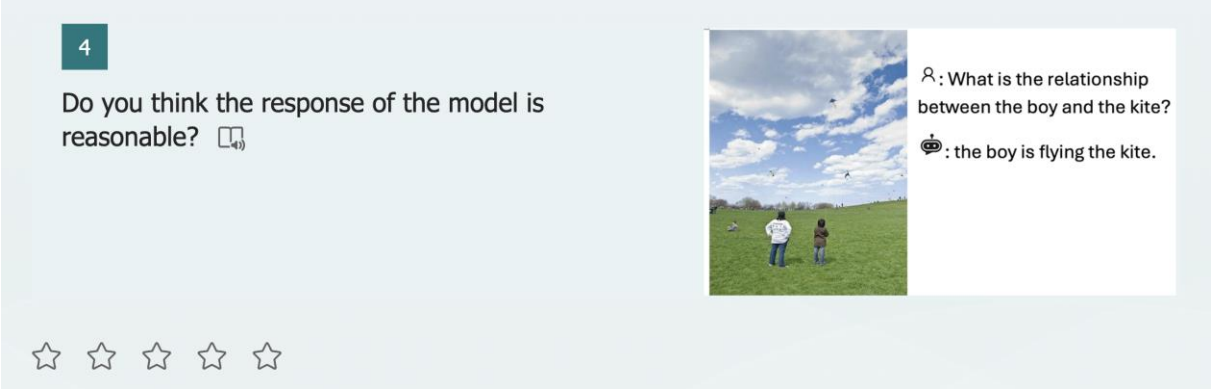

Figure 6 An example from the survey.

The user is then asked to rate on a scale of 1-5 on the model's response. We have collected the responses; they are showed in the table below:

Questions	Avg. Rating (1-5) (3 sig figs)	Questions	Avg. Rating (1-5) (3 sig figs)
Q1:	4.09	Q6:	4.27
Q2:	4.16	Q7:	3.93
Q3:	3.99	Q8:	3.74
Q4:	4.53	Q9:	3.59
Q5:	4.20	Overall	4.06

Table 3 Survey results.

A total of 72 responses were collected from our friends and schoolmates, and the overall feedback was close to a 4 rating out of 5.

There are 5 discoveries in the results that are worth mentioning:

1. The score distribution on the three tasks is uneven, with Object Relationships ranking highest, followed by VQA, and then Detailed Description. This reveals our model capabilities under different user queries.

2. Our model performs the best in finding object relationships. For example, in question 4, it could precisely identify the relationship between the boy and the kite with the correct nouns. This shows the effectiveness of fine-tuning of Visual Genome Relationship.

3. The problem with the VQA task is related to short responses. For example, in question 2, the model only gives the correct answer "skateboarding" instead of denying the query first, which to some evaluators found unnatural.

4. The answer for the Detailed Description task is unorganized and repeated. For example, in question 8, the existence of the object "remote control" was mentioned twice in two different sentences. The same appeared in question 9, the phrases "a curtain on the window" and "a clock on the wall" appeared twice. This hinted that there were not enough instructions used in fine-tuning to overcome to short and unnatural answers as of the results of training on image-text datasets.

5. All the output answers were in lowercase. We believed that this is due to the preprocessing we did which converted all texts to lowercase to simplify the training.

Overall, despite the differences in performance on the three tasks, our model was able to generate answers given image and questions. It has accurately answered more complex questions of involving the relationships between object, and is able to generate detailed descriptions on a given image. Although there are still improvement be made, of which we will discuss in the next section. We were generally satisfied about its performance which lived up to our goals and objectives.

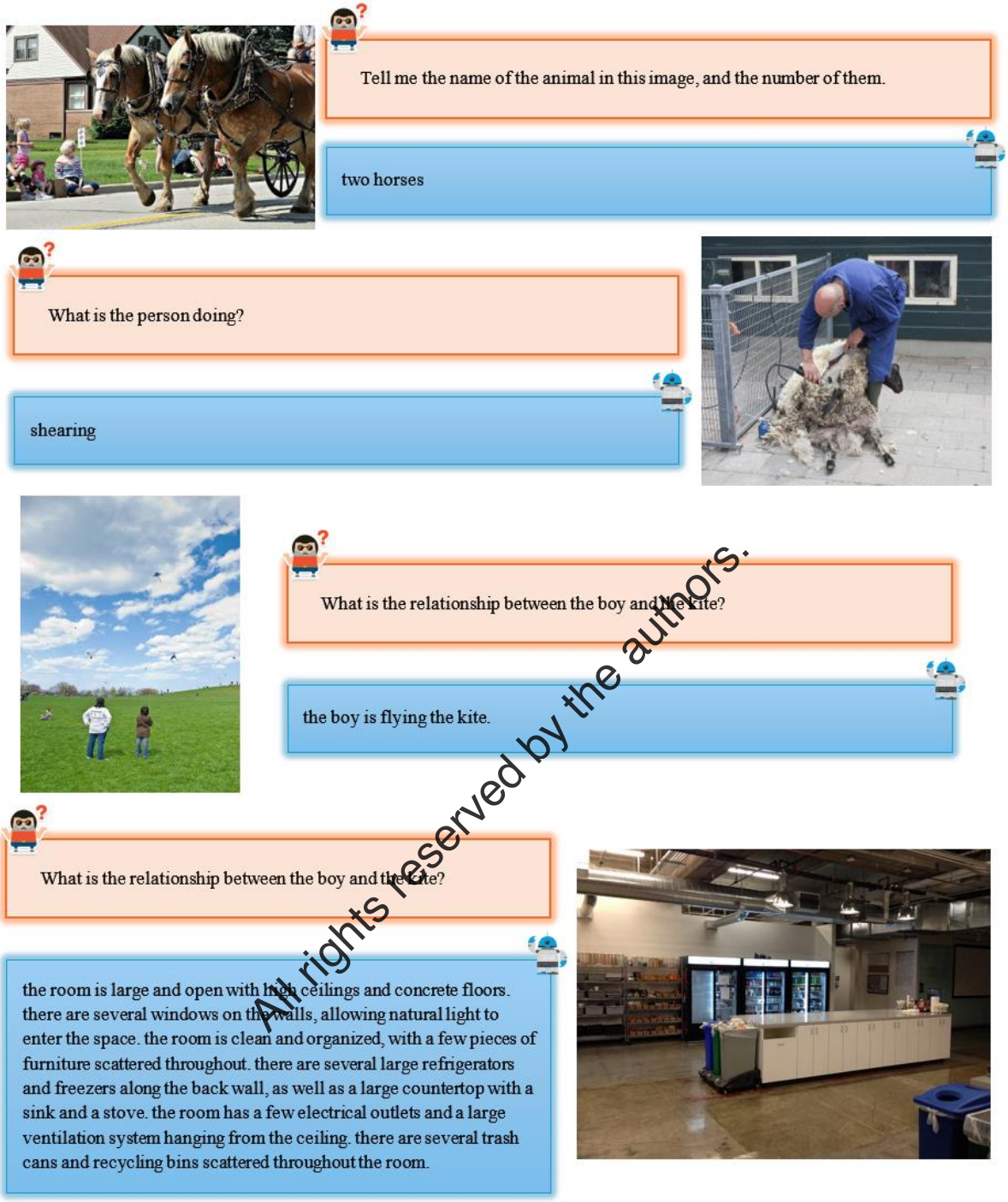

Figure 7 A collections of answers of the model on VQA, Object-Relationship and Caption Generation

3 Discussion

3.1 Poor Performances on the Referring Expression Comprehension (REC)

Other than Image Captioning and VQA tasks, originally, we also wanted our model to have Referring Expression Comprehension (REC) [34] capability. Specifically, if a user provides an object expression or phrase describing a region, the model could locate that object or region in the image and generate a corresponding bounding box for it. However, our experiment results were unsatisfactory after we attempted to fine-tune our model for this task.

In our experiment, 40000 image object/region phrase and bounding box pairs from RefCOCO [35] were used in fine-tuning the adaptor. We used the same fine-tuning pipeline discussed in section **2.1.7** and have the model generated bounding box representation in the following format:

{<top-left-point-x><top-left-point-y><box-width><box-height>}

We then evaluated the fine-tuned adaptor performance on RefCOCO validation set, the accuracy is evaluated by calculating the Intersection over Union (IoU) between generated and the ground truth bounding boxes at a threshold of 0.5. However, the accuracy was only 26.7%. This reflected that the model was not able to capture the spatial position of the objects after fine-tuning.

There are three potential reasons for the mediocre performance that we would like to further discuss:

1. **Insufficient training samples:** We believe that REC task requires an even more fine-grained alignment for capturing complex object location information and would require more training samples to be introduced into the adaptor. Our fine-tuning was only done on 40000 samples. While this is sufficient for instruction-tuning or conversation, it is insufficient for learning the even more fine-grained alignment for REC. Other similar works such as Shikra suggested at least a magnitude more examples should be incorporated [36]. They should be from dataset such as RefCOCO or Visual Genome.

2. **Different training strategy needed:** Instead of learning alignments without query in the early training, we should consider integrating more datasets for responding various tasks (Captioning, Grounding and Referring Expression Comprehension) to form even larger dataset for the early adaptor training stage, as shown by Shikra [36]. Apart from different dataset options, we believe training and fine-tuning involves tuning the LLM along with the adaptor will improvement the benchmark performances as well [11] [36].

3. **Feature Extractor Design:** We utilized the Feature Extractor in our project to extract image features that are more language informative while discarding others. However, these language informative features might not be useful for visual grounding tasks which require complex object location information. In addition, to extract only language informative image features, the Feature Extractor mapped the output from the image encoder, which has a much larger feature space, to a smaller feature space. This mapping reduced the trainable parameters of adaptor and could hinder our adaptor's capability to learn this complex object location information.

3.2 Other Design Choices

The final implemented design of the adaptor is a single layer of fully connected layer. However, there are other designs to the adaptor for mapping feature spaces. For example, we may add more layers to form a multi-layer perceptron to have more trainable parameters, which allows the adaptor to learn with a stronger representation power.

Moreover, even with the same adaptor design, the sequence of which the modules are laid in the pipeline can be changed for further experimentation. In our final design, the BLIP-2 Feature Extractor is solely used to extract image features. However, it is possible to utilize the power of Q-Former's language ability further as in the original structure of BLIP-2.

It is possible to feed the same query that we fed into LLM into Q-Former to make it task-aware [37]. This encourages Q-Former to extract the most task-relevant image features for later mapping. Therefore, not only Q-Former were extracting language-relevant image features, but it also assists in generating better image features specific to the task that the query specified. Then, by unfreezing Q-Former in training and fine-tuning, it is possible to adapt the model to the task better.

It is possible to feed the same query that we fed into LLM into Q-Former to make it task-aware [37]. This encourages Q-Former to extract the most task-relevant image features for later mapping. Therefore, not only Q-Former were extracting language-relevant image features, but it also assists in generating better image features specific to the task that the query specified. Then, by unfreezing Q-Former in training and fine-tuning, it is possible to adapt the model to the task better.

4 Conclusions

Summary of Key Technical Achievements and Lessons Learned

We conducted a comprehensive study of current state-of-the-art machine learning models with VL-Alignment capabilities. We have designed and implemented our own model that achieves a fine-grained VL-Alignment. The remarkable outcome of this project is the model's capability to properly respond to various image-related tasks with natural language queries. This has been demonstrated in the evaluation section where the model outperforms other advanced vision language model on VQA tasks. This is because the preliminary model training allows the model to learn an initial VL-Alignment from a wide range of image-caption pairs and then this learnt VL-Alignment was further improved through fine-tuning with various image-related natural language queries. At the end, we have also integrated our final model inference pipeline into a frontend UI using Gradio which allows the model to be used intuitively without touching development environment.

Issues Not Addressed in Our Project and Future Work

In this FYP, we have created a model that supports VQA and Image Captioning tasks with natural language queries. During the development, we have tried to extend the model to support other vision-language tasks such as Referring Expression Comprehension (REC) but only obtained a poor performance on this task. Due to time and resource constraints, we were not able to experiment the model's ability to learn more vision-language tasks and further improve the model's performance with different model design. On a bigger picture, the issue of model generalization to perform well on multiple tasks must be addressed to ensure it is useful for real world application. In Summary, the following items are suggested for future work to build on this project:

Model Training: Incorporate task-specific training samples in large scale for the model to learn different complex features. These samples should be applied with appropriate sampling ratio and further processed to provide a list of prompt and model output templates for the model to accept natural language query as input and generate more fluent and natural outputs.

Model Design: Our model has a light-weight design in the sense of using minimal training parameters to achieve a fine-grained VL-Alignment. We froze all other modules of the model and only train the single linear projection layer to align complex vision and language features. It is worthwhile to unfreeze the Q-Former in the Feature Extractor during training to allow the model to extract the most useful task-specific image features as mentioned in the discussion section. Moreover, it is also worthwhile to experiment with other implementations of Feature Extractor and LLM that are more powerful. For example, instead of using LLaMA-2 7B version, simply replacing it with 13B version of LLaMA-2 or Vicuna, an instruction-tuned version of LLaMA-2, may already improve the model's performance.

In conclusion, our model has succeeded in achieving a fine-grained VL-Alignment with only a light-weight design and a small amount of training data. Although it demonstrated competitive performance on some vision-language tasks such as VQA, its generalization capability and performance could be further improved. So future work should tackle the question "how to generalize the model to more tasks while achieving a good performance on all tasks".

5 References

[1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," *Arxiv*, 2022.

[2] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, Ajmal Mian, "A Comprehensive Overview of Large Language Models," *Arxiv*, 2023.

[3] J. L. S. A. M. M. C. L. Z. D. B. D. P. Aishwarya Agrawal, "VQA: Visual Question Answering," 2015.

[4] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, Ilya Sutskever, "Learning Transferable Visual Models From Natural Language Supervision," *arXiv*, 2021.

[5] kingyiusuen, "Search images with a text or image query, using Open AI's pretrained CLIP model.," Github Repository, 2023. [Online]. Available: https://github.com/kingyiusuen/clip-image-search.

[6] rom1504, "Easily compute clip embeddings and build a clip retrieval system with them," Github Repository, 2023. [Online]. Available: https://github.com/rom1504/clip-retrieval.

[7] Junnan Li, Dongxu Li, Silvio Savarese, Steven Hoi, "BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models," *arXiv:2301.12597*, 2023.

[8] Teng Wang, Jinrui Zhang, Junjie Fei, Hao Zheng, Yunlong Tang, Zhe Li, Mingqi Gao, Shanshan Zhao, "Caption Anything: Interactive Image Description with Diverse Multimodal Controls," *arXiv:2305.02677*, 2023.

[9] Renrui Zhang, Jiaming Han, Chris Liu, Peng Gao, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Yu Qiao, "LLaMA-Adapter: Efficient Fine-tuning of Language Models with Zero-init Attention," *arXiv:2303.16199*, 2023.

[10] Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, Ashwin Kalyan , "Learn to Explain: Multimodal Reasoning via Thought Chains for Science Question Answering," *Arxiv*, 2022.

[11] Deyao Zhu* , Jun Chen*, Xiaoqian Shen, Xiang Li, Mohamed Elhoseiny, “MiniGPT-4: Enhancing Vision-language Understanding with Advanced Large Language Models,” *arXiv:2304.10592,* 2023.

[12] Chiang, Wei-Lin and Li, Zhuohan and Lin, Zi and Sheng, Ying and Wu, Zhanghao and Zhang, Hao and Zheng, Lianmin and Zhuang, Siyuan and Zhuang, Yonghao and Gonzalez, Joseph E. and Stoica, Ion and Xing, Eric P., “Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90\%* ChatGPT Quality,” March 2023. [Online]. Available: https://lmsys.org/blog/2023-03-30-vicuna/.

[13] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, “Llama 2: Open Foundation and Fine-Tuned Chat Models,” *Arxiv,* 2023.

[14] GenAI, Meta, Llama 2: Open Foundation and Fine-Tuned Chat Models, 2023.

[15] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, C. Lawrence Zitnick, “Microsoft COCO Captions: Data Collection and Evaluation Server,” *arXiv:1504.00325,* 2015.

[16] N. D. S. G. R. S. Piyush Sharma, Conceptual Captions: A Cleaned, Hypernymed, Image Alt-text Dataset, 2018.

[17] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein & Li Fei-Fei , “Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations,” *International Journal of Computer Vision* , 2017.

[18] R. Beaumont, “img2dataset: Easily turn large sets of image urls to an image dataset,” *GitHub repository,* 2021.

[19] Lavis, “https://github.com/salesforce/LAVIS,” [Online].

[20] Meta, *https://huggingface.co/meta-llama/Llama-2-7b-chat-hf.*

[21] I. i. HKUST, *HPC3 CLUSTER,* Hong Kong, 2023.

[22] HuggingFace, *https://huggingface.co/docs/transformers/v4.17.0/en/parallelism.*

[23] HuggingFace, *https://huggingface.co/docs/datasets/en/index.*

[24] PyTorch, *CUDA Automatic Mixed Precision Examples,* PyTorch, 2024.

[25] Akhilesh Gotmare, Nitish Shirish Keskar, Caiming Xiong, Richard Socher, “A Closer Look at Deep Learning Heuristics: Learning rate restarts, Warmup and Distillation,” *Arxiv,* 2018.

[26] Ilya Loshchilov, Frank Hutter, “SGDR: Stochastic Gradient Descent with Warm Restarts,” *arXit,* 2016.

[27] Ilya Loshchilov, Frank Hutter, “Decoupled Weight Decay Regularization,” *Arxiv,* 2017.

[28] Lhoest, Quentin and Villanova del Moral, Albert and Jernite, Yacine and Thakur, Abhishek and von Platen, Patrick and Patil, Suraj and Chaumond, Julien and Drame, Mariama and Plu, Julien, "Datasets: A Community Library for Natural Language Processing," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2021.

[29] Abid, Abubakar and Abdalla, Ali and Abid, Ali and Khan, Dawood and Alfozan, Abdulrahman and Zou, James, "Gradio: Hassle-Free Sharing and Testing of ML Models in the Wild," *arXiv*, 2019.

[30] Kenneth Marino, Mohammad Rastegari, Ali Farhadi, Roozbeh Mottaghi, "OK-VQA: A Visual Question Answering Benchmark Requiring External Knowledge," *Arxiv*, 2019.

[31] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, "Flamingo: a Visual Language Model for Few-Shot Learning," *Arxiv*, 2022.

[32] Yushi Hu, Otilia Stretcu, Chun-Ta Lu, Krishnamurthy Viswanathan, Kenji Hata, Enming Luo, Ranjay Krishna, Ariel Fuxman, "Visual Program Distillation: Distilling Tools and Programmatic Reasoning into Vision-Language Models," *arXiv*, 2024.

[33] A. Agrawal, *Python API and Evaluation Code for v2.0 and v1.0 releases of the VQA dataset*, 2023.

[34] G. S. Ruotian Luo, "Comprehension-guided referring expression," *arXiv*, 2017.

[35] L. Yu, "https://github.com/lichengunc/refer," [Online].

[36] Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, Rui Zhao, "Shikra: Unleashing Multimodal LLM's Referential Dialogue Magic," *arXiv*, 2023.

[37] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, Steven Hoi, "InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning," *arXiv*, 2023.

6 Appendix A - Project Planning

6.1 Distribution of Work

Task	Leung King Suen	Chen Jiawei	Wang Zekai
Research			
Do the literature survey	○	○	●
Analyse VL Foundation Models	●	○	○
Design			
Design the overall inference pipeline		●	○
Design the sub-modules of model (Image Encoder, Adaptor, LLM and Prompt Wrapper)	●	○	○
Design the Training Dataset	○	○	●
Design the Adaptor Training Algorithm	●	○	○
Design the Fine-tuning Strategy on Adaptor	○	○	●
Implementation			
Implement the Training Dataset Preparation and Pre-processing	○	●	○
Implement the Model and Forward-Backward Passes	●	○	○
Implement the Training with Large Model and Dataset	○	○	●
Implement the Adaptor Fine-tuning with Fine-tuning Dataset Preparation and Pre-processing	●	○	○
Testing			
Unit Test on sub-modules of the Model	●	○	○

Model Integration Testing	○	●	○
Evaluation			
Quantitative evaluation on VQA Benchmarks	●	○	○
Qualitative evaluation with User Acceptance Tests	○	○	●
Other Works			
Write the proposal	●	○	○
Draft the monthly reports	○	●	○
Write the progress reports	○	○	●
Draft the final report	●	○	○
Prepare for the presentation	○	●	○
Design the project poster	○	○	●

● Leader ○ Assistant

6.2 GANTT Chart

Task	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr
Do the literature survey										
Analyse VL Foundation Models										
Design the overall inference pipeline										
Design the sub-modules of model (Image Encoder, Adaptor, LLM and Prompt Wrapper)										
Design the Training Dataset										

Design the Adaptor Training Algorithm										
Design the Fine-tuning Strategy on Adaptor										
Implement the Training Dataset Preparation and Pre-processing										
Implement the Model and Forward-Backward Passes										
Unit Test on sub-modules of the Model										
Implement the Training with Large Model and Dataset										
Model Integration Testing										
Implement the Adaptor Fine-tuning with Fine-tuning Dataset Preparation and Pre-processing										
Quantitative evaluation on VQA Benchmarks and Qualitative evaluation with User Acceptance Tests										
Write the proposal										
Draft the monthly reports										
Write the progress reports										
Draft the final report										
Prepare for the presentation										

Design the project poster										

7 Appendix B - Required Hardware & Software

7.1 Hardware

Development PC	PC with dedicated GPUs running Windows (WSL) or Linux
Computing resources	HPC3 in ITSC including: a. 100 GB disk space per login node/person b. 1 TB Scratch space for storing the dataset. c. Maximum of 2 node allocations and 2 job scheduling d. GPU and CPU resources details see HKUST HPC3 website.

7.2 Software

SSH	For remote development on HPC3.
Python3.9	Programming languages for model.
Anaconda (PyTorch, Numpy…)	For installing needed development packages.
Gradio	For building project demo.

8 Appendix C - Meeting Minutes

Minutes of the 1st Project Meeting

Date: 10 May 2023

Time: 3:00 pm

Place: Room 3509

Present: Prof. Xu, Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: Leung King Suen

1. Approval of minutes

   This was the first formal group meeting, so there were no minutes to approve.

2. Report on progress

   2.1 All team members were informed about the topic of the Final Year Project.

3. Discussion items

   3.1 Prof. Xu introduced OpenAI's CLIP high-level ideas to us.

   3.2 We would need to familiarize ourselves with CLIP.

   3.3 Prof. Xu suggested we build an image search system using CLIP.

   3.4 He gave us reading materials (papers on CLIP) to understand the core idea of CLIP:

Contrastive pre-learning.

3.5 He recommended that we read that paper first and get back to him for further discussion.

3.6 Because the meeting was in the final period. As all the group members were busy preparing for final exams, we could not start reading the paper until June 2023.

4. Goals for the coming month

Read CLIP paper and play with it.

5. Meeting adjournment and next meeting

The meeting was adjourned at 4:00 pm.

Minutes of the 2nd Project Meeting

Date: 22 July 2023

Time: 2:00 pm

Place: Pacific Coffee, Hang Hau

Present: Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: Chen Jiawei

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

2.1 All members have begun to read the materials suggested by Prof. Xu and study on

CLIP.

2.2 All members have played with CLIP using the Jupyter Notebook provided by OpenAI.

3. Discussion items

3.1 Discussed the method used in learning CLIP like contrastive learning and pre-training.

3.2 Suggested we can make image search system directly using CLIP, searched for similar projects done on image search using CLIP, we found at least 4-5 publicly available GitHub projects.

3.3 Leung King Suen, Wang Zekai said there are multiple examples of image search system done already using CLIP, and they were not sure what more can be done with it, they were overwhelmed by which type of project of this FYP is going to be. All group members agreed that this question needs to be presented to Prof. Xu during the next meeting.

3.4 We did another round of brainstorming and searching for more papers to read, such that we can extend our FYP.

3.5 All team members will be having internships this summer, so we will not be able to do much or meet face to face, but we can still do some work and meet on Zoom.

3.6 Chen Jiawei summarized the findings and presented them as Discord message to Prof.

Xu.

4. Goals for the coming month

4.1 All members will read the chosen papers and understand them.

4.2 Meet with professor.

5. Meeting adjournment and next meeting

The meeting was adjourned at 4:00 pm.

The next meeting will be at 9:00 pm on July 26th via Discord with Prof. Xu

Minutes of the 3rd Project Meeting

Date: 26 July 2023

Time: 9:00 pm

Place: Discord (Online)

Present: Prof. Xu, Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: Leung King Suen

1. Approval of minutes

   The minutes of the last meeting were approved without amendment.

2. Report on progress

   All members have read about CLIP.

3. Discussion items

   3.1 Leung King Suen started by presenting his understanding on behalf of the whole group on CLIP and questions intended to ask Prof. Xu, such as what kind of projects this FYP is going to be.

   3.2 Prof. Xu said our VL-alignment will be more fine-grained than CLIP.

   3.3 Prof. Xu gave us ideas on connecting/ adapting LLMs to Vision Models, the Segmenting anything model, instruction tuning, VL-alignments.

   3.4 Prof. Xu said we shall come up with our objectives and plans for the FYP.

4. Goals for the coming month

   4.1 All members will research online about the topics mentioned by the Professor.

   4.2 Draft a proposal with several objectives.

5. Meeting adjournment and next meeting

5.1 The meeting was adjourned at 10:00 pm.

5.2 The detailed time and date of the next meeting is not determined as it depends on our progress, but it was determined that after we have made our simple proposal with objectives, we will arrange the next meeting with Prof. Xu.

Minutes of the 4th Project Meeting

Date: 12 August 2023

Time: 11:00 am

Place: Discord (Online)

Present: Prof. Xu, Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: Leung King Suen

1. Approval of minutes

   The minutes of the last meeting were approved without amendment.

2. Report on progress

   A simple draft proposal was done.

3. Discussion items

   3.1 Chen Jiawei presented the proposal with a few objectives to Prof. Xu.

   3.2 Prof. Xu said the project is divided into 4 big objectives.

3.3 These 4 objectives are image retrieval with text, fine-tune VL-alignment, fine-grained description on the retrieved image, and a visual interface for user.

4. Goals for the coming month

4.1 Create a new iteration of a draft proposal based on Prof. Xu's suggestion.

5. Meeting adjournment and next meeting

5.1 The meeting was adjourned at 12:00 am.

5.2 The detailed time and date of the next meeting is not determined.

Minutes of the 5th Project Meeting

Date: 26 August 2023

Time: 11:00 am

Place: Discord (Online)

Present: Prof. Xu, Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: Wang Zekai

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

Done with a proposal that lists our objectives into 4 stages with simple descriptions.

3. Discussion items

3.1 Chen Jiawei presented it to Prof. Xu.

3.2 Prof. Xu said our objectives are okay and would like us to add more details such as methodology (technical solutions) to it.

3.3 Chen Jiawei asked about other parts in the proposal, such as the implementations and testing part.

4. Goals for the coming month

4.1 Continue working on the current objectives, state how we would like to achieve the objectives by adding more details.

4.2 Leung King Suen would apply for a HPC3 account for future computing resource requirement.

5. Meeting adjournment and next meeting

5.1 The meeting was adjourned at 11:30 am.

5.2 The date of the next meeting will be at the start of the Fall semester.

Minutes of the 6th Project Meeting

Date: 26 August 2023

Time: 1:40 pm

Place: Room 3509

Present: Prof. Xu, Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: Leung King Suen

1. Approval of minutes

   The minutes of the last meeting were approved without amendment.

2. Report on progress

   Presented proposals with more details on the objectives part.

3. Discussion items

   3.1 We asked about whether our proposal should follow the structure of the proposal template.

   3.2 Prof. Xu said our literature review part needs more work; the format should be listing several papers within each direction.

   3.3 Prof. Xu suggested our proposal to follow the methodology part in the proposal template (designs, implementation, testing and evaluation)

4. Goals for the coming month

   4.1 Finish a complete version of the proposal and submit it to Prof. Xu by Sunday.

5. Meeting adjournment and next meeting

5.1 The meeting was adjourned at 2:10 pm.

5.2 The date of the next meeting will be after Prof. Xu reads our proposal.

Minutes of the 7th Project Meeting

Date: 29 October 2023

Time: 2:00 pm

Place: Discord (Online)

Present: Prof. Xu, Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: Leung King Suen

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress.

Presented the works we have done in October.

3. Discussion items

a. Work Completed

1) Development environment setup on HPC3

2) First Dataset collection and evaluation – We will use the WiT Dataset.

3) We have tried to create some demos on the BLIP-2 Model with the WiT Dataset. As the vision-processors are frozen, they will be used to generate image features in batch.

b. Difficulties Encountered

1) Env Setup: None of us used GPU cluster in HPC3 before, we need to study and get familiar with SLURM scripts to allocate GPU to our application (like Jupyter Notebook).

2) From the demos, we are confused on the interpretation of ITC/ITM scores generated by BLIP-2, we expect the two metrics to be correlated to each other all the time, but they showed discrepancies in some cases. This could be a problem for the human evaluation part since it showed deviation from our expectation.

c. Overall Progress

We think currently the progress is a little slow. We expect ourselves to finish the baseline evaluation of VL-alignment before October, but this is lacking behind due to other works. But as we all are clear and distributed the work in the future plan below, we should be able to work faster in the coming months.

4. Goals for the coming month

a. Implement/Design an evaluation code (using ITM/ITC scores) to form a baseline before fine-tuning the text-processors only in the blip-2 model for WiT-Dataset.

b. Implement/Design batch dataset processing code for fetching image, calculating image embeddings and store to local before insertion to the database.

c. Investigate how to build up our database, we will consider different approaches.

5. Meeting adjournment and next meeting

a. The meeting was adjourned at 3:00 pm.

b. The date of the next meeting will be after we made progress on the first stage.

Minutes of the 8th Project Meeting

Date: 28 November 2023

Time: 1:40 pm

Place: Room 3509

Present: Prof. Xu, Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: Leung King Suen

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

Presented the works we have done in October.

3. Discussion items

a. Model (pre-trained) evaluation on the text-based image retrieval task.

b. Choose/Design an appropriate evaluation metric for the text-based image retrieval task and implement the evaluation code for two different datasets.

c. Hard to evaluate on a large set (>5K) with current code. The current code is not efficient enough with long computing time.

4. Goals for the coming month

a. Implement/Design the fine-tuning pipeline on the text-processors only in the blip-2 model for WiT-Dataset and evaluate on the fine-tuned model.

b. Read about aligning the two feature space for how to train an ‘alignment’ and evaluate it for multiple tasks such as Object Grounding, Image-to-Text Retrieval, then compare the evaluation to the other existing work.

c. Implement the alignment, our plan is a projection layer from the image or multi-modal (if ground-truth text provided) feature space of blip2 to the feature space of proposed LLM. We also need to determine what LLM we will be projecting into in the coming month.

d. Design the evaluation strategy according to (2) and implement the evaluation. The evaluation can be done on the MSCOCO, or the dataset used in the evaluation of the existing works, then finally on the WiT dataset.

5. Meeting adjournment and next meeting

a. The meeting was adjourned at 2:30 pm.

b. The date of the next meeting will be after we made progress on stage two.

Minutes of the 9th Project Meeting

Date: 11 January 2024

Time: 11:10 pm

Place: Discord (Online)

Present: Prof. Xu, Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: Leung King Suen

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

Presented proposals with more details on the objectives part.

3. Discussion items

a. Studied MiniGPT-4 and MiniGPT-V2 training & evaluation methods (paper & code).

b. Determined the strategy for the alignment between the visual feature space of Q-former and the language feature space of LLM (we use LLaMA-2-7b) by training a linear projection layer.

c. Determined the metric and benchmarks for evaluate the effectiveness of the trained projection layer.

d. Model implementation, training code and evaluation code in-progress.

4. Goals for the coming month

a. Implement the training and evaluation code for projection layer.

b. Start pre-training projection layer on hpc3, evaluate its baseline performance with general VQA evaluation metrics.

c. After we have the baseline, we will investigate which area that could be improved, then the improvement will be carried out by fine-tune on certain dataset.

d. Prepare for Progress Report

5. Meeting adjournment and next meeting

a. The meeting was adjourned at 12:10 pm.

b. The date of the next meeting will be around mid-February.

Minutes of the 10th Project Meeting

Date: 06 February 2024

Time: 10:20 pm

Place: Discord (Online)

Present: Prof. Xu, Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: WANG ZEKAI

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

Presented the work we have done in January.

3. Discussion items

a. To construct our progress report, listen to the comment from professor.

b. Training code implementation of the projection layer (adapter).

1. Discovering difficulties in the training of the projection layer: the training process is too slow
2. After testing our adapter on a small dataset, we find its given LLM output is unrelated to the test images

4. Goals for the coming month

a. Carry on implementing the training and evaluation code for projection layer and solve the issue of slow training.

b. To analyse the issue of unrelated output from the LLM.

c. After solving the issues, we will investigate the baseline performance by carrying out evaluation.

5. Meeting adjournment and next meeting

a. The meeting was adjourned at 12:00 pm.

b. The date of the next meeting will be around the end of February.

Minutes of the 11th Project Meeting

Date: 25 February 2024

Time: 4:20 pm

Place: Discord (Online)

Present: Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: WANG ZEKAI

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

Presented the work we have done in February.

3. Discussion items

a. Determine the method of DDP in acceleration of adapter training. However, we are still suffering from slow training due to limited resources.

4. Goals for the coming month

a. Carry on implementing the training and evaluation code for projection layer and solve the issue of slow pre-training.

b. To analyse the issue of unrelated output from the LLM.

c. After solving the issues, we will investigate the baseline performance by carrying out evaluation.

5. Meeting adjournment and next meeting

a. The meeting was adjourned at 12:00 pm.

b. The date of the next meeting will be around the end of February.

Minutes of the 12th Project Meeting

Date: 09 March 2024

Time: 10:30 am

Place: Discord (Online)

Present: Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: CHEN JIAWEI

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

The pre-training has successfully started.

3. Discussion items

a. Study the finetuning datasets for more fine-grained tasks, such as VQA, Visual Relationship and Image Captioning.

b. Study the finetuning strategy for these fine-grained tasks.

4. Goals for the coming month

a. Implement the fine-tuning methods for the 3 fine-grained tasks

5. Meeting adjournment and next meeting

a. The meeting was adjourned at 12:00 am.

b. The date of the next meeting will be around the end of March.

Minutes of the 13th Project Meeting

Date: 24 March 2024

Time: 5:40 pm

Place: Discord (Online)

Present: Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: CHEN JIAWEI

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

The fine-tuning has successfully implemented.

3. Discussion items

a. Determine the final testing and evaluation metric regarding to the fine-grained tasks' output.

b. Discuss and prepare for demo and presentation.

4. Goals for the coming month

a. Prepare for submission of final report.

b. Finish testing and evaluation of the fine-grained tasks

5. Meeting adjournment and next meeting

a. The meeting was adjourned at 8:00 pm.

b. The date of the next meeting will be around the beginning of April.

Minutes of the 14th Project Meeting

Date: 1st April 2024

Time: 3:00 pm

Place: Discord (Online)

Present: Leung King Suen, Chen Jiawei, Wang Zekai, Ms. Noor Liza

Recorder: CHEN JIAWEI

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

N/A

3. Discussion items

a. Reviewed Progress Report with CT Ms. Noor Liza

b. Discussed about the suitability of charts usage in our progress report.

4. Goals for the coming week

a. Modify structure and content in the Final Report according to Ms. Noor Liza 's advice.

b. Finish training and start fine-tuning of our model.

c. Implement VQA benchmark evaluations.

5. Meeting adjournment

a. The meeting was adjourned at 5:00 pm.

Minutes of the 15th Project Meeting

Date: 8 April 2024

Time: 5:00 pm

Place: Discord (Online)

Present: Leung King Suen, Chen Jiawei, Wang Zekai

Recorder: WANG ZEKAI

1. Approval of minutes

The minutes of the last meeting were approved without amendment.

2. Report on progress

Chen Jiawei said the training is done with 3 epochs, he said he will continue fine-tuning, and we can start evaluating the performances on VQA benchmark now.

Report progress on re-writing some parts of the Final Report according to the communication tutor's suggestion.

3. Discussion items

a. Writing Implementation, Testing and Evaluation part of the Final Report

b. Design and distribute survey for qualitative evaluation.

4. Goals for the coming week

a. Complete final evaluation on the model
b. Final wrap ups on the Final Report
c. Receive survey results and evaluate it, incorporate it into final report.

5. Meeting adjournment and next meeting

a. The meeting was adjourned at 11:00 pm.
b. The date of the next meeting will be around the end of this next week.