# Generative AI & LLM

## Semester Project Report

## Enhancement of Compositional Prompts Using Fine-Tuned BLIP-2

| | |
|---|---|
| **Name:** | **Hassan Javaid**, **Sauda Maryam**, **Javeria Saeed** |
| **Roll No:** | **MSCS23001**, **MSDS22025**, **MSCS23010** |
| Instructor Name: | Dr. Mohsen Ali |
| TA Name: | Rohaan Manzoor |
| Date of Sub: | January 9, 2025 |

# 1 INTRODUCTION & SUMMARY

The main goal of this project was to enhance the compositional prompts generated by a compositional learning model. This learning model generated original prompts of the format "a photo of acrylic sandal", where "acrylic" is the attribute and "sandal" is the object. The enhanced prompt was generated by calculating the similarity score with pre-defined CLIP embeddings of different dataset attributes i.e. color and material. The highest similarity score was chosen, and the enhanced prompt was generated using a pre-defined format: "A photo of acrylic sandal with white color and rubber material" . The dataset used was UT-Zappos-50K [1] which is a shoe dataset and it is publicly available on our Kaggle website [2]. The compositional learning model used was Troika model which is found at the this repo [3]. BLIP-2 LLM [4] used for training and fine-tuning with enhanced prompts and for further inference.

This project can be broken down into three steps:

1. **Enhanced Prompt Data Generation:** Conversion of original prompts of Troika Model to enhanced prompt templates using similarity matching with CLIP embeddings. A total of 24k enhanced prompts was generated by processing the UT-Zappos Dataset.
2. **Fine-Tuning of BLIP-2 using PEFT and LORA:** BLIP-2 was pre-initialized with Low Rank Adaptation (LoRA) configuration and Parameter Efficient Fine-Tuning (PEFT) methodology for fine-tuning.
3. **Inference using Fine-Tuned LLM:** The fine-tuned BLIP model was used for inference of test dataset and evaluation scores were calculated using BLEU and ROUGE metrics etc.

## 1.1 GitHub Repository

The project github repository is available at the following link: https://github.com/hassanjavaid07/CS500-Generative-AI-LLM-Project.

This contains all of the project code, trained models and results.

# 2 PROJECT SPECIFICATIONS

The key specifications of various components used in this project are detailed in this section.

1. Dataset - UT-Zappos50k
2. Compositional Model - Troika Model
3. LLM with PEFT config - BLIP-2 LLM

## 2.1 DATASET SPECIFICATION

The dataset used for this project was **UT-Zappos50k** dataset. Following are its main specifications:

- **Total Number of images** 50,000.
- **Categories:** Includes multiple types like sandals, sneakers, boots, etc.
- **Attributes:** Each image is annotated with fine-grained attributes:
  - **Colors:** Red, blue, black, white, etc.

  - **Materials:** Leather, canvas, rubber, synthetic, etc.
  - **Closure:** Lace-up, slip-on, buckle, velcro.
  - **Heel Height:** Flat, low, medium, high.

- **Train–valid–test split** =80-10-10 .

- **Size of each image** = Variable.

- **Total number of base classes** = 4.

- **Images in each class** = Variable (approx).

## 2.2 COMPOSITIONAL MODEL SPECIFICATION

The compositional model used for this project was **Troika** model. Following are its main specifications:

- **Purpose:** Multi-Path Cross-Modal Traction for Compositional Zero-Shot Learning

- **Key Features:**

  - Establishes three identification branches to jointly model the state, object, and composition.
  - Aligns branch-specific prompt representations with decomposed visual features.
  - Incorporates a Cross-Modal Traction module to adjust prompt representations towards the current visual content.

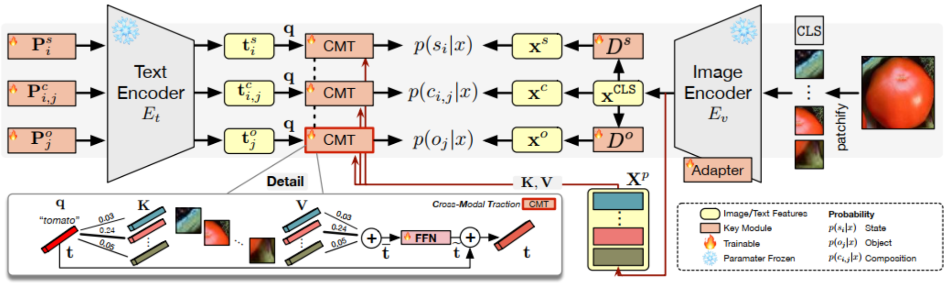


Figure 4. **Overview of the proposed *Troika*.**

Figure 1: Troika Architecture

## 2.3 LLM & PEFT SPECIFICATION

Following are the specifications of **BLIP-2** LLM used for this project

- **Model Name:** BLIP-2 (Bootstrapping Language-Image Pre-training 2)

- **Architecture:**

  - Multi-modal model combining frozen vision encoders (e.g., ViT, CLIP) with large language models (LLMs).
  - Querying Transformer for efficient vision-to-language alignment.

- **Training Details:**

  - Two-stage pre-training:
    - Vision-to-Language Pre-training: Aligns visual embeddings with text representations.
    - Language Model Tuning: Fine-tunes the model on downstream tasks like captioning and VQA.
  - Pre-trained on large-scale image-text datasets such as LAION and CC3M.

- **Input Specifications:**
  - Image resolution: Up to 224 × 224 pixels (adjustable for higher resolutions).
  - Text input: Tokenized text sequences for prompts or questions.

- **Output Specifications:**
  - Textual captions or answers with semantic relevance to visual input.
  - Zero-shot capability for novel image-text tasks.

- **Parameter Count:**
  - Vision Encoder: Pre-trained (e.g., ViT or CLIP models).
  - Querying Transformer: Lightweight, additional layers for alignment.
  - Language Model: Integrates large pre-trained LLMs like OPT or GPT-3.

- **Performance Metrics:**
  - Evaluated using BLEU, ROUGE scores for text generation tasks.

- **Fine-Tuning Capabilities:**
  - Supports LoRA and PEFT for parameter-efficient fine-tuning.
  - Compatible with diverse downstream datasets like Flickr30k and GQA.

- **Pre-Trained Models:** Supports multiple backbone architectures (e.g., ViT-B, CLIP-ViT-L).

Following are the specifications of Low-Rank Adaptation (LoRA) and PEFT configurations used that enabled the fine-tuning of Blip-2:

- **Using LoRA for implementing Parameter Efficient Fine-Tuning (PEFT):**
  - **Rank (r):** 16
  - **Scaling Factor:** 32
  - **Dropout:** 0.1
  - **Target Modules:** [q_proj, v_proj]

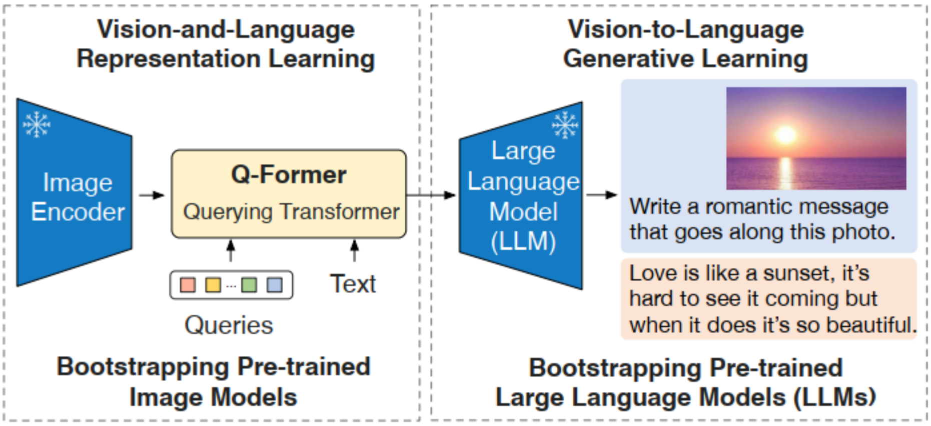

Vision-and-Language
Representation Learning
Image Encoder
Q-Former
Querying Transformer
Queries
Text
Bootstrapping Pre-trained
Image Models
Vision-to-Language
Generative Learning
Large Language Model (LLM)
Write a romantic message that goes along this photo.
Love is like a sunset, it's hard to see it coming but when it does it's so beautiful.
Bootstrapping Pre-trained
Large Language Models (LLMs)

Figure 2: BLIP-2 Framework Overview

# 3 ENHANCED PROMPT DATA GENERATION

For data generation of enhanced prompts using CLIP embedding we followed the steps as detailed below. The architecture for this step is also shown in figure 3.

## 3.1 Steps for creation of Enhanced Prompt Dataset

1. Calculate CLIP embedding vectors for our dataset attributes.
   - **Colors:** Red, blue, black, white.
   - **Materials:** Leather, canvas, rubber, synthetic.
2. Store these pre-defined CLIP embeddings for calculation of similarity index score later on.
3. Generation of original prompt for our dataset images using our compositional learning model. This prompt was of the format:
   - **Original Prompt:** "A photo of Canvas_Shoes.Boat.Shoes" or "A photo of sandal"
4. Encode each input image using CLIP and generate its embedding vector.
5. Calculate the cosine similarity score between image embedding and each of attribute embeddings calculated in the previous step. The highest match score will give us the best color and the best material.
6. Use the **best color** and **best material** calculated to generate our new enhanced prompt that will have been pre-defined for fine-tuning our BLIP-2 model.
   - **Enhanced Prompt:** "A photo of Canvas_Shoes.Boat.Shoes with **{best_color} color** and **{best_material} material**."
   - In case no possible match is found our **enhanced prompt** will become: "A photo of Canvas_Shoes.Boat.Shoes with **{undefined}** color and **{undefined}** material."
7. We repeated this process till we generated the enhanced prompt for all of our dataset images i.e. 24k images.

## 3.2 Data Generation Pipeline Architecture

The architecture for Data Generation Pipeline is given in the following figure 3:

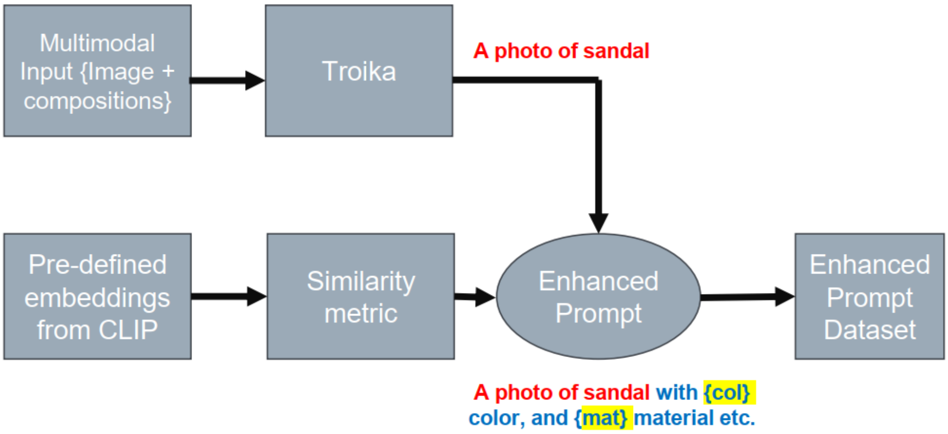

Multimodal Input {Image + compositions}
Troika
A photo of sandal
Pre-defined embeddings from CLIP
Similarity metric
Enhanced Prompt
Enhanced Prompt Dataset
A photo of sandal with {col} color, and {mat} material etc.

Figure 3: Architecture of our Enhanced Prompt Data Generation Pipeline

# 4 FINE-TUNING OF BLIP-2

The following steps outline the fine-tuning process that we used for our BLIP-2 LLM. The pipeline architecture diagram is also shown in figure 4.

## 4.1 Steps for fine-tuning of BLIP-2

1. Initialize our LoRA and PEFT parameters as follows:
    - **Rank (r):** 16
    - **Scaling Factor:** 32
    - **Dropout:** 0.1
    - **Target Modules:** [q_proj, v_proj]
2. Initialize our pre-trained BLIP-2 model and freeze its various transformer layers. The total parameter count of our parameters are as follows:
    - **Total Parameters:** 3,749,922,816
    - **Trainable Parameters:** 5,242,880
    - **Trainable Percentage:** 0.1398%
    - The relevant code-snippet is shown in the figure **??**
3. We provide our multi-modal LLM i.e. BLIP-2 with two inputs; a train image, the corresponding enhanced prompt for that image as calculated using steps mentioned in section 3.
4. Run the model for 15 epochs of training and calculate the train and validation losses.
5. Log and plot the results.

## 4.2 Fine-Tuning Pipeline Architecture

The architecture for Fine-tuning Pipeline is given in the following figure: 3:

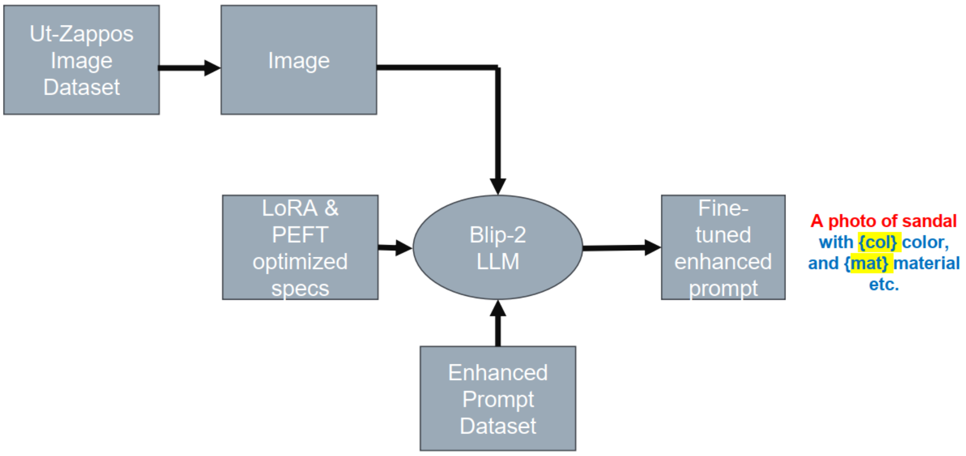
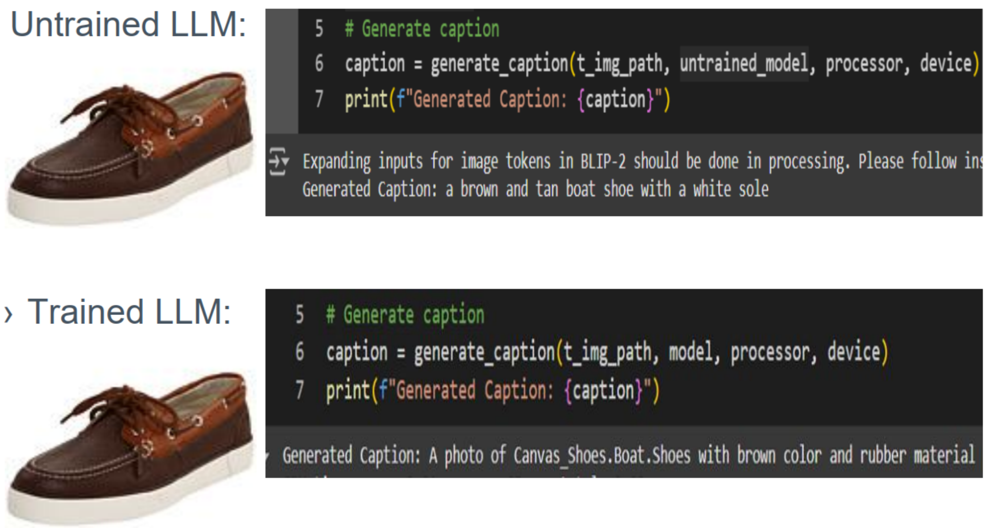

Ut-Zappos Image Dataset
Image
LoRA & PEFT optimized specs
Blip-2 LLM
Fine-tuned enhanced prompt
A photo of sandal with {col} color, and {mat} material etc.
Enhanced Prompt Dataset

Figure 4: Architecture of our BLIP-2 Fine-tuning Pipeline

## 4.3 Results

The fine-tuned LLM result comparison and plots of training and validation losses are given in the following figures:

### 4.3.1 Comparison of Fine-tuned LLM with untrained version



```
# Generate caption
caption = generate_caption(t_img_path, untrained_model, processor, device)
print(f"Generated Caption: {caption}")
```

```
Expanding inputs for image tokens in BLIP-2 should be done in processing. Please follow ins
Generated Caption: a brown and tan boat shoe with a white sole
```

```
# Generate caption
caption = generate_caption(t_img_path, model, processor, device)
print(f"Generated Caption: {caption}")
```

```
Generated Caption: A photo of Canvas_Shoes.Boat.Shoes with brown color and rubber material
```

Figure 5: Comparison of our Trained BLIP-2 LLM with Untrained version

### 4.3.2 Train and Validation Losses

The losses are given in table 1 and plot is shown in figure 6.

| Loss Type | Min Loss | Max Loss | Mean Loss |
|---|---|---|---|
| Train | 0.026 | 0.365 | 0.16 |
| Validation | 0.0795 | 0.2499 | 0.1001 |

Table 1: Training and Validation Loss Statistics

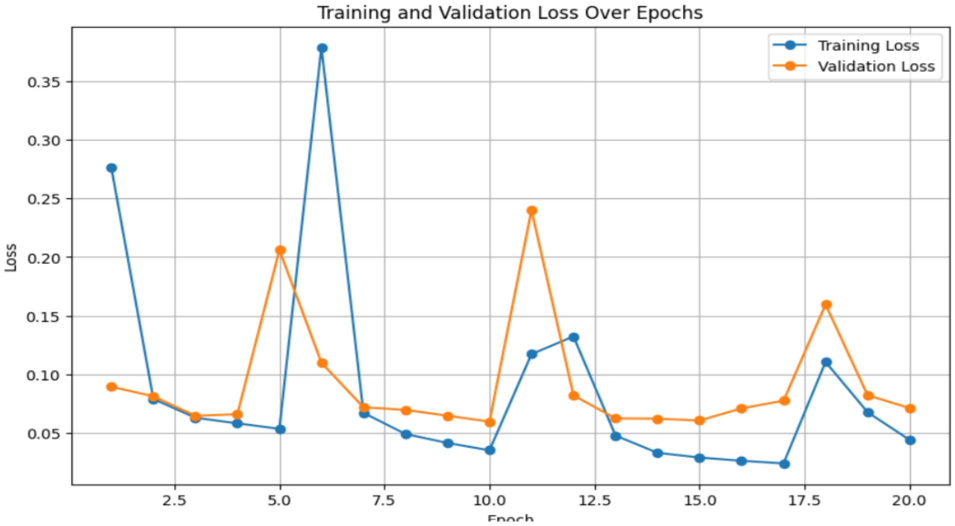


Figure 6: Train and Validation Losses

# 5 INFERENCE USING FINE-TUNED LLM

Our fine-tuned LLM was then evaluated on the test dataset. The testing metrics and strategy are detailed in this section:

## 5.1 Evaluation Metrics

We used the following evaluation metrics for this part. A brief summary and mathematical formula of each is as follows:

### 5.1.1 BLEU (Bilingual Evaluation Understudy) Score

The BLEU score is a precision-based metric used for evaluating machine-generated text by comparing n-grams (up to a certain size) between the candidate and reference translations. The formula for BLEU is:

$$\text{BLEU}(p_n) = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

Where:

- $p_n$ is the precision for n-grams of order $n$,
- BP is the brevity penalty,
- $w_n$ are weights (usually $w_n = \frac{1}{N}$ for equal weighting),
- $N$ is the maximum n-gram order (usually 4).

Brevity penalty BP is defined as:

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{1-\frac{r}{c}} & \text{if } c \leq r \end{cases}$$

Where:

- $c$ is the length of the candidate translation,
- $r$ is the length of the reference translation.

### 5.1.2 ROUGE-N (Recall-Oriented Understudy for Gisting Evaluation)

ROUGE-N is a recall-based metric, where $N$ refers to the n-gram precision and recall. For ROUGE-N, the formula for recall is:

$$\text{ROUGE-N Recall} = \frac{\sum_{n \in \text{n-grams}} \text{count}_{\text{match}}(n)}{\sum_{n \in \text{n-grams}} \text{count}_{\text{reference}}(n)}$$

Where:

- $\text{count}_{\text{match}}(n)$ is the number of n-grams matching between the candidate and reference,
- $\text{count}_{\text{reference}}(n)$ is the total number of n-grams in the reference.

### 5.1.3 ROUGE-L (Longest Common Subsequence)

ROUGE-L measures the longest common subsequence (LCS) between the candidate and reference text. The formula for ROUGE-L is:

$$\text{ROUGE-L} = \frac{LCS(\text{candidate}, \text{reference})}{\text{length of reference}}$$

Where:

- $LCS(\text{candidate}, \text{reference})$ is the length of the longest common subsequence.

### 5.1.4 Cosine Similarity

Cosine similarity measures the cosine of the angle between two vectors, commonly used for comparing text. The formula for cosine similarity is:

$$\text{cosine similarity} = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|}$$

Where:

- $\mathbf{A}$ and $\mathbf{B}$ are the vectors representing the two texts,
- $\mathbf{A} \cdot \mathbf{B}$ is the dot product of the vectors,
- $\|\mathbf{A}\|$ and $\|\mathbf{B}\|$ are the magnitudes (norms) of the vectors.

## 5.2 Evaluation Strategy

The description and architecture of our evaluation strategy is are given in this section.

### 5.2.1 Main Idea and overview

- Calculate the evaluation metrics as detailed in the above section and log them.
- Next we used the comparison of CLIP embedding scores obtained from Ground Truth Image and fine-tuned BLIP-2 LLM. If the ground truth result was higher then it was classified as a wrong prediction otherwise it was classified as a correct prediction.

### 5.2.2 Description & steps

- Input a sample of test images and calculate their scores using the evaluation metrics as outlined in section 5.1.
- Next we input our test image and ground truth caption to CLIP and obtain the embedding vector. Calculate its ground truth score.
- Input our test image and BLIP-2 generated caption to CLIP and generate and embedding vector. Calculate its BLIP score.
  - **Correct Prediction:** If BLIP Score is higher.
  - **Wrong Prediction:** If GT Score is higher.

### 5.2.3 Inference Architecture

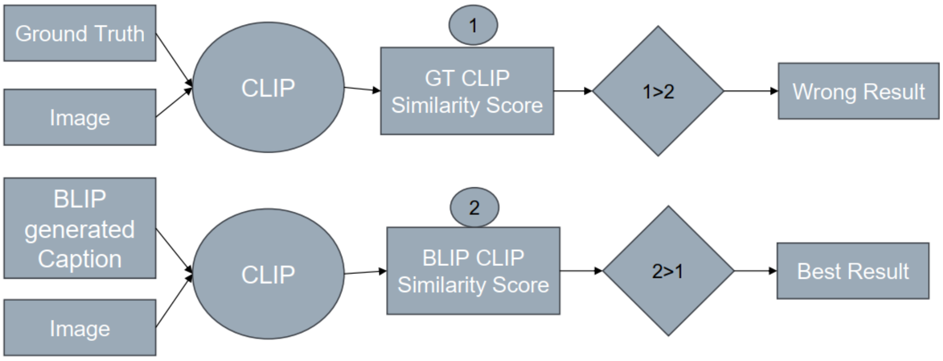


Figure 7: Inference Scheme for comparison score

## 5.3 Results

The tabulated results of our evaluation metrics along with correct and wrong predictions are detailed here:

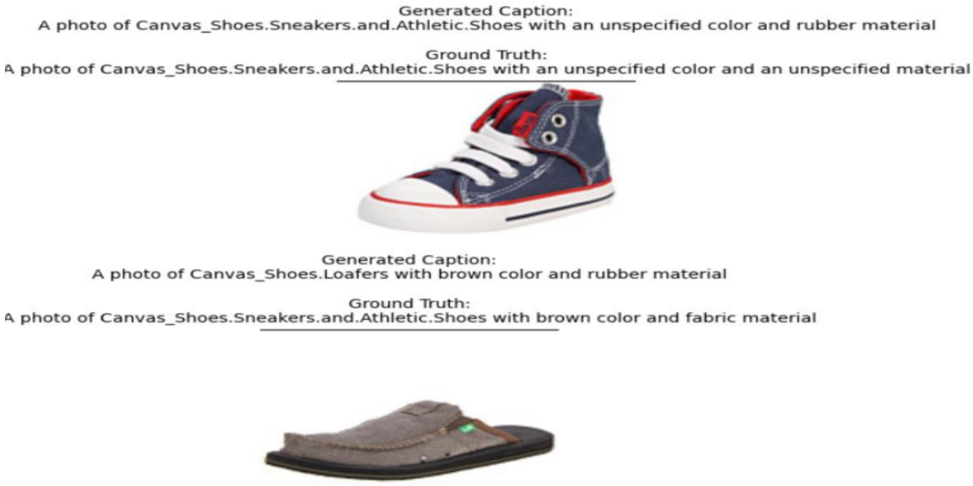
### 5.3.1 Evaluation metrics results

| Metrics | Best Case | Average | Worst Case |
|---|---|---|---|
| BLEU | 1.0 | 0.777 | 0.3554 |
| ROUGE-2 | 1.0 | 0.852 | 0.444 |
| ROUGE-L | 1.0 | 0.9204 | 0.6206 |

Table 2: Evaluation Metrics Results

### 5.3.2 Correct, Wrong and Comparison Prediction Results

**Correct Result**



Generated Caption:
A photo of Canvas_Shoes.Sneakers.and.Athletic.Shoes with an unspecified color and rubber material

Ground Truth:
A photo of Canvas_Shoes.Sneakers.and.Athletic.Shoes with an unspecified color and an unspecified material

Generated Caption:
A photo of Canvas_Shoes.Loafers with brown color and rubber material

Ground Truth:
A photo of Canvas_Shoes.Sneakers.and.Athletic.Shoes with brown color and fabric material

Figure 8: Correct Prediction Result

### Wrong Result

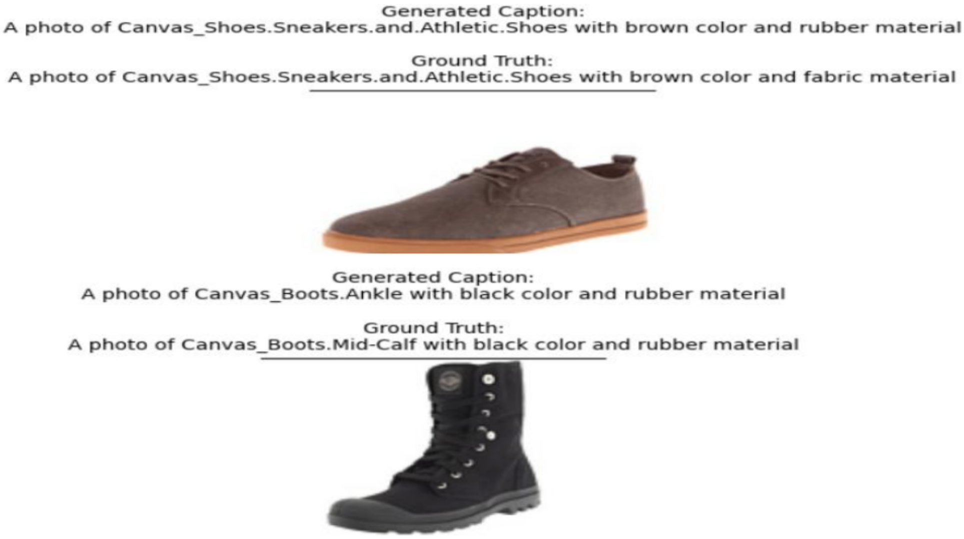

Generated Caption:
A photo of Canvas_Shoes.Sneakers.and.Athletic.Shoes with brown color and rubber material
Ground Truth:
A photo of Canvas_Shoes.Sneakers.and.Athletic.Shoes with brown color and fabric material
Generated Caption:
A photo of Canvas_Boots.Ankle with black color and rubber material
Ground Truth:
A photo of Canvas_Boots.Mid-Calf with black color and rubber material

Figure 9: Wrong Prediction Result

### Comparison Result

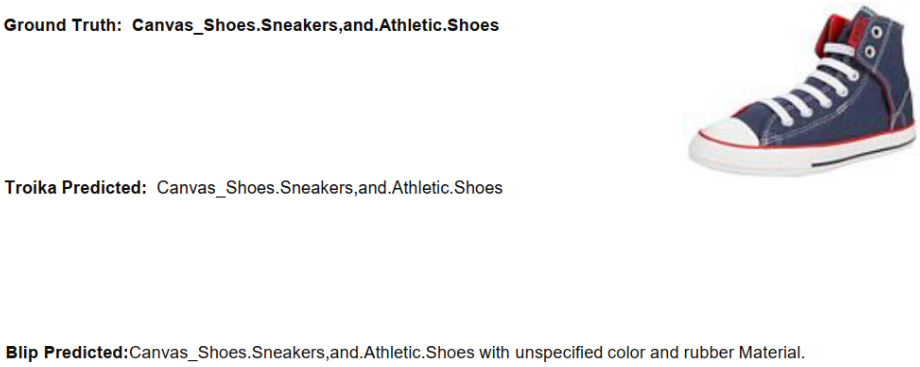

Ground Truth: Canvas_Shoes.Sneakers,and.Athletic.Shoes
Troika Predicted: Canvas_Shoes.Sneakers,and.Athletic.Shoes
Blip Predicted:Canvas_Shoes.Sneakers,and.Athletic.Shoes with unspecified color and rubber Material.

Figure 10: Comparison Prediction Result

# 6 CONCLUSION

- Incorporating CLIP embeddings for enhanced prompt generation has proven effective in providing more accurate and descriptive representations of images compared to the original approach.
- The fine-tuning of the BLIP-2 LLM was successfully carried out using the LoRA and PEFT methods, leading to significant improvements in prompt generation.
- The best training and validation losses were 0.026 and 0.0795 respectively which showed the success of our training methodology.
- The generated prompts were effectively tested against a range of text and natural language processing (NLP) evaluation metrics, including BLEU, ROUGE, and semantic similarity, demonstrating their robustness.
- The average results of 0.777 and 0.852 for BLEU and ROUGE-2 showed good performance on the testing dataset.
- Future work can focus on the following optimizations:
  - Expanding the dataset to include more object categories, such as the MIT-States or CIFAR datasets, to further improve the diversity and relevance of generated prompts.
  - Generalizing CLIP embeddings to encompass a wider range of objects, enhancing prompt quality, provided that additional fine-tuned data is available for better representation.

## References

[1] https://vision.cs.utexas.edu/projects/finegrained/utzap50k/.

[2] https://www.kaggle.com/datasets/steer01/ut-zap50k-images.

[3] Kyon Huang. bighuang624/Troika, January 2025. original-date: 2023-03-13T14:11:51Z.

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