VR Mini Project 2 : Multimodal Visual Question Answering and Finetuning with LoRA

IMT2022017 Prateek Rath
IMT2022076 Mohit Naik
IMT2022519 Vedant Mangrulkar

May 16, 2025

Contents

1 Introduction

This assignment involves creating a Visual Question Answering (VQA) dataset using the Amazon Berkeley Objects (ABO) dataset, evaluating baseline models such as BLIP and ViLT, fine-tuning using Low-Rank Adaptation (LoRA), and assessing performance using standard metrics. The Github Repository containing all code and data can be found here.

2 Dataset

We are using the Amazon Berkeley Objects Dataset, which contains around 150k Amazon product listings with multilingual metadata and around 400k catalog images.

- The listings metadata contains information like the product name, category, brand, color, keywords associated with the product etc.
- The images are variable-sized catalog images with a single item on a white background.

3 Preprocessing and Curation

Around 85% of the 398212 images are mobile phone cases and covers. The rest 15% come from a variety of categories such as clothing, shoes, home items, electronics, furniture, accessories, kitchen and groceries etc.

We first made a dataset of selected images along with cleaned up metadata, for which we filtered the data (images + listings) as follows:

- Ignore all products which have non-English non-ASCII metadata.
- Ignore all products which do not have metadata for product name, type, color and keywords.
- Consider all remaining products which are **not** mobile phone covers (around 9500), and then consider mobile phone covers to bring the total number of images to 10000. This was done to offset the huge imbalance in the number of mobile phone cover listings compared to other products.
- Resize all images to 256×256 for consistency, sort by filename and save the images to a separate folder, while saving the filename, name, product type, color and keywords in a csv. We then get a dataset of 10000 images, each with their product name, product type, color, and keywords.

To create the VQA dataset, we used the Google Gemini API to generate question-answer pairs for each image. Specifically, we used the `gemini-1.5-flash` model, to which we passed the product image, along with its metadata, and a prompt to generate 5 distinct questions per product, with some instructions as follows:

```
prompt = f"""
    You are given an image, some metadata about that image, and a set of instructions.
    Follow the instructions exactly.
    Image: {img}
    Metadata:
    Name: {name}
    Product Type: {product_type}
    Color: {color}
    Keywords: {keywords}
    Instructions:
    1. Generate 5 distinct questions.
    2. The questions can be answered by looking at the image or can be inferred by thinking.
    3. Difficult questions are preferred.
    4. Do not use quotation marks anywhere.
    5. The answer should exactly be 1 word.
    6. Provide the output strictly in the format given below.
    """
```

This prompt encourages the model to generate a mix of questions; ones which can be directly answered by looking at the image such as the product color or product name, and ones which require knowledge and thinking such as product functionality or fine details. We set the `temperature` parameter to 0.5 for a balance of conservative and creative questions.

After the entire dataset processing and VQA generation process, we get a set of 50000 question-answer pairs, 5 pairs each for 10000 selected images. Each question has a single-word answer.

4 Baseline VQA models

We worked with 2 off-the-shelf VQA models provided by HuggingFace :

- **Salesforce/blip-vqa-base**: A simple BLIP-base model with a ViT backbone and a text transformer. Pretrained on VQA tasks. We chose it because of ease of implementation and smaller footprint (∼400M params), along with good support for finetuning with `LoRA` and `peft`.
- **dandelin/vilt-b32-finetuned-vqa**: A smaller, lighter and faster model (∼100M params), composed of a ViT along with a BERT encoder. Pretrained on masked image modeling and image captioning tasks, and finetuned on VQA tasks. Unlike BLIP, this model is discriminative; it generates a distribution over the vocabulary instead of decoding text tokens. We chose it due to its small resource consumption and fast inference.

We ran inference for these 2 models on a fixed random sample of 10000 questions from the VQA dataset. The predicted answers for both models were recorded, and compared with the reference answers generated by the Gemini API, using the following metrics:

- **Exact Match Accuracy**: Proportion of examples where the model's prediction exactly equals (character-for-character) the ground-truth answer.
- **Substring Match Accuracy**: Proportion of examples where the prediction is a (contiguous) substring of the ground truth, or vice versa.
- **Bert F1**: Continuous similarity score between prediction and ground truth word embeddings. Indicates semantic closeness.
- **Levenshtein Similarity**: Word similarity based on edit distance i.e. minimum number of single-character edits (insertions, deletions, substitutions) needed to turn the prediction into the reference.
- **Meteor Score**: F1 score over exact matches, stem matches and synonym matches combined with a penalty for fragmented matches.

We did not use quite a few other metrics such as **BartScore**, **BLEU 1-4**, **Rouge Score**, since they are meant to compare sequences of words, and hence are of limited value when applied over single words, even though they can technically still be computed. Since the ground truth answers as well as most of the baseline predictions are single words, we chose the metrics which are easiest to understand and interpret, as well as give sufficient amount of information regarding the performance of the models, both structurally and semantically. A 'good' model, i.e. one which gives predictions close to the reference answers, will have high values for all the metrics listed above. The performance of both models before finetuning is given below:

Model	Exact Match	Substring Match	BERT F1	Levenshtein	METEOR
BLIP	0.3610	0.3787	0.8383	0.5117	0.2245
ViLT	0.2487	0.2551	0.8439	0.4058	0.1592

Table 1: Comparison of BLIP and ViLT on various evaluation metrics

We can see that both models perform similarly, with poor exact match accuracies but high BERT scores.

5 Finetuning

6 Results and Experiments

7 Observations and Challenges

8 Contributions

- IMT2022017 Prateek Rath:
- IMT2022076 Mohit Naik:
- IMT2022519 Vedant Mangrulkar: