

AIM 825 - Visual Recognition

Mini-Project-2 Report

Daksh Rajesh

(IMT2022019)

Jinesh Pagaria

(IMT2022044)

Aaditya Gole

(IMT2022087)

May 18, 2025

Contents

1	Dataset Curation	2
1.1	Exploratory Data Analysis (EDA)	2
1.2	Filtering Strategy	2
1.2.1	Split Generation	3
1.3	Question-Answer Generation	3
1.3.1	Final Aggregation and Cleaning	4
2	Model Choices & Baseline Evaluation	5
2.1	Model Choices and Justification	5
2.2	Evaluation metrics	5
2.3	Justification for Fine-Tuning Only BLIP-VQA-Base	6
3	Fine-Tuning Approaches & LoRA	6
4	Post-finetuning Evaluation	7
5	Importance of BERTScore in Visual Question Answering	7

Throughout this report,

- **Blue text** indicates paths, filenames, or scripts that are part of the codebase and should be referred to in the VRP2_Code and is surely present in the github repository provided.

1 Dataset Curation

1.1 Exploratory Data Analysis (EDA)

The dataset comprises multiple components:

`dataset/images/small` : Raw image folder.

`dataset/abo-listings/listings/metadata/*.json` : Metadata files containing attributes such as `main_image_id`, `other_image_ids`, and descriptive features.

`dataset/images/metadata/images.csv` : Image metadata with fields: `image_id`, `height`, `width`, and `path`.

Intuition suggests that we would need to perform a join between `*.json` listings metadata and `images.csv` on `main_image_id` to associate product metadata with corresponding image paths.

In [VRP2_Code/scripts/eda.ipynb](#), we analyze the distribution of key attributes:

- **Product Types:** 576 unique entries.
- **Color Codes:** 4306 unique values.
- **Style Attributes:** 12168 unique values.

We also examine the distribution of product listings across different language codes (see Table. 1)

Given the dominance of English (India and US), we focus our sampling on these languages to ensure the text embeddings learned by our vision-language models are linguistically meaningful.

1.2 Filtering Strategy

The filtering process is implemented in [VRP2_Code/scripts/filter.ipynb](#) and involves the following steps:

1. Retain metadata entries in English variants: `en_IN`, `en_US`, `en_CA`, `en_GB`, `en_AU`, and `en_SG`.

2. The complete metadata contains numerous attributes that are irrelevant in the context of Visual Question Answering (VQA). For instance, the attribute `item_weight` is not meaningful in this setting, as it is not possible to accurately assess the weight of an object solely from its image. Therefore, we curate a list of attributes that are deemed relevant to our VQA setup: `bullet_point`, `color`, `color_code`, `fabric.type`, `item_name`, `item_shape`, `material`, `pattern`, `product_description`, `product_type`, and `style`. Only these selected attributes are retained for the subsequent filtering step.
3. Save these filtered entries into `shortlisted_listings.jsonl`.
4. Duplicate entries by `image_path` may have made it to `shortlisted_listings.jsonl`, so we eliminate them keeping the one with the highest number of attributes to maximize contextual richness. The resulting file is `filtered_listings.json` (see Figure. 1).

```

1 {"image_path": "b4/b4f9d0cc.jpg", "color": ["Stone Brown"], "item_name": ["Stone & Beam Stone Brown Swatch, 25020039-01"]}
2 {"image_path": "2b/2b1c2516.jpg", "bullet_point": ["Embroidered flowers bloom against understated tan suede in this backless l
3 {"image_path": "9d/9dfccb37.jpg", "bullet_point": ["Snug fit for Mi Redmi Go, with perfect cut-outs for volume buttons, audio
4 {"image_path": "9f/9f903271.jpg", "bullet_point": ["Snug fit for Xiaomi Redmi Y2, with perfect cutouts for volume buttons, aud
5 {"image_path": "07/075e5d67.jpg", "bullet_point": ["Lead-free glass: made of crystal-clear, lead-free glass for stunning clari
6 {"image_path": "77/77412532.jpg", "bullet_point": ["Snug fit for Vivo Y8li, with perfect cut-outs for volume buttons, audio an
7 {"image_path": "73/736f202c.jpg", "bullet_point": ["Product Detail: Package include 6 pieces of chair.The Umi high elastic cou
8 {"image_path": "5f/5f0c12f2.jpg", "bullet_point": ["Snug fit for Karbonn Aura Power 4G Plus, with perfect cut-outs for volume
9 {"image_path": "8b/8b519b9b.jpg", "bullet_point": ["Snug fit for Samsung Galaxy A70s, with perfect cut-outs for volume buttons
10 {"image_path": "a1/a1774265.jpg", "bullet_point": ["3D Printed Hard Back Case Mobile Cover for Motorola Moto G4 Plus", "Easy t

```

Figure 1: Entries in `filtered_listings.json`

1.2.1 Split Generation

We partition `filtered_listings.json` into 9 balanced splits, S1 to S9, each containing 2500 entries. Each split satisfies the condition that all 576 unique `product_types` are represented, ensuring data diversity for downstream QA tasks.

- `Si/Si_images`: Image files
- `Si/Si_metadata` : Corresponding metadata entries with resolved image paths.

1.3 Question-Answer Generation

This component is implemented in [VRP2_Code/scripts/curate.ipynb](#). Using `gemini-1.5-flash`, we generate 3–6 question-answer (QnA) pairs per metadata entry.

The prompt used to generate question-answer pairs for an image is:

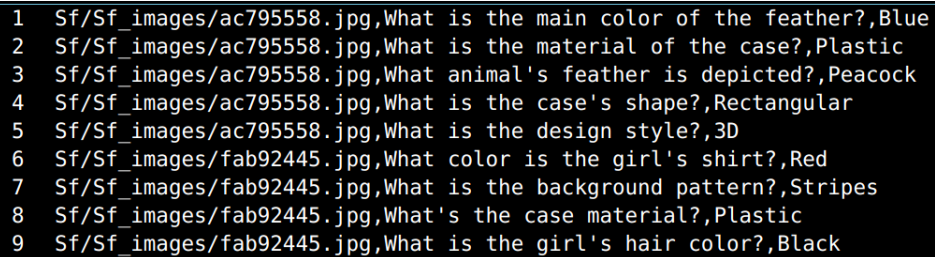
```
PROMPT_TEMPLATE = "Given the product image and the following context:
{context},
generate 5 short question and one-word answer pairs that test visual
understanding.
Each answer must be a single word (e.g., 'Red', 'Bag', 'Plastic').
Format as: Q1: <question> A1: <answer> ..."
```

Here, the context is supplied by the refined metadata corresponding to that image from `filtered_listings.json`. Each question targets visual attributes inferred from the image and its associated metadata.

Each generated entry is stored in an appropriate structured format for downstream use.

- `Si/Si_qa_data.csv`: Contains columns `image_path`, `question`, and `answer`.

1.3.1 Final Aggregation and Cleaning



```
1 Sf/Sf_images/ac795558.jpg,What is the main color of the feather?,Blue
2 Sf/Sf_images/ac795558.jpg,What is the material of the case?,Plastic
3 Sf/Sf_images/ac795558.jpg,What animal's feather is depicted?,Peacock
4 Sf/Sf_images/ac795558.jpg,What is the case's shape?,Rectangular
5 Sf/Sf_images/ac795558.jpg,What is the design style?,3D
6 Sf/Sf_images/fab92445.jpg,What color is the girl's shirt?,Red
7 Sf/Sf_images/fab92445.jpg,What is the background pattern?,Stripes
8 Sf/Sf_images/fab92445.jpg,What's the case material?,Plastic
9 Sf/Sf_images/fab92445.jpg,What is the girl's hair color?,Black
```

Figure 2: `Sf_qa_data.csv`

- All splits (S1 to S9) are merged into a final dataset Sf (see Fig. 2):
 - `Sf/Sf_images`
 - `Sf/Sf_qa_data.csv`
- The initial merged QnA file contains several mixed or compound answer strings, e.g., "Red/Maroon", "Red (Crimson Red)". These are cleaned and standardized to one-word answers in:
 - `Sf/Sf_qa_data_cleaned.csv` (approx. 97,000 rows)
- To reduce dataset size, a smaller subset with only 3 QnA pairs per image is created:
 - `Sf/Sf_qa_data_trimmed.csv` (approx. 59,000 rows)
- A train-test split (80-20) is made on the trimmed dataset:
 - `Sf_qa_data_trimmed_train_r.csv`
 - `Sf_qa_data_trimmed_test_r.csv`

2 Model Choices & Baseline Evaluation

2.1 Model Choices and Justification

We experimented with multiple off-the-shelf Visual Question Answering (VQA) models to establish baseline performance and identify suitable candidates for fine-tuning. The models evaluated were:

- **BLIP-VQA-Base** ([VRP2.Code/scripts/modle-evaluation.ipynb](#), only change is to load just the base modle and not the LoRA adapter)
- **ViLT** ([VRP2.Code/scripts/vilt-baseline.ipynb](#))
- **BakLLaVA** ([VRP2.Code/scripts/bakllava-baseline.ipynb](#))

2.2 Evaluation metrics

In a scenario with N ground truth strings and N corresponding predicted strings, each pair either matches (correct) or does not match (incorrect) i.e., binary correctness implied by exact matching of the strings. We evaluate performance using accuracy, precision, recall, and F1 score, defined below in the context of exact string matching, where true positives (TP) are correct matches, false negatives (FN) are mismatches, and false positives (FP) and true negatives (TN) are typically zero. Apart from exact mathcing of the strings we also use BERTScore.

- **Accuracy:** The fraction of correct predictions.

$$\text{Accuracy} = \frac{\text{TP}}{N}$$

Measures overall correctness but may be misleading if exact matches are rare, hence prompts us to explore metrics like BERTScore.

- **Precision:** The proportion of predicted matches that are correct.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = 1$$

Typically 1 in exact matching, as $\text{FP} = 0$.

- **Recall:** The proportion of true matches correctly predicted.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{N}$$

Equals accuracy in this setup, reflecting the ability to find true matches.

-
- **F1 Score:** The harmonic mean of precision and recall.

$$\text{F1 Score} = \frac{2 \cdot \text{Recall}}{1 + \text{Recall}}$$

Balances precision and recall, reducing to a function of recall since precision = 1.

The metrics for non-finetuned models are shown in Table 2

2.3 Justification for Fine-Tuning Only BLIP-VQA-Base

After evaluating these models, we chose to fine-tune only BLIP-VQA-Base based on the following rationale:

1. **Performance:** BLIP-VQA-Base outperformed ViLT and BakLLaVA on key metrics such as Accuracy, F1 Score, and BERTScore (see Table 2).
2. **Parameter Size and Efficiency:** While BakLLaVA is a strong model in terms of reasoning, its large parameter size ($\sim 7\text{B}$) made it impractical to fine-tune on constrained hardware. In contrast, BLIP-VQA-Base has $\sim 380\text{M}$ parameters, as compared to the extremes of ViLT and BakLaVA which have very few and very large number of parameters respectively.
3. **Compatibility:** BLIP is designed specifically for VQA and captioning tasks and integrates well with LoRA-based fine-tuning.

3 Fine-Tuning Approaches & LoRA

(Script: [VRP2_Code/scripts/blip-vqa-base-finetuning.ipynb](#))

- To adapt the BLIP-VQA-Base model to a curated Visual Question Answering (VQA) dataset, we employed Low-Rank Adaptation (LoRA) as a parameter-efficient fine-tuning strategy. The base model and processor were initialized from the pre-trained `Salesforce/blip-vqa-base`, which was also used for baseline evaluation.
- The dataset consisted of image-question-answer triples stored in a CSV file located at [dataset_curated/Sf/Sf_qa_data_trimmed_train_r.csv](#). A custom PyTorch `Dataset` class was implemented to load image files, tokenize the corresponding questions using the BLIP processor, and prepare answer labels via the tokenizer. Each image was converted to RGB format, and in cases where loading failed, a fallback black image of size 224×224 was used as a placeholder.

-
- LoRA was applied by targeting the `query` and `value` projection layers within the attention mechanism. Various configurations were experimented with (see Table 3). The PEFT (Parameter-Efficient Fine-Tuning) library was used to wrap the BLIP model with the specified LoRA configuration. Common configurations across all experiments:

- `lora_alpha` = 32
- `lora_dropout` = 0.1
- `per_device_train_batch_size` = 4
- `gradient_accumulation_steps` = 4
- `learning_rate` = 5e-5
- `weight_decay` = 0.01

- The model was trained using the HuggingFace `Trainer` API, with `TrainingArguments` specifying mixed-precision training (fp16), logging, and checkpoint saving strategies. The `Accelerate` library was used to prepare the model for distributed or accelerated training.
- Finally, the fine-tuned model was saved to disk for downstream inference or evaluation. This approach enabled effective adaptation of the VQA model with minimal additional parameter overhead, showcasing the practical advantages of LoRA for vision-language tasks.

4 Post-finetuning Evaluation

Evaluation of the trained/finetuned model is done on:

[dataset_curated/Sf/Sf_qa_data_trimmed_test_r.csv](#). We finetune the base model using several different ranks (r-values) in LoRA and evaluate the predictions on our test split (see Table 3). The table shows that at $r = 16$ the performance is better than others.

Then we $r = 16$, we experiment by training it for multiple epochs and evaluating their predictions over test split (see Table 4). When the metrics are plotted, the peak at epoch = 7 (see Figure. 3) hints that at epochs > 7 the model most likely overfits. Hence we choose $r = 16$, epoch = 7 as our optimal model.

5 Importance of BERTScore in Visual Question Answering

During evaluation, we encountered several cases where traditional metrics such as exact string match (e.g., token-level accuracy or F1 score) failed to reflect the semantic cor-

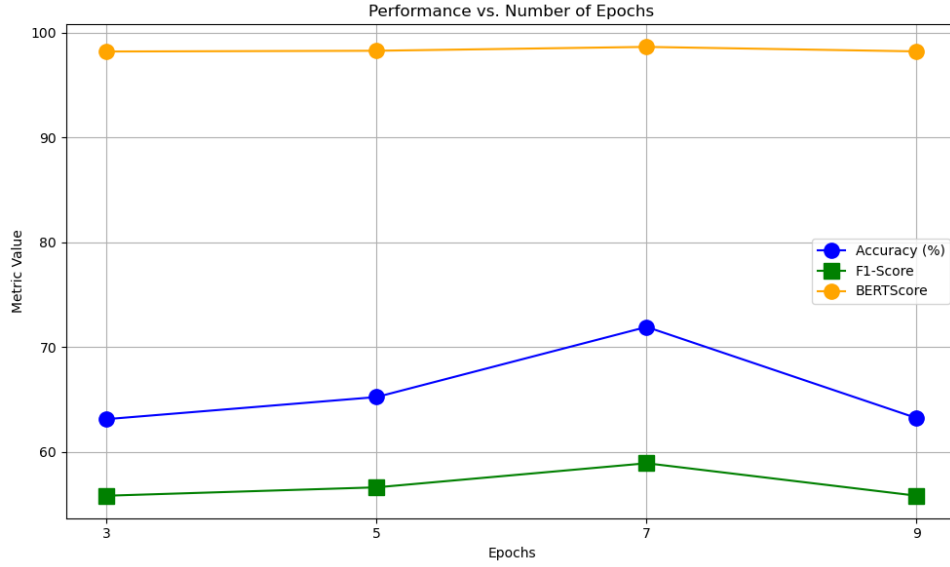


Figure 3: mean BERTScore and F1-score converted on scale 0-100

rectness of the model’s predictions. These observations highlight the necessity of using a semantic similarity metric like **BERTScore** in the context of Visual Question Answering (VQA), where lexical variations may still convey equivalent meanings. Below, we elaborate on some representative cases:

1. Singular vs. Plural Forms

Example: Truth: “*Elephant*” vs. prediction: “*Elephants*” for “*Which animal is printed on the shirt?*” (see Figure. 4)

Issue: Traditional string-based metrics treat this as incorrect due to a mismatch in number, despite both referring to the same animal class.

Significance: In VQA, especially in object identification tasks, plurality often does not impact the semantic correctness of the response. BERTScore correctly identifies their semantic equivalence.

2. Verb-Noun Form Variations

Example: Truth: “*Enjoy*” vs. predicted: “*Enjoying*” for “*What is the text written?*”(see Figure. 5)

Issue: The prediction may differ from the ground truth only in grammatical form (e.g., base vs. gerund).

Significance: Morphological variations do not necessarily indicate semantic errors. Moreover the model caters to VQA setup, its not a strict OCR-like model.

3. Numerical Form Mismatch

Example: “5” vs. “Five”

Issue: Despite referring to the same quantity, string comparison fails due to format differences.

Significance: This is common in quantity-based questions. Semantic-aware metrics like BERTScore account for such equivalence.

4. Object Saliency and Ambiguity

Example Scenario: An image contains both monkey, elephant and a hare. The question is: “*What animal is present?*”, prediction: monkey (see Figure. 5)

Observation: The model often chooses the animal that is more prominently featured (e.g., larger or centrally located).

Significance: In ambiguous cases with multiple valid answers, the model’s reliance on visual saliency results in plausible outputs. Exact match metrics fail to acknowledge such legitimate variability.



Figure 4: Random image taken from web for tetsiing

Language Code	Number of Products
en_IN	76,443
en_US	26,426
de_DE	15,100
es_US	12,016
zh_CN	11,708
pt_BR	9,953
ko_KR	8,778
zh_TW	8,766
en_GB	8,147
he_IL	8,016
hi_IN	7,463
⋮	⋮
mr_IN	2

Table 1: Distribution of products by language code in metadata.

Table 2: Baseline Evaluation Metrics for VQA Models

Model	Params	Accuracy	F1 Score	BERTScore (F1)
BLIP-VQA-Base	380M	38.2%	0.5527	0.972
ViLT	88M	28.57%	0.4467	0.9149
BakLLaVA	7000M	35.06%	0.05191	0.968

Table 3: Performance across different LoRA rank (r) values

LoRA Rank (r)	Accuracy (%)	F1-Score	BERTScore
8	62.71	0.556	0.9819
16	63.11	0.558	0.9820
24	63.03	0.557	0.9821
32	62.85	0.5569	0.9821

Table 4: Effect of Epochs on Model Performance

Epochs	Accuracy (%)	F1-Score	BERTScore
3	63.11	0.558	0.9820
5	65.22	0.566	0.9827
7	71.91	0.589	0.9864
9	63.21	0.558	0.9821



Figure 5: Random image taken from web for tetsiing