BLIP-2 Classifier: Using generative models on classification tasks

Razvan Florian Vasile [1]

[1]Computer Science, University of Bologna

Introduction

While visual multimodal models like BLIP-2 excel at generative question-answering tasks, implementations present in the popular Huggingface Transformers library [1] present limitations when applied to classification problems. This reduces the model's versatility and its applicability to a wide range of datasets. In this work, I propose a method to extend BLIP-2's capabilities for classification tasks and benchmark its performance on two datasets: EasyVQA and Daquar.

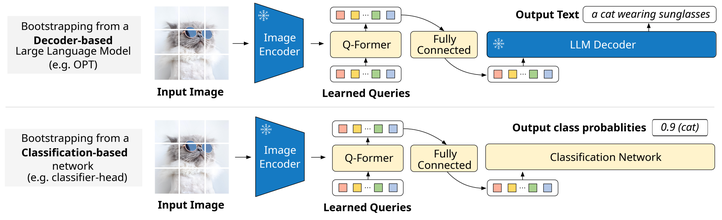


Figure 1. Proposed architecture. Diagram adapted from the original paper [3].

Problem Statement

- **Identify the Problem:** BLIP-2, while effective for generative objectives, it lacks support for classification tasks in the current HuggingFace implementations [1],
- **Put it into context:** Existing BLIP-2 classes are designed for tasks like image captioning and visual question answering which make them unsuitable to use for classification purposes.
- **Find the root cause:** The model's architecture is optimized for generative tasks, with components that aren't immediately compatible for classifying images.
- **Ideal outcome:** Develop a method that leverages existing components of the architecture in a way that allows easy extendability to classification problems.
- **Propose a solution:** The approach reutilizes the key components (Visual Encoder and Q-Former) in order to tailor the features for classification purposes.

Methods: Key Ideas

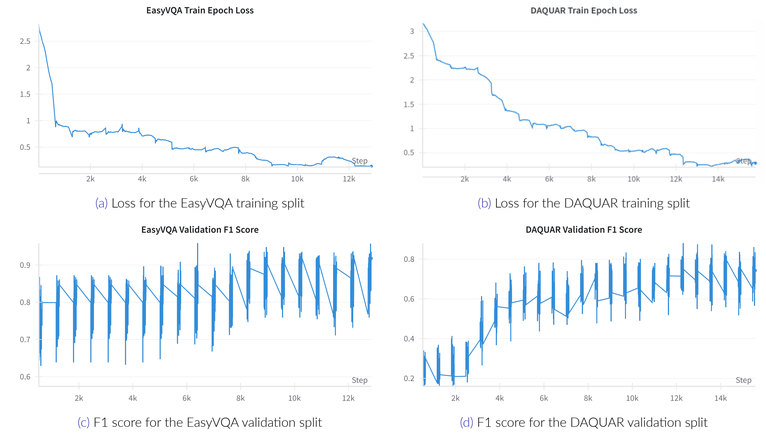
- **Inspiration:** Inspiration for the approach due to the Transformers Library source code [2] (especially Blip2TextModelWithProjection and Blip2VisionModelWithProjection).
- **Efficient Training:** Peft and LoRa significantly reduce the number of trainable parameters by using 'wrappers' around the weights enabling efficient task-specific training.
- **Parameters:** trainable params: 9,791,504 || all params: 1,181,823,762 || trainable%: 0.8285.
- **Memory Usage:** approx. 16 GB VRAM.
- **Training Time:** 5 hours and 6.5 hours for the EasyVQA and DAQUAR datasets respectively.

	EasyVQA		DAQUAR	
Metric	Classification	Generative	Classification	Generative
Validation Accuracy	91%	70%	78%	73%
Improvement	+21%	–	+5%	–
Number of classes	13 classes		18 classes	
Train/Valid samples	10,978/2744		13,524/3381	
Early Overfitting	No	Yes	No	Yes

Table 1. Synthesize of the results.

Results: Generated Responses with Citations

- **Regularization Techniques:** Utilizing low Lora ranks (8), L1 and L2 regularization (via AdamW optimizer), and CosineAnnealingLR to avoid overfitting and local minima.



(a) Loss for the EasyVQA training split (b) Loss for the DAQUAR training split

(c) F1 score for the EasyVQA validation split (d) F1 score for the DAQUAR validation split

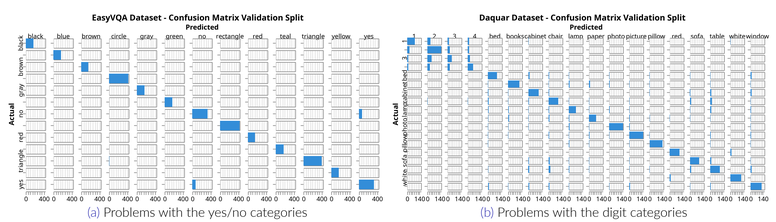
Figure 2. Training loss and validation F1 scores for both datasets.

Architectural Overview

- **Frozen Vision Encoder**: extracts visual information from images (reliant on pre-trained data).
- **Q-Former 32 Learnable Queries**: processes textual input combined with visual features.
- **Algorithm**: Visual features use the vision encoder and the Q-Former, while text embeddings use only the Q-Former. The resulting features are projected into a common dimensional space using simple linear layers which are then concatenated and classified (source).

Analysis: Confusion Matrices

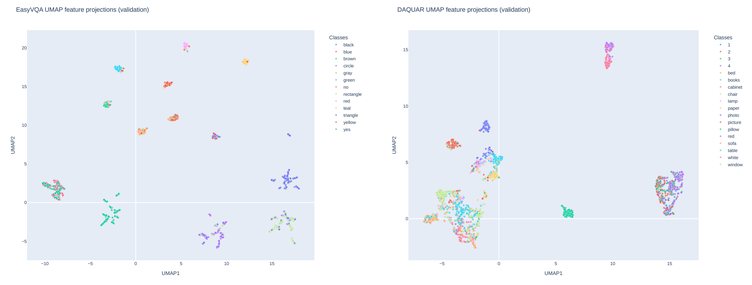
- **EasyVQA.** Consistently performs well on most classes, however the yes/no classes are the categories that may pose potential challenges.
- **DAQUAR.** Human accuracy baseline is 60% [4] and the model achieved 78% accuracy. Particular difficulties were encountered around the classification of the numerical classes.



(a) Problems with the yes/no categories (b) Problems with the digit categories

Figure 3. Confusion matrices for the validation splits: EasyVQA and DAQUAR.

Visual Interpretation using UMAP



(a) EasyVQA dataset (interaction version). (b) DAQUAR dataset (interactive version).

Figure 4. The distribution of the embeddings extracted from the validation datasets.

- **Knowledge representation**: unrelated classes are separated and related ones are grouped.
- **No perfect separation**: points may still be misclassified if they fall within the same cluster.
- **Intuitive animation**: progressive feature learning for EasyVQA and DAQUAR.

Conclusion

- **Extended capabilities.** The project demonstrates a way of extending the BLIP-2 architecture, effectively allowing users to perform classification tasks.
- **Broader adaptation.** The BLIP-2 model excels at reutilizing pre-trained data, effectively reducing computational requirements and leading to broader experimentation.
- **Reproducible Experiments.** Training and experimental data is available in the repository. Additional training metrics can be found on Wandb.

Future Work

- **Multimodal expansion.** Extend the architecture to handle additional modalities including graph and audio classification. This could provide insights into the model's feature extraction capabilities across different data types.
- **Architecture exploration.** Investigate the trade-offs between freezing and fine-tuning different components (vision model, LLM) to better understand their impact on performance.

References

[1] Huggingface.
BLIP-2 — huggingface.co.
`https://huggingface.co/docs/transformers/model_doc/blip-2`, 2024.
[Accessed 02-11-2024].

[2] Huggingface.
transformers/src/transformers/models/blip_2/modeling_blip_2.py at main · huggingface/transformers — github.com.
`https://github.com/huggingface/transformers/blob/main/src/transformers/models/blip_2/modeling_blip_2.py#L1890`, 2024.
[Accessed 02-11-2024].

[3] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi.
Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023.

[4] Mateusz Malinowski and Mario Fritz.
A multi-world approach to question answering about real-world scenes based on uncertain input, 2015.