Implementation Report: CIFAR-10 CLIP Comparison Application

Saleh Safarnejad

1 Introduction

This report details the implementation process of a Python application that evaluates the Open CLIP ViT-B-32 model on the CIFAR-10 dataset, comparing zero-shot classification (with multiple prompt templates and an ensemble) and linear probe classification. The application visualizes results using a bar plot and table, as specified in `cifar10_clip_comparison.py`. The implementation leverages PyTorch, Open CLIP, and scikit-learn, targeting both local and Google Colab environments with GPU support. Below, we discuss the implementation process, challenges encountered, and potential improvements, particularly focusing on the issue of persistent misclassifications despite prompt modifications.

2 Implementation Process

The application was developed as a modular Python script with the following components:

- **Data Loading**: The `load_cifar10_data` function loads the CIFAR-10 dataset using `torchvision`, applying resizing (224x224) and normalization to match CLIP's input requirements. Data is split into training and test sets with configurable batch sizes (default: 128).
- **Model Loading**: The `load_pretrained_clip` function initializes the Open CLIP ViT-B-32 model with the `laion2b_s34b_b79k` checkpoint, ensuring GPU compatibility via PyTorch's device management.
- **Zero-Shot Classification**: The `zero_shot_classification` function computes accuracy for user-specified prompt templates (e.g., "a photo of a {}"). It tokenizes prompts, encodes text and image features, and calculates logits using matrix multiplication, achieving typical accuracies of 88–90%.
- **Ensemble Classification**: The `ensemble_classification` function averages text embeddings from multiple prompts, improving zero-shot accuracy by 1–3%.
- **Linear Probe Classification**: The `linear_probe_classification` function extracts image features, trains a logistic regression model using scikit-learn, and achieves accuracies of approximately 92–95%.
- **Visualization**: The `visualize_and_compare` function generates a bar plot (`cifar10_comparison.png`) and a table (using `tabulate`) to compare accuracies.

The script accepts command-line arguments via `argparse` for prompts, batch size, data directory, encoder, and checkpoint, with defaults suitable for CIFAR-10 and Open CLIP. For Colab, an `args` dictionary replaces command-line parsing. The implementation was tested on a Colab GPU (T4), with feature extraction taking 5–10 minutes and linear probe training 1–2 minutes.

3 Challenges Faced

A significant challenge was the model's persistent misclassifications for certain CIFAR-10 images, even when modifying prompts to provide more context. For example:

- A deer was misclassified as a cat.
- An automobile was misclassified as a truck.
- An airplane was misclassified as a bird.

To address these, we experimented with prompts that added context, such as "a photo of a fast war {airplane}" instead of "a photo of a {}". However, these modifications often failed to correct the predictions. This issue likely stems from the model's pretraining on the LAION-2B dataset, which may not sufficiently distinguish fine-grained visual features (e.g., airplane vs. bird) or align specific prompts with CIFAR-10's low-resolution (32x32, resized to 224x224) images. The resizing process may also introduce artifacts, complicating feature extraction. Additionally, ensuring compatibility with Colab required careful dependency management and GPU configuration, as some packages (e.g., `open_clip_torch`) have specific version requirements.

4 Potential Improvements

To address persistent misclassifications, one promising improvement is to fine-tune the CLIP model on a subset of CIFAR-10 or a similar dataset with higher-resolution images. Fine-tuning could enhance the model's ability to distinguish fine-grained classes (e.g., airplane vs. bird) by adapting its image encoder to the dataset's visual characteristics. This could involve:

- **Contrastive Fine-Tuning**: Use a small labeled subset of CIFAR-10 to fine-tune the image and text encoders with a contrastive loss, aligning embeddings for specific classes.
- **Prompt Engineering with Domain Knowledge**: Incorporate domain-specific prompts (e.g., "a military jet {airplane}" for airplanes) informed by CIFAR-10's class characteristics, potentially combined with prompt ensembling.
- **Higher-Resolution Preprocessing**: Explore alternative preprocessing techniques (e.g., super-resolution) to mitigate artifacts from resizing CIFAR-10 images.

Additionally, integrating a validation step to check prompt effectiveness before evaluation could reduce errors from poorly formatted prompts. For Colab, automating dependency installation and GPU checks in a notebook could streamline setup. These improvements would enhance accuracy and usability, particularly for challenging cases like deer vs. cat or airplane vs. bird.

5 Conclusion

The CIFAR-10 CLIP comparison application successfully evaluates Open CLIP ViT-B-32, providing insights into zero-shot and linear probe performance. Despite challenges with persistent misclassifications, the modular design and robust visualization make it a valuable tool for exploring CLIP's capabilities. Fine-tuning and advanced prompt engineering offer promising avenues for improving accuracy, particularly for fine-grained classification tasks.