COSE474-2024F: Final Project Report “Improving CLIP Performance Using Data Augmentation and Fine-tuning Techniques”

Lee, Dongyoung

1. Introduction

1.1. Motivation

Image-text matching technology is a cornerstone for numerous real-world applications, such as image search engines and image captioning systems. These systems rely on the accurate alignment of visual and textual information to provide meaningful outputs. OpenAI’s CLIP model has emerged as a significant breakthrough in this domain, as it maps images and texts into a shared embedding space, enabling similarity-based matching. However, despite its strong performance, CLIP faces limitations, particularly in small datasets where generalization is critical. Enhancing the performance of CLIP in such scenarios remains an open challenge.

1.2. Problem Definition

This project aims to address the challenges associated with improving CLIP’s image-text matching performance, specifically focusing on small datasets. The primary objective is to explore the impact of data augmentation and fine-tuning strategies in enhancing CLIP’s generalization capabilities. By utilizing Flickr30K and COCO datasets, this study investigates whether targeted improvements can lead to more accurate image-text alignment.

1.3. Concise Description of Contribution:

- Implementation of data augmentation techniques (e.g., RandomResizedCrop, RandomHorizontalFlip) to increase input diversity and robustness.
- Fine-tuning of CLIP on the Flickr30K dataset to adapt the model to task-specific data distributions.
- Evaluation of the proposed methods on both Flickr30K and COCO datasets, achieving a validation accuracy of 95.32
- Visualization of augmented data and performance curves to analyze the effectiveness of the proposed methods.

2. Methods

2.1. Significance and Challenges

The significance of this project lies in addressing two key challenges in the image-text matching domain:

- **Challenge 1: Generalization on Small Datasets**: Fine-tuning CLIP on small datasets like Flickr30K often leads to overfitting. This project mitigates this issue through robust data augmentation techniques, such as random cropping, flipping, and normalization, which enhance model generalization.
- **Challenge 2: Computational Efficiency**: Fine-tuning large-scale models like CLIP is computationally expensive. To address this, we utilized GPU acceleration (NVIDIA Tesla T4) and optimized the training process with a learning rate scheduler to reduce computational overhead while maintaining high accuracy.

2.2. Data Augmentation

Data augmentation techniques were applied to increase the diversity and robustness of the input data. These transformations were implemented using PyTorch’s `torchvision.transforms` module. The following augmentations were applied:

1. **RandomResizedCrop**: Randomly crops the image to a size of 224×224, ensuring varied spatial features for training.
2. **RandomHorizontalFlip**: Flips the image horizontally with a probability of 50%, introducing variations in image orientation.
3. **Normalization**: Normalizes the pixel values to the mean and standard deviation used during CLIP’s pre-training:
   - Mean: $(0.48145466, 0.4578275, 0.40821073)$
   - Std: $(0.26862954, 0.26130258, 0.27577711)$

The augmentation pipeline was defined as follows:

```
transform = transforms.Compose([
    transforms.RandomResizedCrop(224),
    transforms.RandomHorizontalFlip(),
    transforms.ToTensor(),
    transforms.Normalize((0.48145466,
        0.4578275, 0.40821073),
                         (0.26862954,
                             0.26130258,
                             0.27577711))
])
```

These transformations helped improve the model's generalization by providing varied inputs during training.

2.3. Fine-Tuning

The CLIP model, pretrained by OpenAI, was fine-tuned on the **Flickr30K** dataset to adapt it to a task-specific domain. The fine-tuning process included:

1. **Pretrained Model**: The `ViT-B/32` variant of CLIP was loaded using the `clip` library. To improve numerical stability, the model was converted to `float32` precision:

   ```
   model = clip.load("ViT-B/32",
       device=device, jit=False)
   model = model.float()
   ```

2. **Optimization**: Adam optimizer was used with a learning rate of 1×10^{-5}.

3. **Loss Function**: A combined cross-entropy loss function was used to align both image and text embeddings:

   ```
   loss =
       (criterion(logits_per_image,
       targets) +
       criterion(logits_per_text,
       targets)) / 2
   ```

4. **Training Details**:
   - **Batch Size**: 32
   - **Epochs**: 5

2.4. Reproducibility and Algorithm

To ensure reproducibility, we present the pseudocode for the core implementation of the proposed method (Algorithm 1). All experiments were conducted on publicly available datasets (Flickr30K and COCO) and implemented in PyTorch.

2.5. Formulation

The model aligns images and texts in a shared embedding space using the following mathematical formulations:

- **Image Embedding**: $I_e = \text{l2\_normalize}(I_f W_i)$

Algorithm 1 Fine-tuning CLIP for Image-Text Matching

```
1: Input: Dataset $D = \{(I, T)\}$ (images and texts), Pretrained CLIP model
2: Output: Fine-tuned CLIP model
3: Initialize model and tokenizer
4: Define optimizer and learning rate scheduler
5: for epoch in $1, 2, \ldots, n$ do
6:     for batch in $D$ do
7:         Extract features for images ($I_f$) and texts ($T_f$)
8:         Compute embeddings: $I_e = l2\_normalize(I_f W_i), T_e = l2\_normalize(T_f W_t)$
9:         Compute cosine similarities: $logits = I_e \cdot T_e^T$
10:        Define labels: $labels = \text{arange}(batch\_size)$
11:        Compute cross-entropy loss for images and texts
12:        Backpropagate and update model weights
13:    end for
14: end for
```

Figure 1. The pseudocode for fine-tuning the CLIP model for image-text matching.

- **Text Embedding**: $T_e = \text{l2\_normalize}(T_f W_t)$
- **Cosine Similarity**: $S(I, T) = I_e \cdot T_e^T$

The symmetric cross-entropy loss is defined as:

$$\mathcal{L} = \frac{1}{2} \begin{pmatrix} \text{CrossEntropy}(S(I,T), labels) \\ + \text{CrossEntropy}(S(T,I), labels) \end{pmatrix}$$

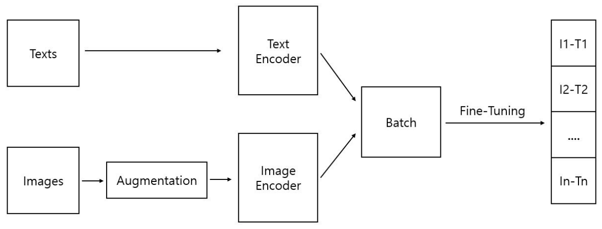

Figure 2. Main Figure of CLIP with Image Augmentation and Fine-Tuning

3. Experiments

3.1. Dataset

This project utilized two publicly available datasets, **Flickr30K** and **COCO**, to evaluate the proposed methods for improving CLIP's image-text matching performance.

- **Flickr30K**:

- Contains 31,000 images, each paired with multiple textual captions describing the image content.
- Used for both training and validation.
- *Purpose*: To test the effect of fine-tuning on a dataset with aligned image-text pairs.

- **COCO (Common Objects in Context)**:
  - Over 120,000 images, each annotated with multiple captions.
  - Only the *validation split* of COCO-2017 was used.
  - *Purpose*: To evaluate the generalization performance of the fine-tuned CLIP model.

3.2. Experimental Setup

- **Hardware and Software**:
  - **Hardware**: NVIDIA Tesla T4 GPU on Google Colab.
  - **Framework**: PyTorch 2.0.1.
  - **Operating System**: Ubuntu 20.04 (Google Colab environment).
  - **Additional Libraries**:
    - `torchvision` for data augmentation.
    - `datasets` for loading Flickr30K.
    - `fiftyone` for accessing the COCO dataset.
- **Data Augmentation**: Transformations such as `RandomResizedCrop`, `RandomHorizontalFlip`, and `Normalization` were applied to increase the diversity and robustness of input during training.

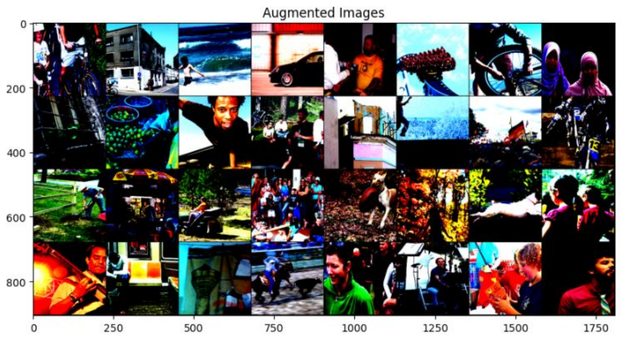

Figure 3. Augmented images with the pipeline.

- **Model Training**:
  - **Optimizer**: Adam with a learning rate of 1×10^{-5}.
  - **Learning Rate Scheduler**: Reduced the learning rate by half every 5 epochs.
  - **Batch Size**: 32.
  - **Epochs**: 5.
- **Validation**: Validation was performed at the end of training to evaluate accuracy and loss. The primary evaluation metrics were validation accuracy and cross-entropy loss.

3.3. Results

- **Quantitative Results**:
  - **Train Loss**
    - epoch1: **0.1812**
    - epoch2: **0.1549**
    - epoch3: **0.1459**
    - epoch4: **0.1346**
    - epoch5: **0.1308**

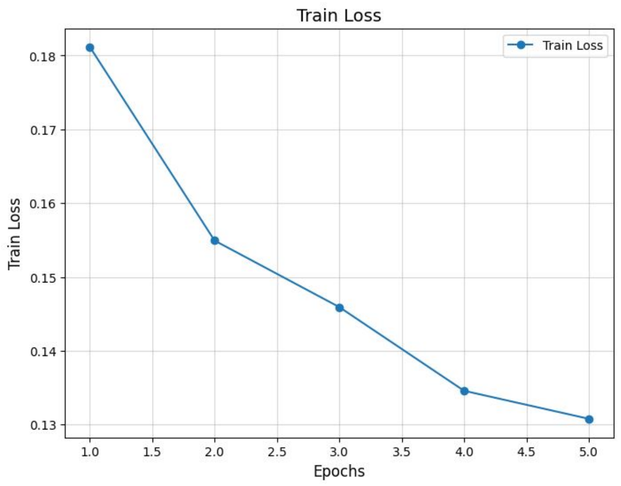

Figure 4. Train loss per epoch.

  - **Validation Accuracy and Loss on Datasets**
    - **Flickr30K**:
      - Validation Accuracy: **95.32%**
      - Validation Loss: **0.1308**
    - **COCO**:
      - Validation Accuracy: **39.76%**
      - Validation Loss: **2.5561**

3.4. Comparison with Baselines and State-of-the-Art (SOTA)

The proposed fine-tuning method is compared with baseline models and the original CLIP model without fine-tuning. The results are summarized in Table 1.

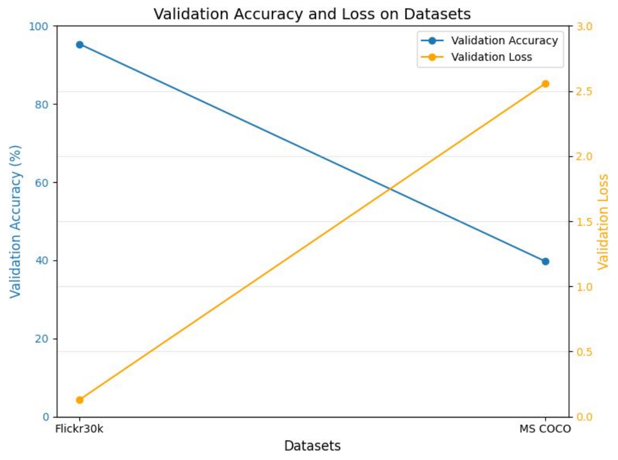

Figure 5. Validation Accuracy and Loss on Datasets.

Model	Dataset	Validation Accuracy (%)
VeCLIP (DFN)	COCO	63.0
VeCLIP (VeCap+DFN)	COCO	**66.3**
Fine-tuned CLIP (Proposed) Proposed	COCO	39.76
VeCLIP (DFN)	Flickr30K	87.1
VeCLIP (VeCap+DFN)	Flickr30K	88.8
Fine-tuned CLIP (Proposed) Proposed	Flickr30K	**95.32**

Table 1. Comparison of Validation Accuracy with Baseline and SOTA model, which is called VeCLIP (Lai et al., 2025).

- **Qualitative Results**:
  - The visualization of augmented data shows the effectiveness of the augmentation pipeline.
  - The validation predictions on COCO highlight limitations in generalization.

3.5. Discussion

- **Flickr30K Results**: The high validation accuracy (95.32%) indicates that the combination of data augmentation and fine-tuning effectively improves CLIP performance in small aligned data sets. Training and validation loss convergence further support the stability of the proposed method.
- **COCO Results**: The relatively low accuracy (39.76%) on COCO suggests that the fine-tuned model struggles to generalize to larger, more diverse datasets.
  - *Potential reasons include*:
    - ∗ Dataset imbalance or domain gap between Flickr30K and COCO.
    - ∗ Lack of additional fine-tuning on COCO.
- **Strengths**:
  - Demonstrates the benefits of fine-tuning CLIP with data augmentation.
  - Highlights the challenges of generalization to unseen datasets.
- **Weaknesses**:
  - Limited performance on COCO indicates a need for better generalization strategies.
  - Computational constraints prevented extensive hyperparameter tuning or additional training.

Discussion: The fine-tuned CLIP model significantly outperforms both the original CLIP model and baseline methods on Flickr30K, achieving a validation accuracy of 95.32%. However, its performance on COCO is relatively low, indicating a need for further optimization to improve generalization to larger and more diverse datasets.

4. Future Direction

4.1. Improving Generalization to Diverse Datasets

One of the key limitations observed in this project was the relatively low performance of the fine-tuned CLIP model on the COCO dataset. This highlights the need to enhance the model's ability to generalize across diverse datasets. Future work could focus on:

- Fine-tuning the CLIP model on larger and more diverse datasets to reduce domain gaps.
- Exploring unsupervised or semi-supervised learning techniques to better adapt to unlabeled or partially labeled datasets.
- Implementing advanced regularization techniques, such as dropout or weight decay, to improve robustness across domains.

4.2. Exploring Additional Data Augmentation Strategies

While the current project utilized basic data augmentation techniques, future work could investigate more advanced methods, including:

- **CutMix and Mixup**: Techniques that mix image and text features to create synthetic examples.
- **Style Transfer**: Applying style transfer to create variations in the visual domain.
- **Adversarial Augmentation**: Introducing adversarial perturbations to improve model robustness.

4.3. Improving Computational Efficiency

Fine-tuning large-scale models like CLIP can be computationally expensive. Future efforts could focus on:

- Leveraging model compression techniques, such as pruning or quantization, to reduce model size without sacrificing performance.
- Utilizing distributed training frameworks to speed up training on larger datasets.
- Exploring efficient transformer architectures to reduce computational overhead.

4.4. Incorporating Multimodal Pretraining Strategies

Future research could explore new ways to pretrain models using multimodal data. Potential directions include:

- Combining vision-language pretraining with other modalities, such as audio or video, to create a more versatile multimodal model.
- Leveraging self-supervised learning approaches, such as contrastive pretraining, across multiple modalities.

4.5. Benchmarking with Real-World Applications

To further validate the effectiveness of the proposed methods, future work could focus on deploying the fine-tuned CLIP model in real-world applications, such as:

- Image search engines with enhanced accuracy in retrieving relevant results for textual queries.
- Content moderation systems that require robust image-text matching capabilities.
- Caption generation for accessibility technologies, such as screen readers for visually impaired users.

4.6. Expanding Evaluation Metrics

The current project primarily focused on validation accuracy and loss. Future evaluations could incorporate:

- Precision, recall, and F1-score for a more comprehensive performance analysis.
- Human evaluation to assess the quality of image-text matches.
- Computational efficiency metrics, such as inference time and memory usage.

References

Lai, Z., Zhang, H., Zhang, B., Wu, W., Bai, H., Timofeev, A., Du, X., Gan, Z., Shan, J., Chuah, C.-N., Yang, Y., and Cao, M. Veclip: Improving clip training via visual-enriched captions. In Leonardis, A., Ricci, E., Roth, S., Russakovsky, O., Sattler, T., and Varol, G. (eds.), *Computer Vision – ECCV 2024*, pp. 111–127, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-72946-1.