Self-Supervised CLIP Fine-Tuning with Medical Image-Text Data

Lukas Ingemarsson

Abstract

Contrastive Language-Image Pre-Training (CLIP) is a multimodal model which connects images and text through large-scale pre-training, and has been greatly acclaimed ever since its release. Fine-tuning the base pre-trained model for image classification has shown increasing interest, especially in the medical domain. Generally, for this purpose, CLIP has been fine-tuned with image-label pairs, but labeled data may not always be available. In this paper, we fine-tuned one CLIP model instance with medical image-text data, and another instance with medical image-label data, to measure how much they affected the base performance. Our results showed that fine-tuning with image-text data provided a significant performance increase compared to the base model. In turn, as expected, fine-tuning with image-label data performed even better, but the performance increase provided by image-text fine-tuning is not negligible, and is worth considering when working with unlabeled data.

1 Describe

This section covers the key concepts of our project, the proposed method, and the obtained results together with relevant analysis.

1.1 Introduction

The Contrastive Language-Image Pre-training (CLIP) model by OpenAI, connects images and natural language text by learning joint high-dimensional vector representations, or *embeddings*, for the two modalities. In the embedding space, related images and texts are close to each other, while dissimilar examples are further apart.

By pre-training on a dataset with 400M image-text pairs, CLIP has displayed impressive *zero-shot* performance in various downstream tasks (Radford et al., 2021). Zero-shot learning refers to a model's ability to predict class labels that were not observed during pre-training.

To further improve downstream performance for more specific tasks, it may be desirable to fine-tune CLIP on domain-specific data, which presumably was not abundant during pre-training. This interest has been evident in the medical image domain, where in recent years the amount of published literature has seen a great increase (Zhao et al., 2024).

Naturally, for image classification tasks, it is preferred to fine-tune using image-label pairs. Although, such labeled data may not always be available, and is expensive to produce. If one has access to descriptive text that corresponds to each image, one could still perform fine-tuning using the same self-supervised approach used during CLIP's pre-training process.

For that reason, the focus of this project was to fine-tune one CLIP instance on medical image-text pairs, another instance on medical image-label pairs, and compare their image classification performance to that of the base pre-trained CLIP model. The class labels that we chose for this task were different body parts.

1.2 Method

To make the contents of our method easier to follow, an overview of the pipeline is displayed in Figure 1. First, we downloaded the PubMedVision dataset (Chen et al., 2024) from HuggingFace[1]. The fields of interest were those containing the image, caption, and body-part label, respectively.

Apart from the general processing that was necessary to properly handle the dataset, a notable inconsistency was that the label field contained non-label information for some entries. This was resolved by simply excluding the invalid entries.

After filtering the dataset, it contained 533,076 entries, which would be too extensive to fully pro-

[1] `https://huggingface.co/datasets/FreedomIntelligence/PubMedVision`

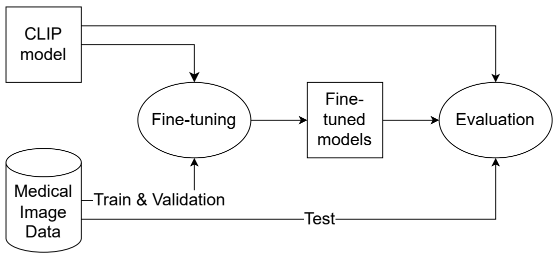

Figure 1: Fine-tuning pipeline.

cess and test given the time and resource constraints of our project. Therefore, after testing different sample sizes to estimate the training time, we settled for a sample of 4,096 entries from the dataset, which we distributed 80-10-10 between the training, validation, and test set. For the training set, we sampled the same number of examples from each class, to avoid inducing bias during fine-tuning. For the validation and test set, samples were drawn from each class proportionally to the full dataset's label frequency distribution, to more accurately represent the expected evaluation for the full dataset.

As we proceeded to the fine-tuning steps, we used a method proposed by Goyal et al. (2022), which is to fine-tune with the same loss that was used during pre-training. CLIP uses *contrastive loss* for pre-training, and we do the same for fine-tuning to follow this principle.

The objective of contrastive loss is to maximize similarity between each image and its associated text, while minimizing similarity to all other images and texts. As previously mentioned, CLIP encodes images and text into a shared embedding space, enabling direct similarity calculations between the modalities. For a batch of size N, the model generates $N \times E$ image and text embeddings, where E is the embedding size. Then, the model calculates the cosine similarity between these embeddings to create a $N \times N$ similarity matrix. Image-to-text similarities can be observed row-wise in the matrix, and text-to-image similarities column-wise in the matrix. Cross-entropy loss is calculated for both the image-to-text and text-to-image similarities, and the total loss is simply the average of the two.

Finally, we evaluated the two fine-tuned models and the base pre-trained model against the test set. During fine-tuning of the image-label model, as well as evaluation, labels were prepended with the context-enhancing string "A medical image of ". The idea behind this is to aid CLIP in understanding the context of the data. Furthermore, the hyperparameters of the fine-tuning configuration were tweaked to maximize downstream performance. Notably, the learning rate was downscaled from the one used during CLIP's pretraining process, in proportion to how much smaller our batch size was.

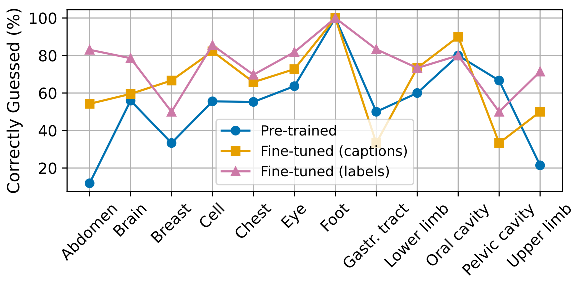

Figure 2: Correct predictions per class (%) for each model. (Gastr. tract = Gastrointestinal tract)

1.3 Results & Analysis

To evaluate the performance of the base pre-trained model against our fine-tuned models, we recorded the top-1 and top-3 accuracies for each model, as can be seen in Table 1. Our caption-fine-tuned model outperformed the pre-trained model, with improvements of approximately 15 points in top-1 accuracy and 4 points in top-3 accuracy. Unsurprisingly, the label-fine-tuned model showed even greater improvement, achieving increases of around 28 points in top-1 accuracy and 13 points in top-3 accuracy, compared to the pre-trained model.

Model	**Top-1** (%)	**Top-3** (%)
Pre-trained	49.4	78.9
Fine-tuned (captions)	64.3	83.1
Fine-tuned (labels)	76.9	92.1

Table 1: Model top-1 and top-3 accuracy scores.

Figure 2 displays the prediction accuracy per body part of each model during evaluation. The general trend shows the same relative performance as for the top-1 and top-3 accuracies between the three model variants, indicating that the performance improvement is relatively consistent across different classes.

Although our experiments were specifically based on medical image data, our findings should be generalizable to any domain-specific dataset with a comparable format to PubMedVision. As expected, the label-fine-tuned model exhibited the best performance out of the three candidate models. However, our results also show that fine-tuning CLIP on image-text data can yield significant improvements when performing image classification compared to the base CLIP model. Therefore, in the absence of labeled data, this may be a valid approach to improve downstream performance.

2 Examine

In this section, I examine my experience throughout the project, highlighting the aspects that were most impactful for my personal learning.

Understanding CLIP

To effectively work with the core model of our project, CLIP, it was important to have a solid understanding of how it functions in theory.

When image-text data is input into CLIP, at least in the version that we used in the project, it encodes images using a vision transformer and text using a masked self-attention transformer (Radford et al., 2021). Thanks to the coverage of the transformer architecture and the attention mechanism in the second unit of the course, I could better visualize the components behind the image-text encoding.

Furthermore, my knowledge from the first unit about tokenization and embeddings, allowed me to understand and interpret how the input text was decomposed into tokens before encoding, as well as how the encoders generate high-dimensional real-valued vector representations. This in turn enabled me to grasp how CLIP brings image and text data into the same embedding space, and that this makes it possible to directly calculate the similarity between examples from the two modalities – a key idea in the CLIP pipeline. This showcased the power of embeddings in a way that the definition itself, as presented in the course, could not fully convey on its own.

Another concept that was necessary to understand the inner workings of CLIP, although not covered in the course, was contrastive loss. I acquired the necessary information regarding the type of contrastive loss used by CLIP from the CLIP paper itself (Radford et al., 2021), and could dive a bit deeper into the idea behind the loss by reading the pioneering paper that inspired the CLIP authors (Zhang et al., 2020). This theoretical knowledge was not only important to comprehend how CLIP works, but also to effectively debug and make the implementation work as expected.

Data Processing

Loading and processing data is rarely a convenient process, and this was evident in the project when we were testing out different datasets. Thankfully, as we worked with parsing data, forming loadable batches, and more in several of the labs in the course, the challenge of converting raw data into a manageable format was significantly reduced.

However, an aspect that we did not practice during the labs was handling and filtering out inconsistencies in the data that we are working with. In the project we resorted to the Python library Pandas[2], which provided sufficient utility to structure and filter the data and thereby extract a valid subset. After this project, my impression is that familiarity with a data manipulation tool of this kind is very useful, and is in many cases crucial to apply the concepts presented in the course in practice.

Another question that I encountered during the project, that we did not discuss during the course, was from which distribution we can or should draw samples for the training set in comparison to the evaluation sets. When training for classification tasks, it is desirable to mitigate the bias introduced by class imbalance. There are different ways of approaching this, but we chose to use *undersampling*, where the idea is to only sample as many examples from each class as the number of entries of the most infrequent class in the dataset (Rawat and Mishra, 2022). In our case, as we did not use the full dataset, this instead meant choosing a number of samples that did not exceed the number of examples of the most infrequent class, and accordingly sampling that many examples from each other class. On the contrary, for our evaluation sets, we drew samples from the class-frequency distribution of the full dataset, so that our results would be more representative of a full dataset evaluation (Stando et al., 2023). Given our limited time frame and lack of resources, this was an effective approach, as the smaller experiments were feasible and yielded results suitable for meaningful inter-model comparison and analysis.

Fine-Tuning & Evaluation

As we moved on to fine-tuning and evaluating our models, we needed to implement the training pipeline, including the contrastive loss function. This was not excessively difficult, much thanks to the fact that we could utilize the training-loop code

[2] https://pandas.pydata.org/

from the labs as a reference for our implementation.

Although, we did run into an interesting issue related to the model hyperparameters. CLIP uses the batch size 32,768 during pre-training whereas we were limited to 64 for fine-tuning due to memory constraints. Initially, we simply copied the same learning rate used to pre-train CLIP, without taking the difference in batch size into consideration. As a result, after fine-tuning, our models appeared to be guessing randomly, which suggested that their memory had essentially been completely erased; a phenomenon known as *catastrophic forgetting* (Kirkpatrick et al., 2017). We therefore reduced our learning rate in proportion to the ratio between our batch size and the original, which solved this problem and gave significantly better evaluation performance.

In Hindsight: Insights & Ideas

This project has provided various valuable insights that strengthened my understanding of the content from the course, simultaneously improving my confidence in handling these concepts in practice.

Working with CLIP helped me deepen my understanding of how transformer-based models can process and align language with other modalities. If I were to redo the project today, I would like to explore making different changes to the model configuration/architecture to hopefully improve downstream performance further.

An observation that I unfortunately discovered towards the end of the project was how great of an impact prompt engineering could have on the model's alignment ability. The rudimentary addition of prepending a context-enhancing string to labels during evaluation offered roughly a 15 percentage point increase across all models for top-1 accuracy. Had I redone the project now, I would have liked to look into and test other prompt variations as well as more advanced prompt engineering techniques. It would also have been interesting to investigate the possibilities of engineering the text examples in the data, in addition to the labels.

Our experiments were downscaled in terms of the data quantity, training time, and efforts made to tune hyperparameters, mainly to conform with the time and resource constraints of the project. If I were to do the project again, I would have suggested distributing the experiment workload among the group members' hardware, to streamline the fine-tuning process and increase the capacity of the training pipeline in the aforementioned regards.

3 Articulate

Prior to this project, I had not considered the aspect of fairly selecting data, and how the definition of fairness can differ depending on the data's use case. I gained this insight by critically analyzing our pipeline to identify potential factors that could cause invalidity in our results. This experience highlighted the importance of taking data fairness into account when sampling data, both for training and evaluation, which is something that I will make sure to consider in future projects.

Another lesson was the initial fine-tuning attempt that simply guessed randomly, which taught me that you should not neglect the choice of hyperparameters when configuring and training a model. I learned this by naively adopting CLIP's pre-training hyperparameters for fine-tuning, without acknowledging the differences in their configuration. Failing to consider this can not only cause significant problems, but also make debugging very difficult, making this lesson an important one.

Similar roadblocks appeared often throughout the project, and despite causing frustration at times, they ultimately developed my ability to handle NLP literature. In pursuit of ideas or explanations, it was crucial to learn how to quickly find relevant research articles online, analyze their content, and based on that, identify if any information was applicable for our project. Had I not worked on this skill, the search process would have consumed more time, leaving less space for implementation.

As done in this project, fine-tuning CLIP with domain-specific data for different downstream tasks has been researched in plenty of existing literature. I believe that the novelty of our project lies in the fact that we analyze fine-tuning CLIP on data that does not directly match the data format of the downstream task. More specifically, we fine-tune CLIP with image-text data but want the model to make image-label predictions during inference. The scarcity of research for this particular case is understandable, as it is of interest to search for training data that aligns with the input-output behavior of the downstream task. However, as stated in the project description, there may be situations where we have data of a different format than what is applied during inference. Our work shows that such mismatched data can offer great value in increasing downstream performance, which is especially interesting when data that matches the inference format is unavailable.

References

Junying Chen, Ruyi Ouyang, Anningzhe Gao, Shunian Chen, Guiming Hardy Chen, Xidong Wang, Ruifei Zhang, Zhenyang Cai, Ke Ji, Guangjun Yu, Xiang Wan, and Benyou Wang. 2024. Huatuogpt-vision, towards injecting medical visual knowledge into multimodal llms at scale. *Preprint*, arXiv:2406.19280.

Sachin Goyal, Ananya Kumar, Sankalp Garg, Zico Kolter, and Aditi Raghunathan. 2022. Finetune like you pretrain: Improved finetuning of zero-shot vision models. *Preprint*, arXiv:2212.00638.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Preprint*, arXiv:1612.00796.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. *Preprint*, arXiv:2103.00020.

Satyendra S. Rawat and Amit K. Mishra. 2022. Review of methods for handling class-imbalanced in classification problems. *Preprint*, arXiv:2211.05456.

Adrian Stando, Mustafa Cavus, and Przemysław Biecek. 2023. The effect of balancing methods on model behavior in imbalanced classification problems. *Preprint*, arXiv:2307.00157.

Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D. Manning, and Curtis P. Langlotz. 2020. Contrastive learning of medical visual representations from paired images and text. *Preprint*, arXiv:2010.00747.

Zihao Zhao, Yuxiao Liu, Han Wu, Mei Wang, Yonghao Li, Sheng Wang, Lin Teng, Disheng Liu, Zhiming Cui, Qian Wang, and Dinggang Shen. 2024. Clip in medical imaging: A comprehensive survey. *Preprint*, arXiv:2312.07353.