Commentary on Homework – CS 182

Haoxing Du, Sid Jha, Zipeng Lin, Rahul Shah

December 2022

1 Introduction

Vision Transformers (ViT) are a type of artificial neural network architecture that is used in computer vision tasks. They were introduced in a paper by Dosovitskiy et al in 2020 [3].

Vision Transformers are based on the Transformer architecture, which was originally developed for natural language processing tasks. In the case of Vision Transformers, the Transformer architecture is adapted for use in tasks such as image recognition and object detection.

One key aspect of Vision Transformers is that they are designed to be more efficient and scalable than other computer vision models. This is achieved by using a self-attention mechanism, which allows the model to process input data in a parallel manner rather than in a sequential manner. This allows the model to process data more quickly and accurately.

Overall, the concept of Vision Transformers is to use the Transformer architecture in the field of computer vision in order to improve the efficiency and performance of artificial neural networks. They have been shown to out-perform Convolutional Neural Networks (CNNs), the traditional architecture for vision tasks, because they are better able to process long-range relationships that exist

in vision data.

2 Assignment design

2.1 Choice of language framework

For this assignment, we choose to use JAX [1] as our framework. The reason behind this is that it is quite different PyTorch which is what current iteration of CS182 uses. For instance, there are functions such as `jax.pamp` that let students get a taste of functional programming in machine learning. Besides, student will another way to implement neural network layers and taking the gradient, so it is a way for student to explore other machine learning frameworks. Also, we show students how power some functions that flax, a system built to make JAX better [4]

2.2 Key Concepts

We decided to make fine-tuning the key concept in the homework assignment. To do this, we had students load in pre-trained weights for the ViT and test its accuracy on CIFAR-10 without fine-tuning. The dataset used in pre-training is ImageNet, which is a much more complex dataset than CIFAR-10, and provides a solid foundational for image fine-tuning tasks. Students are asked to fine-tune their model on CIFAR-10 and compare the results before and after: over 90% accuracy can be achieved with a single epoch of training, taking about a minute on a fast GPU (a bit longer on Colab). One important point we wanted to get across to the students was the importance of having fine-tuning begin with weights that capture some structure of the problem. To emphasize this, we then had students redo the training but with randomly initialized weights instead of the pre-trained weights. The accuracy after training for one epoch

with randomly initialized weight is much worse, only less than 20%.

3 Structure of the assignment

We have two files for student to fill their code in, including a notebook to run. In the file `vit_architecture.py`, we ask students to implement the embeddings, such as identity embedding, and how to incorporate embedding into the transformer model. For each step in the file `vit_architecture.py`, we describe what the module in the model does and provide sufficient hints for the student to complete code in sufficient time. The second file is `vit_homework.ipynb` which serves to display the students the results of the model they have implemented. In the notebook, students can see their model, if implemented correctly, would reach high accuracy under a short period of time. Besides, there would be plots generated for them to show the loss and learning rate over the time. Students could evaluate the correctness of the code by looking at the plots and the training result.

4 Going Above and Beyond

This homework assignments allows students to go above and beyond what was presented in the original paper in several ways. For example, the assignment asks students to implement the embeddings for the Vision Transformer (ViT) model and incorporate them into the model, which is not discussed in the paper. Often times, students in EECS are obsessed with theory yet lack practical implementation skills – which this assignment gives.

Additionally, the assignment includes a section where students are asked to train the model using randomly initialized weights, to demonstrate the importance of using pre-trained weights when fine-tuning, which goes beyond what

was discussed in the paper. Furthermore, the assignment introduces students to not only the JAX framework and the use of learning rate schedulers [2] – which are not discussed in the paper but are important practical considerations when training deep learning models – but it also teaches students how to deal with implementations at a fundamental level. Frameworks like JAX come and go but the knowledge of how embeddings work gained by completing this assignment extends beyond any one framework. Overall, the proposed assignment provides a more hands-on and practical application of the ideas presented in the paper. The inclusion of this systematic experimentation with explainable results in such an easy to understand manner is a point of notice.

References

[1] JAX Documentation. "JAX Documentation". In: *https://jax.readthedocs.io/en/latest/notebooks/quickstart.html* (2022).

[2] PyTorch 1.13 Documentation. "Cosine Annealing LR." In: *https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.CosineAnnealingLR.html* (2022).

[3] Alexey Dosovitskiy et al. "An image is worth 16x16 words: Transformers for image recognition at scale". In: *arXiv preprint arXiv:2010.11929* (2020).

[4] Flax.jax_utils Package. "Flax Jax utility Package." In: *https://flax.readthedocs.io/en/latest/api_reference/flax.jax_utils.html* (2022).