Exploring ViT for Image Classification on CIFAR-10

COMP8539: Advanced Topics in Computer Vision

1 Introduction

Vision Transformers (ViTs) are a class of models that have achieved state-of-the-art results in computer vision, often outperforming traditional Convolutional Neural Networks (CNNs). In this project, you will gain hands-on experience implementing, training, and analyzing ViT models for image classification on CIFAR-10. You will explore how different architectural choices and modern training techniques affect model performance.

This is a research-based assignment. We provide a set of guided experiments, but you should also think critically, formulate your own hypotheses, and test them. The goal is understanding *why* certain methods work, not just achieving high accuracy. You are encouraged to explore techniques beyond those listed here.

2 Project Goals

The objectives of this project are to:

- Understand the ViT architecture.
- Implement and modify deep learning models in PyTorch / JAX.
- Investigate how architectural parameters (e.g., patch size, model dimension, and number of layers) impact performance.
- Implement and evaluate modern training techniques for transformer models.
- Analyze experimental results and communicate your findings clearly.
- Compare ViT performance and characteristics to a standard ResNet.

3 Tasks

The tasks below offer a path for exploration. You are not required to implement every item; choose the experiments you find most interesting. You are also encouraged to research and implement other techniques.

3.1 Basic Task 1: Baseline ViT Implementation, 3 marks

Implement and train a baseline ViT model on CIFAR-10 to establish a performance baseline.

- Implement a standard ViT architecture. You may refer to the paper and the existing implementation for guidance.

- Train the baseline model on CIFAR-10 until convergence. Record the training and validation accuracy/loss curves.

Report the final test accuracy of your baseline model.

3.2 Basic Task 2: Ablation Studies on ViT Architecture, 3 marks

With a working baseline, investigate the effect of patch size and model size (embedding dimension).

- **Patch Size:** Train models with different patch sizes (e.g., 2×2, 4×4, 8×8). How does patch size affect the trade-off between performance and computational cost?
- **Model Size:** Experiment with different embedding dimensions for the transformer (e.g., 96, 192, 256) and number of layers (e.g., 4, 6, 12). How does model capacity influence performance on a smaller dataset like CIFAR-10?
- **Attention Heads:** Train models with different numbers of attention heads in the self-attention layers (e.g., 1, 3, 8, 96). Analyze how the number of heads influences classification accuracy and computational cost.

For each experiment, you should compare the results to your baseline and provide a brief analysis of your observations. Finally, find the best combination and report the results on CIFAR-10.

3.3 Beyond the above given materials - some directions, 4 marks

You could try implementing and evaluating modern techniques that improve Transformer stability and performance. These are optional. You will get high marks for this section if you explore a lot.

- **Overlapping Patch Embedding:** Replace the standard non-overlapping patch projection with a convolutional patch embedding that uses overlapping patches (as in Swin Transformer). Analyze whether overlapping patchification improves local feature modeling and downstream performance on this dataset.
- **Normalization Layers:**
  - Replace the standard LayerNorm with RMSNorm. Analyze the effects on training stability, computational efficiency, and final performance.
  - Apply QK-Norm as an additional normalization technique in the attention mechanism. Evaluate its impact on training stability and overall performance.
- **FFN Variants:** Try alternative feedforward designs such as $\text{FFN}_{\text{SwiGLU}}$, and compare their effects on model performance; remember to adjust hidden dimensions to keep model sizes comparable, as these variants typically increase parameter count.
- **Position Embedding Variants:** Explore different strategies for incorporating position information into the model, and compare their effects.
  - Consider whether position embeddings should also be applied to the [CLS] token, and explain your reasoning.
  - **Fixed Sinusoidal Embeddings (1D/2D):** Use the original sinusoidal method, treating patches either as a sequence (1D) or as a spatial grid (2D); refer to the Transformer paper, the ViT paper, and the code repository.

  - **Learnable Position Embeddings:** Use trainable position vectors initialized randomly; refer to the Convolutional Sequence Learning paper, the Transformer paper, and the code repository.
  - **Rotary Position Embeddings (RoPE):** Apply a fixed rotary transformation to queries and keys that encodes relative position information directly into the attention mechanism; refer to the paper and the code repository.
- **Model Registers:** Implement model registers, which are extra learnable tokens added to the input sequence to improve feature representation.

For each technique, you should provide a brief explanation of how it works and analyze its impact on your ViT model's performance.

You can also try other datasets.

Find the best combination and compare your new ViT with the one from Basic Task 2.

4 Suggestions for your presentation, which is worth 3 marks

No matter how much work you do, it is important that your work is properly analysed. Your final presentation should address the following questions:

- Which architectural choices and "tricks" improved the performance of your ViT model on CIFAR-10? Which ones did not? Provide hypotheses for your observations.
- Despite these improvements, ViT models often underperform compared to a well-tuned ResNet on smaller datasets like CIFAR-10. Why do you think this is the case? One way to investigate this is by performing Principal Component Analysis (PCA) on the feature representations (e.g., patch token embeddings) from your best ViT and a ResNet model. Visualize the feature spaces for different classes and discuss any differences you observe.

The presentation is a 6-12 minute presentation (will detail after enrollment is finalised) summarizing your work. Your presentation should cover:

- A brief overview of the techniques you chose to implement and explore.
- A summary of your key findings: what worked, what didn't, and your hypotheses for why.
- Visualizations of your results, including training curves and, optionally, a PCA analysis of the feature space.
- Your final conclusions about training Vision Transformers on a smaller dataset like CIFAR-10.
- A brief discussion of potential future experiments.
- Note that your presentation quality will affect marking in previous sessions, e.g., a cluttered presentation may cause tutors to miss critical things and lead to lower mark in Section 3. Therefore, it is important to make sure you clearly present what you have done.

Slides should be submitted to Wattle prior to your presentation.

5 Referenced Materials

This section shows all referenced materials for this project:

- **ResNet**
  - Paper: https://arxiv.org/abs/1512.03385
  - Weights: https://docs.pytorch.org/vision/main/models/generated/torchvision.models.resnet50.html
- **ViT**
  - Paper: https://arxiv.org/abs/2010.11929
  - Code: https://github.com/lucidrains/vit-pytorch
- **CIFAR-10 Dataset**
  - Dataset: https://www.cs.toronto.edu/~kriz/cifar.html
- **Overlapping Patch Embedding**
  - Paper: https://arxiv.org/abs/2103.14030
- **RMSNorm**
  - Paper: https://arxiv.org/abs/1910.07467
- **QK-Norm**
  - Paper: https://arxiv.org/abs/2010.04245
- **FFN Variants**
  - Paper: https://arxiv.org/abs/2002.05202
- **Position Embedding**
  - Transformer Paper: https://arxiv.org/abs/1706.03762
  - ConvSeq2Seq Paper: https://arxiv.org/abs/1705.03122
- **Rotary Position Embedding (RoPE)**
  - Paper: https://arxiv.org/abs/2104.09864
  - Code: https://github.com/lucidrains/rotary-embedding-torch
- **Model Registers**
  - Paper: https://arxiv.org/abs/2309.16588