Project Report - ECE 176

Keyi Chen
Mathematics & Cognitive Science
A16870360

Jayla Cho
Computer Science & Engineering
A18058562

Abstract

This project aims to implement and improve the approach presented in "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale"[2]. Transformer architecture was traditionally used in NLP, but is now leveraging for large-scale image recognition tasks. Our project targets three critical areas for enhancement: patch size and structure optimization, transformer architecture improvements, and regularization and training techniques. Adapting patch size and structure beyond 16x16 pixels could improve model sensitivity to crucial features. In addition, our project will refine the Transformer architecture itself to better capture the intricacies of image data. Finally, we expect regularization and advancing training techniques to help improve model robustness.

1 Introduction

1.1 Motivation

The motivation behind solving the problem lies in the growing importance of computer vision tasks in various domains such as healthcare, autonomous vehicles, and security systems. By developing an effective solution to the problem at hand, we aim to contribute to advancements in computer vision technology, which can lead to improved diagnosis in medical imaging, enhanced safety in autonomous driving, and better surveillance systems, among other applications.

1.2 Understanding of the problem

The problem revolves around implementing and enhancing the Vision Transformer (ViT) model, a state-of-the-art architecture for image classification tasks. Our understanding of the problem stems from a comprehensive review of the ViT paper by Dosovitskiy et al. (2020) and subsequent research in the field of transformer-based models for computer vision. We recognize the challenges associated with scaling ViT to larger image sizes, handling long-range dependencies, and achieving competitive performance compared to convolutional neural networks on standard benchmarks.

1.3 Approach

Our approach to solving the problem involves several key components. Firstly, we plan to implement the ViT model architecture following the specifications outlined in the original paper. This includes designing the transformer encoder, positional encoding, and classification head.

1.4 Summarization

1.4.1 Implementing ViT

Upon implementing the Vision Transformer (ViT) model, we achieved an accuracy of training data approximately 60%.

1.4.2 Improving ViT

After refining the ViT model with various enhancements, including adjusting hyperparameters and augmenting data, we observed a significant improvement in accuracy, reaching up to 90%. However, this enhancement led to overfitting issues, where the model performed well on the training data but struggled to generalize to unseen data.

1.4.3 Applying Enhanced ViT to STL-10 Dataset

We further experimented by applying the improved ViT model to the STL-10 dataset. While the accuracy increased to around 90%, similar to the CIFAR-10 dataset, overfitting remained a challenge, indicating the need for additional regularization techniques or model adjustments.

2 Related Work

2.1 An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. [2]

This seminal work by Dosovitskiy et al. introduces the concept of replacing convolutional layers with self-attention mechanisms in image classification tasks. They propose the Vision Transformer (ViT) model, which treats images as sequences of patches and applies transformer architectures to process them. This paper is highly relevant to our project as it serves as the foundation for our implementation of ViT in image classification tasks.

2.2 How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers[4]

The paper addresses various aspects to enhance the performance and efficiency of Vision Transformers (ViTs). Firstly, it tackles ViT's dependency on large-scale datasets by proposing methods to improve data efficiency. This is achieved through the introduction of advanced data augmentation techniques, which aid in regularizing the model and enhancing its generalization capabilities. Additionally, novel regularization strategies are suggested to mitigate overfitting, enabling ViTs to learn more robust features from the available data. Furthermore, the paper presents improvements in training procedures, resulting in better performance of ViTs across a range of vision tasks. Overall, these idea of the paper is collectively related to our project by advancing the effectiveness and versatility of ViTs in various real-world applications.

3 Method

Our tentative method involves the use of Vision Transformers (ViT) for image classification with some modifications.

3.1 Detailed Structure

Patch Embedding: The input image is first resized to a fixed dimension (96×96) and then divided into a grid of non-overlapping patches (16×16). These patches are flattened and linearly projected into a D-dimensional embedding space through a trainable linear projection, where D represents the number of dimensions in this space. This process effectively transforms the 2D image patches into a sequence of 1D token embeddings, preparing them for processing by the Transformer encoder. (NB: For a hybrid architecture, the input sequence can be formed from feature maps of a CNN as an alternative to raw image patches.)

Class Token Embedding: A special token, known as the class token ('[CLS]'), is prepended to the sequence of patch embeddings. This token is not associated with any image patch but is used to accumulate information across the Transformer encoder layers. The final state of this token, after passing through the Transformer encoder, serves as the aggregate representation of the input image for classification purposes.

Position Embedding: To incorporate spatial information into the sequence of embeddings, position embeddings are added to the patch embeddings and the class token embedding. These embeddings are learnable parameters that allow the model to understand the order and position of each patch in the image. The operation is represented as $E' = E + PE$, where E is the patch or class token embedding and PE represents the position embeddings.

Transformer Encoder Block ($\times$ depth): The Transformer Encoder Block is the core of our method designed to process sequences of embeddings, such as image patches in Vision Transformers, capturing intricate relationships and features. Each block consists of alternating layers of multi-headed self-attention (MSA) and Multi-Layer Perceptron (MLP) blocks, with residual connections after every block. Layer norm (LN) is applied before every block. Details following:

- **Layer Normalization (LN1) – Pre-Attention:** The input embeddings first undergo Layer Normalization (LN1) to stabilize the learning process.

  The formulation is given by $LN(x) = \gamma \frac{x-\mu}{\sigma} + \beta$, where μ and σ are the mean and standard deviation of x, and γ and β are learnable parameters.
- **Multi-Headed Self-Attention (MSA):** This mechanism allows the model to focus on different parts of the input sequence by computing attention scores.

  The attention function is defined as $Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$, where d_k is the dimension of the key vectors. This operation is performed across multiple "heads" for diverse representation subspaces.
- **Residual Connection – Post-Attention:** A residual connection adds the input of the MSA block to its output, helping to mitigate the vanishing gradient problem.

  Formulation as $x + F(x)$, where x is the input for LN1 layers, and $F(x)$ is the output from the MSA block.
- **Layer Normalization (LN2) – Pre-MLP:** A second LN2 layer is applied before passing the input to the Multi-Layer Perceptron (MLP) block, ensuring consistent training dynamics.
- **Multi-Layer Perceptron (MLP):** The MLP block, consisting of two linear layers with a non-linearity in between (e.g., GELU), is represented as $MLP(x) = W_2(\phi(W_1 x + b_1)) + b_2$, where W_1, W_2 are the weights, b_1, b_2 are the biases, and ϕ is the GELU function in our case.
- **Residual Connection – Post-MLP:** Similar to the post-attention phase, another residual connection sums the input to the MLP block with its output, enhancing gradient flow and network depth without performance degradation.

  Formulation as $x + F(x)$, where x is the input for LN2 layers, and $F(x)$ is the output from the MLP block.

By stacking multiple Transformer Encoder Blocks, the model iteratively refines the input representations, capturing highly complex patterns and dependencies, crucial for tasks like image classification.

Multi-Layer Perceptron Head: After processing through the Transformer encoder, the state of the '[CLS]' token is passed through a Multi-Layer Perceptron (MLP) head to produce the final class predictions. The MLP head typically consists of one or more linear layers followed by a softmax function to output probabilities over the target classes.

The formulation for a simple MLP head with a single hidden layer is $Y = softmax(W_2(\phi(W_1 C + b_1)) + b_2)$, where C is the '[CLS]' token representation, W_1, W_2 are the weights, b_1, b_2 are the biases, ϕ is a non-linear activation function (Tanh in our case), and Y represents the output class probabilities.

3.2 Training/Testing Algorithm

Training Algorithm: Our training algorithm utilizes the supervised learning for the labeled parts of the dataset. We trained the model on the on the labeled dataset to encourage the model to learn useful feature representations with the class labels. Subsequently, a cross-entropy loss is used for backtracking and fine-tuning.

Testing Algorithm: During testing, images are processed through the same patch embedding and position embedding steps, followed by the Transformer encoder and MLP head to generate class predictions. The dynamic position embeddings are fixed at this stage, using the learned embeddings from the training phase.

3.3 New Techniques

Parameter Reduction and Hyperparameter Tuning

Reason:

The decision to implement parameter reduction and hyperparameter tuning in Vision Transformers (ViTs) stems from several key considerations. Firstly, as ViT models grow in size and complexity to handle increasingly large-scale datasets and tasks, the computational demands and memory requirements also escalate significantly. By reducing the number of parameters, we address the challenge of model scalability and alleviate the burden on computational resources, making ViTs more practical and efficient for real-world applications.

Secondly, overfitting remains a persistent concern in deep learning models, especially with the proliferation of parameters in ViTs. By streamlining the model architecture through parameter reduction, we mitigate the risk of overfitting and enhance the model's ability to generalize well to unseen data. This ensures that ViTs maintain high performance across diverse datasets and scenarios, improving their reliability and applicability in various domains.

Strengths:

The strength of implementing parameter reduction and hyperparameter tuning lies in its ability to strike a balance between model efficiency and effectiveness. By optimizing the ViT architecture through these modifications, we achieve significant improvements in both computational efficiency and task performance. This enables ViTs to deliver competitive results with fewer parameters and reduced computational overhead, making them more accessible and practical for a wide range of applications.

Furthermore, the fine-tuning of hyperparameters empowers us to tailor ViT models to specific tasks or datasets, further enhancing their adaptability and robustness. This flexibility ensures that ViTs can excel in diverse settings and outperform traditional convolutional neural networks (CNNs) in various image recognition tasks.

4 Experiments

4.1 Introduction

In our experimental framework, we utilize both the CIFAR-10 and STL-10 datasets to evaluate the performance of our models. The CIFAR-10 dataset, which was employed in the original Vision Transformer (ViT) paper, serves as a benchmark for comparison between our implementation and the baseline established in the literature. This choice allows us to directly assess the enhancements our models bring to the table in a well-understood context. CIFAR-10's relatively modest size also facilitates a more rapid training process, enabling efficient experimentation and iterative refinement of our models at an early stage.

Furthermore, we extend our evaluation to the STL-10 dataset, a more challenging and diverse dataset that has not been explored in the original ViT study. This choice is motivated by our aim to test the robustness and scalability of our improved models under more demanding conditions. By applying our enhancement techniques to both CIFAR-10 and STL-10, we aim to demonstrate the versatility and improved accuracy of our models across different datasets. The incorporation of STL-10, with its unique characteristics and higher-resolution images, provides a comprehensive platform to validate the effectiveness of our proposed improvements in real-world scenarios.

4.2 Data Format

CIFAR-10

CIFAR-10 [3] is a widely used dataset in the field of computer vision. It consists of 60,000 color images, each of size 32x32 pixels, belonging to 10 different classes. These classes include common objects such as airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks. The dataset is split into 50,000 training images and 10,000 test images, with an equal number of images for each class.

The images in CIFAR-10 are generally low-resolution compared to some other datasets, which presents challenges for models to accurately classify objects. Despite its simplicity, CIFAR-10 remains a popular choice for researchers and practitioners in the field of deep learning.

STL-10

STL-10 [1] is introduced by Adam Coates, Honglak Lee, and Andrew Y. Ng in their 2011 paper, the dataset is designed to provide a more challenging testbed than the widely-used MNIST dataset, with a focus on developing algorithms that can learn feature representations from unlabeled data.

It consists of 13,000 color images, divided into 5,000 training images, 8,000 test images, and 100,000 unlabeled images for unsupervised learning tasks. STL-10 is inspired by the CIFAR-10 dataset but with some modifications. Each image is 96x96 pixels in size, significantly larger than the 32x32 pixels format of CIFAR-10 images, allowing for more detailed features. The dataset encompasses 10 classes, with each class represented equally in the labeled sets. The images are provided in a raw format, with separate files for labeled and unlabeled data, class labels, and fold indices for the standard 10-fold cross-validation setup.

4.3 Other Information

Here are some parameters we are using:

Parameter	Baseline ViT	Improved ViT
Learning Rate	0.001	0.001
Weight Decay	0.01	0.00001
Optimizer	AdamW	Adam
Loss	CrossEntropy	CrossEntropy
Dropout	0.1	0.1
Patch Size	4×4	4×4
# Attention Heads	12	4
# Hidden Layers	7	4
Hidden Size	384	48
# Epoch	100	100
# Parameters	6.5m	1.1m

Table 1: Parameters for Baseline ViT and Improved ViT configurations for CIFAR-10 and STL-10

4.4 Results

After conducting extensive training sessions for the three models under consideration, we tracked the evolution of their accuracy over time. This approach enabled us to capture the dynamic progress of each model's learning process, providing valuable insights into their performance and efficiency.

Figure 1 and Figure 3illustrates the accuracy trajectory for the baseline ViT model when applied to the CIFAR-10 and STL-10 datasets. The graph delineates how the model's accuracy evolves throughout the training epochs, offering a visual representation of its learning curve. One can see that the model quickly gets over-fitted at the first 20 epochs for both datasets with the baseline model with final accuracy at around 50%.

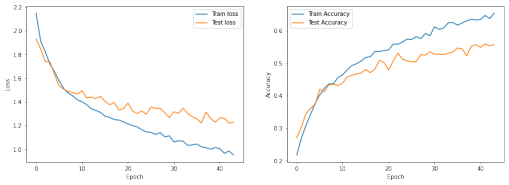

Figure 1: Baseline ViT for CIFAR-10.

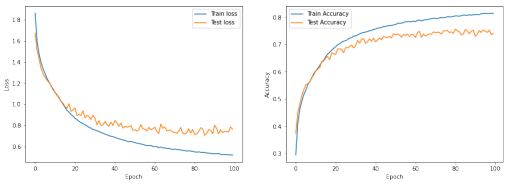

Figure 2: Improved ViT for CIFAR-10.

Figure 2 and Figure 4illustrates the accuracy trajectory for the improved ViT model when applied to the CIFAR-10 and STL-10 datasets. The graph delineates how the model's accuracy evolves throughout the training epochs, offering a visual representation of its learning curve. One can see that the improved model achieves a higher accuracy than the baseline, and smaller over-fitting throughout the training.

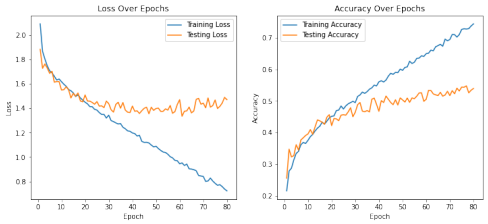

Figure 3: Baseline ViT For STL-10.

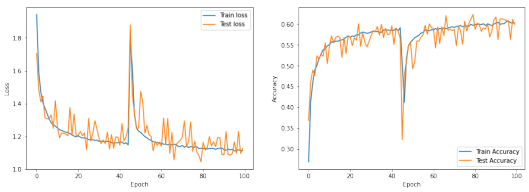

Figure 4: Improved ViT for STL-10.

5 Supplementary Material

You should also include a video recording a presentation (with motivation, approach, results) for this project.

References

[1] Adam Coates, Andrew Y. Ng, and Honglak Lee. An analysis of single layer networks in unsupervised feature learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011. `https://cs.stanford.edu/~acoates/papers/coatesleeng_aistats_2011.pdf`.

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[3] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[4] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers, 2022.