Vision Transformer Pruning

Bui, Vy Chen, Chiu-Chun Helms, Derek Mueller, Sebastian
Oregon State University
{buivy, chenchiu, helmsd, muellese}@oregonstate.edu

Abstract

Since its introduction in late 2020, the vision transformer architecture has shown promising results competitive with comparable CNN architectures in terms of performance. However, both the scale and required computational power of this new architecture have been limiting factors in its deployment on edge devices. In this paper, we present two methods for reducing the model complexity and computational requirements to aid in preparation of edge device deployment: block pruning and token pruning. By removing entire attention head blocks of the model architecture, and masking/removing uninformative tokens from the image sequence input, we present that vision transformer efficiency can be improved upon with minimal performance reductions while staying competitive with similar CNN architectures.

1. Introduction

Since their introduction by Vaswani et al. [14] in 2017, Transformers have become the dominant architecture choice for many natural language processing tasks. The Transformer architecture removes recurrence and convolutions in place of an attention mechanism in order to discover dependencies between the input and output representations [14]. Due to the scalability of this architecture, training large scale Transformer models has become a common practice. First introduced by Devlin et al. [3] for language tasks, the Bidirectional Encoder Representations from Transformers (BERT) base model contains 110M total parameters while the large version of the model contains 340M total parameters. Although increasing the size of a model can lead to improved performance on large-scale tasks, this scaling has also become a limiting factor in their deployment on edge devices.

In recent years, many researchers have begun to apply the Transformer architecture to computer vision (CV) tasks such as image classification and object detection. First introduced by Dosovitskiy et al. [4] in 2020, the Vision Transformer (ViT) architecture splits an image into patches with position embeddings that are fed into the ViT model as a sequence of inputs – similar to text data. When trained on extremely large data sets (14M - 300M), Dosovitskiy et al. [4] concluded that the large scale ViT architecture was able to surpass the inductive bias found in the Convolutional Neural Network (CNN) architecture. However, the required scale, and in turn computational cost, of the models greatly differ between ViT and CNN architectures of similar performance. From Dosovitskiy et al. [4], the ViT-Large model requires 307M parameters and has equivalent performance to the ResNet152 architecure with 60M parameters, first introduced by He et al. [5] in 2015. Further, Dosovitskiy et al. [4] acknowledge that training on mid-sized datasets result in ViT models performing worse that current CNNs [5] of equivalent size due to the lack of inductive bias in the ViTs. This has led to extensive methods being explored in order to reduce the number of parameters and FLOPs required for these ViT models.

In this report we further investigate efficiency methods for ViT models in hopes of elucidating the potential of edge device deployment capabilities. We aim to achieve this goal by showcasing a potential for improved ViT efficiency without sacrificing performance competitive with popular CNN architectures. Generally, we aim to improve ViT efficiency through attempting to combine pruning methods targeting different architectural aspects. Specifically, we will implement two pruning methods: Block Pruning explored by Wang et al. [16], and Dynamic Pruning explored by Rao et al. [9].

2. Related Work

In general, pruning methods aim to remove low contributing parameters or architectural aspects of a model to increase efficiency in memory requirements and inference speeds. In the context of ViT [13], a small collection of different pruning methods have been proposed and tested since their introduction. One method is proposed by Zhu et al. [17] in which the dimension of the linear projections are pruned based on a specified threshold of learned features importance scores between $[0, 1]$. Although their methods will not be utilized, we find it crucial to acknowledge their

research as the suggested future work pointed towards potential methods of pruning attention heads and inputs, for which we will expand on in this paper.

2.1. Token Pruning

Similar to the original Transformer [14] architecture that receives a 1D sequence of embedded tokens to represent text, the ViT [4] architecture receives a sequence of flattened 2D patches with positional encoding (i.e. embedded tokens) to represent an image. In both architectures, the final prediction is often only influenced by a small subset of informative tokens and many unimportant tokens from the sequence can be removed with minimal performance deprecation. In language models, this has led researchers to implement pruning methods to adaptively remove uninformative tokens in the input sequences [6,8] and only perform inference on a subset of tokens with high contribution levels. Transferring these methods to ViT, Rao et al. [9] has proposed a method in which redundant tokens are pruned progressively and dynamically based on the input. Specifically, their framework utilizing a prediction module to estimate the importance score of each token given the current features – the module is added to different layers to prune redundant tokens hierarchically through a binary decision mask.

2.2. Block Pruning

Some well performing transformer pruning methods were first explored in the context of natural language processing (NLP) and later applied to the domain of computer vision. One such method was block pruning introduced by Lagunas et al. [7] in which model blocks of any size can be pruned. The block pruning method proposed by Lagunas et al. allows for greater flexibility than highly structured ridged pruning protocols. Their method was able to learn which full components could be removed – in the end dropping a high number of attention heads. This method was then implemented by Wang. et al [16] in the context of ViT in a 2022 paper, in which the authors explore a collection of efficiency frameworks for ViT mobile device deployment. Using a collection of pruning techniques (e.g. block pruning), Wang et al. uncover key insights into pruning choices that showcase promising outcome for improved ViT performance, such as pruning both FNN and attention heads.

Taking inspiration from the experimental design of Wang et al. we explored a novel combination of punning methods: block pruning [16] and dynamic pruning [9]. We implemented the methods in tandem with the goal of adding to the understanding of ViT efficiency and edge device deployment.

3. Methodology

3.1. Implementation of DeiT Baseline

The Data-Efficient image Transformer (DeiT) proposed by Touvron et al. [13] in 2020 was used as our baseline transformer model for experimentation. DeiT is a convolution free transformer architecture that was pre-trained on ImageNet-1k [2] at a resolution of 224x224. The DeiT baseline has 86.6 million learnable parameters, is able to achieve a top-1 accuracy of 81.8% on ImageNet, and requires 17.6 billion FLOPs (floating point operations per second) to pass a single input through the model. At inference, images were preprocessed to match the official DeiT implementation [13] with inputs being resized to 256×256, center-cropped at 224×224, and normalized with the official ImageNet mean and standard deviation values across all channels.

3.2. Pruning Method Implementation

As previously stated, we employ a combination of block [16] and dynamic [9] pruning in order to test the hypothesis that utilizing these methods in tandem will allow for greater FLOPs reduction with minimal drops in performance. Despite requesting access early into the term, an inability to be approved for the official ImageNet [2] data set led to an alternative format needing to be obtained through the Hugging Face API [11]. Data was stored in the form of a cache that was loaded into a Dataset object, but both block [16] and dynamic [9] pruning required a folder formatted version of ImageNet. This modification caused the need for much of the code from both implementations to be reformatted in order to work with the new data set. This is discussed in the following implementation details.

Block Pruning. The implementation of dynamic pruning has been adapted from the https://github.com/EdgeVisionTransformer repository, containing the official code from the research methods implemented by Wang et al. [16]. Due to changed data formatting, adapting their code for an alternative format led to restructuring the majority of their framework to account for the stored image cache. This work was still not functional at the end of the project. We also attempted to generate an ImageNet-formatted image folder by iterating through the Hugging Face cache and saving one by one image. Although the resulted image folder was compatible with Wang et al.'s code, the evaluation result was undesirable. In particular, we used their framework with the newly created image folder to evaluate the pre-train Deit-base from the Hugging Face and obtained 1% validation accuracy. As a results, at the time this report was written, we were unable to reproduce the results reported by Wang et al. [16].

Dynamic Pruning. The implementation of dynamic pruning has been adapted from the

https://github.com/DynamicViT repository, containing the official code from the research methods implemented by Rao et al. [9]. As discussed previously, the requirement of adapting their code for an alternative data format led to restructuring the majority of their code to work with the stored image cache. After successfully adapting their source code, training time became a new limiting factor in the ability to implement their methods. Multiple GPU training could not be accomplished, so single GPU training had to be utilized with training on the ImageNet-1k data set (1.28M images) resulting in 8 hours per epoch on an RTX 2080 Ti. Compared to Rao et al. who were able to train on a system with 8 GTX 1080 Ti GPU's, our computational resources limited our ability to reproduce their results.

To account for this deficiency in resources, a subset of 10% of the training data was utilized on a single GTX 1080 Ti GPU, resulting in approximately 45 minutes per epoch. It is important to disclose that this subset was not class balanced, or known to be inclusive of all classes, but was only a subset of the first 10% of the loaded ImageNet cache. This allowed us to reproduce results from Rao et al. [9] on a smaller scale, with only 128K images in the training set. Since vision transformer architectures require large amounts of data to be trained on in order to see similar performance to CNN architectures, we do not expect results to be competitive with either pruned ViT or full-scale CNN models. A range of values were tested for the *base rate* parameter, which determines the sparse ratio for pruned layers, in order to determine what level of pruning is required to achieve comparable throughput and performance.

Combined Methods. Despite the inability to recreate the results produced by Rao et al. [9], they provide their official model for the DeiT baseline architecture that was successfully pruned and has the highest performance. Our small-scale implementation was not utilized for the combination of methods due to low confidence in its ability to fully capture of benefits of pruning. The pruned DeiT baseline model from Rao et al. [9] was attempted to be utilized as the input model for block pruning [16], but no success was able to be achieved. We suspect this is due to the extra learnable parameters being in the model produced by Rao et al. not being supported in the implementation of block pruning. A very brief overview will be given in the Results section, but were not notable enough to warrant a full discussion.

3.3. Evaluating Model Performance

Following that the DeiT model is pre-trained on ImageNet-1k, the validation set for this data will be utilized for evaluating and comparing our pruned models to that of other reduced ViT implementations [9, 16]. The ImageNet [2] validation set contains 50,000 images across 1,000 classes, from which the top-1 accuracy will be recorded. In addition to this, both the number of parameters (millions) and FLOPs (giga) are measured and compared relative to the DeiT baseline for each pruned model. To define both metrics:

1. **Parameters (M)**: a measurement of all learnable parameters within the model architecture, inclusive of all layers.
2. **FLOPs (G)**: a measurement of the floating point operations per second required to pass a single $1\times3\times224\times224$ RGB image from input to output.

The code for measuring both of these metrics comes from an adaption of the work done by Rao et al. [9], where the fvcore library is implemented in tandem with an N-dimensional discrete Fourier transform on the input to count the number of FLOPs for a single throughput. Parameters are measured through an internal method within the model implementation library, and has been shown to be reliable relative to other papers measurements.

4. Results

From the original research papers [4, 9, 13, 16], Table 1 presents the highest performing models from each implementation. All pruned models utilize a base architecture of the DeiT-Base/16 model for consistency in comparison to our implementation. For all results discussed below, models were run on the Oregon State University high performance computing (HPC) cluster. No specific GPU was utilized for all training implementations, each method was allocated different resources based on the time of request. No results were acquired from combining the two pruning methods in tandem, so no discussion can be provided. Inference on the model resulted in accuracy below 1%, with no reductions in parameters or FLOPs being achieved. Similarly, no results could be obtained from applying only block pruning to the DeiT architecture, as adapting their code was beyond our capabilities given the time we had.

4.1. Dynamic Pruning

From our small-scaled implementation of the methods proposed by Rao et al. [9], the resulting models show promising results in the efficiency of dynamically pruning uninformative tokens. From Table 1, the models are denoted in the format DyVit-Sub/*base_rate* from our experiments, where *base_rate* was tested across 4 different values. The ratios are computed for a base rate value of b such that for layers 3, 6, and 9, the ratio of tokens that are kept are (b), $(b-0.2)$, and $(b-0.4)$, respectively. All models were trained across 5 epochs and batch size of 32 for consistency in our comparisons to one another.

As the base rate decreased, the number of tokens that are kept also decreases. As reflected in our results 1, decreasing this base rate from 0.7 to 0.4, by steps of 0.1, resulted in

Model	Params (M)	FLOPs (G)	Image Resolution	ImageNet Top-1 Acc (%)
ViT-Base [4]	86.6	17.6	224×224	77.9
DeiT-Base/16 [13]	86.6	17.6	224×224	81.8
EfficientFormer-L1 [16]	12.3 (-74.3)	2.6 (-15)	224×224	79.2 (-2.6)
EfficientFormer-L3 [16]	31.3 (-55.3)	7.8 (-9.8)	224×224	82.4 (+0.6)
EfficientFormer-L7 [16]	82.1 (-4.5)	20.4 (+2.8)	224×224	83.3 (+1.5)
DynamicViT-B/0.7 [9]	*N/A*	11.2 (-6.4)	384×384	81.3 (-0.5)
DyViT-Sub/0.7	89.4M (+2.8)	11.5 (-6.1)	224×224	80.1 (-1.7)
DyViT-Sub/0.6	89.4M (+2.8)	9.9 (-7.7)	224×224	78.9 (-2.9)
DyViT-Sub/0.5	89.4M (+2.8)	8.6 (-9)	224×224	74.4 (-7.4)
DyViT-Sub/0.4	89.4M (+2.8)	7.4 (-10.2)	224×224	63.1 (-18.7)

Table 1. Experimental results from model evaluation on ImageNet-1k validation set (50,000 images). Numbers in blue denote change between pruned and DeiT-Base/16 model in units of respective measurement. Results from Wang et al. [16] were calculate in GMACs and converted to GFLOPs, so estimates may differ slightly.

both FLOPs and accuracy decreasing. This is expected as we are masking an increased number of tokens as base rate increases, so only a small fraction of the input is actually being utilized at inference time. Comparing our models to the DeiT-Base/16 [13] performance, we were able to greatly reduce the flops with minimal increases to the number of parameters. However, the performance does begin to significantly drop as we further prune the model with lower base rates. Relative to the DynamicViT-B/0.7 from Rao et al. [9], our models seem to under perform as a whole. Our equivalent model, DyViT-Sub/0.7 1, evaluated to +0.3 GFLOPs and -1.2% accuracy on ImageNet, as we expected. This is due to a difference in the training data size and the number of epochs. Since we only utilized a subset of 10% on the training data, and 5 epochs rather than 30 as Rao et al. implemented, the amount of learning the model was able to achieve became a limiting factor in the ability to produce a comparable pruned ViT. Despite this, our experiments 1 do show promising results that even a small-scale pruning implementation can greatly reduce the throughput necessary for these large scale vision transformers.

An important note is that the number of parameters has actually increased as a result of applying dynamic pruning. We suspect this is a result of the increased parameters needed in order to learn which tokens can be removed. Rao et al. [9] do not include the measurement of parameters for their DynamicViT-B/0.7, but running their official release of the model through their inference code results in 89.4M parameters as well. In addition to this, our DyViT-Sub/0.4 1 model performed extremely low as a result of layer 9 being completed masked out, since the calculated sparse ratio from a base rate of 0.4 led to a ratio of 0 tokens begin kept. Although performance was decreased, we have been able to present that small-scale pruning can still see benefits of reducing throughput requirements and aid in the progress of ViT deployment on edge devices.

5. Conclusion

Although our results were inconclusive for determining if a combination of pruning methods [9, 16] could successfully further reduce the required throughput of complex vision transformer architectures, we still stand by our hypothesis that if implemented correctly, these methods in tandem could achieve comparable performance to equivalent CNN architectures. As stand alone methods, both block and dynamic pruning have been proven to reduce the parameters and FLOPs requirement for ViT architectures with minimal implications on the performance. To our surprise, even a small-scale pruning implementation with limited training data was able to successfully reduce the number of FLOPs with very low reductions in model performance. This does lead us to wonder if pruning can be accomplished with a small-scale version of the data set if pre-training is implemented with a much larger data set. This would be one idea for a future research direction for ViT pruning methods. Although we did not achieve the initial goal set forth by our project, we are still optimistic that future research into pruning methods will be able to close the gap between ViT and CNN performance for edge device deployment capabilities.

5.1. Limitations and Future Directions

Limited Training Data Size. As acknowledged by Dosovitskiy et al. [4], the ViT architecture requires extremely large data set sizes (14M - 300M) in order to have comparative results to current state-of-the-art CNN architectures. The original ViT architecture was pre-trained on both the JFT-300M [12] and ImageNet-21k [10] datasets, with 300M and 14.2M images respectively, in order to see comparable results to competitive CNN models. In our experiments, and in experiments of other researchers that

have implemented these pruning methods, the DeiT-Base model [13] is utilized and only pre-trained on ImageNet-1k [2], which contains 1.28M training instances. This insufficient amount of training data, although only being utilized for fine tuning the pruned models, can be a limiting factor in the true performance that these methods can produce. With larger training sets we believe that equivalent, and potentially superior, performance could be produced by these vision transformer models when compared to equivalent CNN's.

Model Scale/Performance Trade-off. When creating compact model architectures for edge device deployment, both the model scale and performance need to be taken into consideration. There is often a trade-off between reducing the required computational throughput and scale of the model architecture. This is a common limitation for ViT's, as there is often a high trade-off between performance and reducing the model size for mobile deployment. From mobile device deployment research by Abrahamyan et al. [1] on compact CNN architectures and Wadekar et al. [15] on hybrid CNN and ViT architectures, we examine two models: SkipblockNet [1] and MobileViTv3 [15]. When matching the number of GFLOPs between models, SkipblockNet-L achieves 77.1% accuracy on ImageNet-1k and requires 0.364 GFLOPs, while MobileViTv3-0.5 achieves 74.01% and requires 0.481 GFLOPs. Vision transformer based architectures require much higher computational throughput, and although they can be made equivalent to comparable CNN models, the trade-off in performance will continue to be a limiting factor in why they are currently not a competitive replacement for CNN's on edge device deployment.

References

[1] Lusine Abrahamyan, Valentin Ziatchin, Yiming Chen, and Nikos Deligiannis. Bias loss for mobile neural networks, 2021. 5

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2, 3, 5

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. 1

[4] 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. 1, 2, 3, 4

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 1

[6] Sehoon Kim, Sheng Shen, David Thorsley, Amir Gholami, Woosuk Kwon, Joseph Hassoun, and Kurt Keutzer. Learned token pruning for transformers, 2022. 2

[7] François Lagunas, Ella Charlaix, Victor Sanh, and Alexander Rush. Block pruning for faster transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10619–10629, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. 2

[8] Chonghan Lee, Md Fahim Faysal Khan, Rita Brugarolas Brufau, Ke Ding, and Vijaykrishnan Narayanan. Token and head adaptive transformers for efficient natural language processing. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4575–4584, Gyeongju, Republic of Korea, Oct. 2022. International Committee on Computational Linguistics. 2

[9] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification, 2021. 1, 2, 3, 4

[10] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses, 2021. 4

[11] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 2

[12] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era, 2017. 4

[13] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *CoRR*, abs/2012.12877, 2020. 1, 2, 3, 4, 5

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. 1, 2

[15] Shakti N. Wadekar and Abhishek Chaurasia. Mobilevitv3: Mobile-friendly vision transformer with simple and effective fusion of local, global and input features, 2022. 5

[16] Xudong Wang, Li Lyna Zhang, Yang Wang, and Mao Yang. Towards efficient vision transformer inference: A first study of transformers on mobile devices. In *Proceedings of the 23rd Annual International Workshop on Mobile Computing Systems and Applications*, HotMobile '22, page 1–7, New York, NY, USA, 2022. Association for Computing Machinery. 1, 2, 3, 4

[17] Mingjian Zhu, Yehui Tang, and Kai Han. Vision transformer pruning, 2021. 1