Multi-Scale And Token Mergence: Make Your ViT More Efficient

Zhe Bian, Zhe Wang, Wenqiang Han, Kangping Wang*
Jilin University
{bianzhe21,hanwq20}@mails.jlu.edu.cn, {wz2000,wangkp}@jlu.edu.cn

Abstract

Since its inception, Vision Transformer (ViT) has emerged as a prevalent model in the computer vision domain. Nonetheless, the multi-head self-attention (MHSA) mechanism in ViT is computationally expensive due to its calculation of relationships among all tokens. Although some techniques mitigate computational overhead by discarding tokens, this also results in the loss of potential information from those tokens. To tackle these issues, we propose a novel token pruning method that retains information from non-crucial tokens by merging them with more crucial tokens, thereby mitigating the impact of pruning on model performance. Crucial and non-crucial tokens are identified by their importance scores and merged based on similarity scores. Furthermore, multi-scale features are exploited to represent images, which are fused prior to token pruning to produce richer feature representations. Importantly, our method can be seamlessly integrated with various ViTs, enhancing their adaptability. Experimental evidence substantiates the efficacy of our approach in reducing the influence of token pruning on model performance. For instance, on the ImageNet dataset, it achieves a remarkable 33% reduction in computational costs while only incurring a 0.1% decrease in accuracy on DeiT-S.

1 Introduction

The transformer architecture [24] has introduced the ability to model global relationships, which was not found in previous convolution-based methods. This has led to impressive performance on a variety of computer vision tasks, including image classification [9, 22, 33], semantic segmentation [27, 23], object detection [3, 15], and image generation [11]. Furthermore, a multitude of supervised, unsupervised, and alternative training techniques [28, 7, 2] have been developed for these tasks. Nonetheless, the quadratic computational complexity of the Vision Transformer (ViT), which stems from dense long-range dependencies among image tokens, presents a considerable challenge in computational cost. In general, ViTs require more training iterations and larger datasets than convolutional neural networks (CNNs).

To address the computational burden of ViTs, researchers have developed various techniques to reduce internal feature information within the model. One effective strategy involves evaluating token importance and conducting selective pruning. DynamicViT [20] introduces a module for determining token importance, while EViT [16] leverages the class token to assess other tokens' significance. Evo-ViT [30] considers the global class token's importance and updates non-crucial tokens differently. AS-ViT [17] employs a learnable threshold to adaptively control the number of retained tokens, and AdaViT [19] introduces modules to evaluate the importance of block, head, and token, streamlining the model from multiple aspects. Although many of these methods directly discard non-crucial tokens, some fuse them based on their importance scores. However, it is important to note that the dropped information may be valuable for classification tasks, as argued in [26]. Therefore, it is essential to

*Corresponding author.

Preprint. Under review.

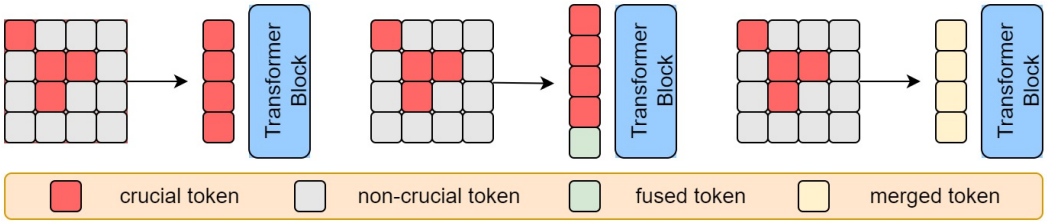

Figure 1: Different handling methods for non-crucial tokens, (a) Only crucial tokens are reserved and non-crucial tokens are discarded. (b) Crucial tokens are kept, and non-crucial tokens are fused into a new token. (c) Non-crucial and crucial tokens are merged.

adopt more effective processing techniques for non-crucial tokens, rather than merely discarding or fusing them. This approach preserves as much information as possible while mitigating the impact of token pruning on model accuracy.

Building on the previous findings, we present a novel approach to token pruning in this paper. Firstly, we assess the relevance of tokens in specific layers. Then, these tokens are classified into two groups: crucial and non-crucial, based on their relative important score. To preserve as much token information as possible while also reducing the number of tokens, we introduce an innovative Token Merging Module. This module calculates similarity metrics between crucial and non-crucial tokens, and merges them based on the metrics. Figure 1 illustrates the differences between our method with the previous works.

To improve the accuracy of feature representation, we draw inspiration from the groundbreaking methods introduced in CrossViT [5] and MPViT [14]. In our approach, we integrate multi-scale features within token embedding to achieve a richer feature set. To counteract potential precision loss during token pruning, we align and fuse multi-scale features, thereby enhancing the overall feature information. However, the incorporation of multi-scale features demands increased computations, which was tackled by an innovative technique. This strategy separates our approach from the previous works in CrossViT [5] and MPViT [14] too.

We introduce an innovative token pruning technique that achieves a 33% reduction in FLOPs and a 0.1% precision increase when applied to DeiT-S [22]. Notably, our approach is compatible with the most existing pruning methods. The experiments results show that as the proportion of pruned tokens rises, the module's performance improves significantly. For instance, our technique improves accuracy by 0.2% when applied to EViT [16] with a 0.7 keep rate, and by 0.7% with a 0.5 keep rate.

In summary, our contributions are twofold:

- We propose a novel token pruning strategy that effectively utilizes the original input information. Our method strikes a balance between model accuracy and speed, preserving as much original information as feasible during the pruning process.
- We improve model accuracy by incorporating multi-scale features and reducing computation prior to token pruning.

2 Related work

Vision Transformers. Transformer [24] has been successfully applied to natural language processing (NLP) tasks, achieving state-of-the-art results. Furthermore, ViT [9] has achieved impressive results by converting images into sequential tokens and processing them through the transformer architecture. Transformers recently have applied to computer vision (CV) domain, producing promising results in image classification [22], object detection [3], semantic segmentation [34], and other tasks [11]. Simultaneously, several high-performing backbone models have emerged. For example, DeiT [22] exploits knowledge distillation to effectively train ViT [9] on ImageNet alone, while Swin Transformer [18] incorporates local and sliding window operations to introduce inductive bias for ViT [9], consequently reducing its computational complexity. LV-ViT [12] enhances model accuracy by computing losses for all tokens.

Efficient ViTs. Unlike natural language, images often contain a substantial amount of redundant information. As a result, for transformer models with high computational demands, techniques for reducing the computational burden by pruning redundant information from images have become increasingly important. Various approaches for pruning redundant information from images have developed in recent years. One such method focuses on pruning solely the input token information. DynamicViT [20] assesses the importance of input tokens using a learnable module and determines the number of tokens to retain according to a predefined keep rate. EViT [16] and Evo-ViT [30] evaluate the importance of other tokens by considering the characteristics of the class token. The former approach merges non-crucial tokens into a new token, while the latter retains non-crucial tokens while employing rapid updates to maintain information flow integrity. A-ViT [31] dynamically adjusts the number of tokens according to the input's complexity, thereby controlling the model's computational complexity. AS-ViT [17] emphasizes the role of distinct heads and employs a learnable threshold to determine token retention. Another method involves pruning token, head, block, and other model components by analyzing redundancy from multiple perspectives. AdaViT [19] introduces learnable parameters to determine whether block, head, and token should be pruned. MIA-Former [32] introduces an MIA-Controller to decide if a block should be skipped; if not, a learnable module is employed to determine head and token pruning.

3 Method

3.1 Preliminaries

Drawing inspiration from the successful implementation of Transformer [24] in the natural language processing domain, Vision Transformer (ViT) [9] converts an image into a sequence of tokens, analogous to words in a sentence. It also incorporates a class token (CLS) to obtain an image representation. To differentiate between tokens at various positions, position embeddings are integrated. The resulting input is then fed into stacked transformer encoders, with the output features forming the foundation for downstream tasks. The ViT's core component, the transformer encoder, comprises multi-head self-attention (MHSA) and a feed-forward network (FFN). MHSA is employed to obtain the query, key, and value through a linear mapping of the input information. Subsequently, the query and key are multiplied to produce an attention map, which is ultimately multiplied by the value and outputted via a matrix mapping. For an input $X \in R^{N \times D}$ to ViT, the MHSA process can be mathematically expressed as follows:

$$\begin{aligned} &Q_i = X_i W_Q^i; K_i = X_i W_K^i; V_i = X_i W_V^i, \\ &\text{Attention}(Q_i, K_i, V_i) = \text{Softmax}(\frac{Q_i K_i^T}{\sqrt{d}}) V_i, \\ &\text{MHSA}(X) = W_O \text{concat}[\text{Attention}(Q_i, K_i, V_i)]_{i=1}^H, \end{aligned} \tag{1}$$

Let $i \in \{1, 2, \ldots, H\}$ denote the index of the i-th head in a multi-head attention mechanism. Let $X_i, Q_i, K_i, V_i \in R^{N \times d}$ be the input, query, key, and value matrices of the i-th head, where N is the number of tokens, D is the embedding dimension, and d is the dimension of the embedding of a single head.

$$\text{FFN}(X) = \text{Sigmoid}(\text{Linear}(\text{GeLU}(\text{Linear}(X)))). \tag{2}$$

3.1.1 Computation complexity

In the ViT architecture, the computational cost of the MHSA and FFN modules are given by $O(4ND^2 + 2N^2D)$ and $O(8ND^2)$, respectively. Therefore, the total computational cost of a transformer block can be estimated as $O(12ND^2 + 2N^2D)$. Token pruning methods can be used to reduce the number of tokens by a certain percentage, denoted by $\lambda\%$. As a result, such methods can effectively reduce the FLOPs of a transformer block by at least $\lambda\%$.

3.2 Token selection

In the context of image classification, ViT solely relies on the class token for classification-related information. Specifically, we employ the following formulation 3:

$$X_{CLS} = \text{Softmax}(\frac{Q_{CLS} K^T}{\sqrt{d}}) V. \tag{3}$$

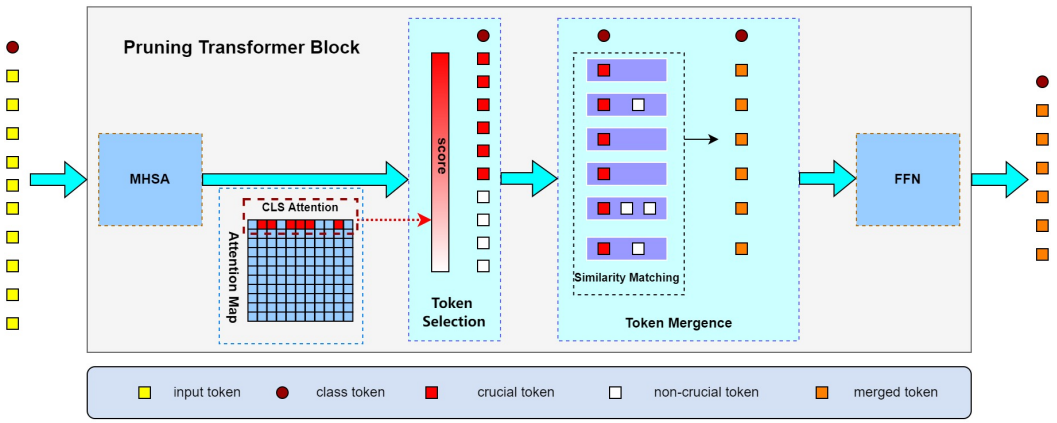

Figure 2: Token pruning within a single transformer encoder. Like EViT, We use the value of CLS attention as the score to judge the importance of each token. And identify the top-k crucial tokens. Then, we use cosine similarity of crucial and non-crucial tokens as similarity evaluation criterion and merge them.

where Q_{CLS} represents the query vector's class token. The class token X_{CLS} constitutes a linear combination of all token values, enabling its attention score to encapsulate the relative significance of other tokens for classification outcomes. In alignment with previous methods [20, 16, 30], we merely average dimensions within the head. During token selection, we preserve the first N tokens possessing the highest attention scores as crucial tokens, while the remaining tokens are classified as non-crucial tokens; the class token is inherently considered an crucial token by default.

3.3 Token mergence

Most token pruning techniques depend on a predefined criterion to decide whether to retain or prune a token. However, this reduction of input data for subsequent transformer layers can lead to significant accuracy losses when many tokens are pruned. To address this issue, our approach aims to prioritize crucial token information processing while concurrently managing non-crucial token information. We propose merging non-crucial token information into the most similar crucial token, as depicted in Figure 2.

In our method, tokens are classified into two categories: crucial and non-crucial. To prevent the pruning of non-crucial tokens from negatively impacting the model's accuracy, A merging stage is introduced for these two categories before pruning. Cosine similarity is employed as the primary metric for calculating the similarity between the categories, chosen for its straightforward implementation. We determine the similarity between each non-crucial token and the set of crucial tokens by computing the cosine similarity of the non-crucial token with each token in the crucial set. Subsequently, for each non-crucial token, we identify the crucial token with the highest cosine similarity as its most similar counterpart and merge the tokens. The weighted merging method is utilized to accommodate the varying relevance of tokens. The weights are determined by corresponding item in class token. Formula 4 represents the merged token.

$$m_j = (w_j i_j + \sum_{k \in S_j} w_k u_k)/(w_j + \sum_{k \in S_j} w_k). \tag{4}$$

Where w_j denotes the importance score of the crucial token i_j, w_k denotes the importance score of the non-crucial token u_k and S_j denotes the set of non-crucial tokens merged with the j-th crucial token, m_j denotes the j-th merged token. The size of merged tokens is calculated from keep rate.

3.4 Multi-scale features

To achieve higher accuracy, input tokens should present as many features as possible before token pruning. A multi-scale feature extraction approach is proposed for obtaining richer feature representations, with the complete model architecture illustrated in Figure 3. Specifically, we generate

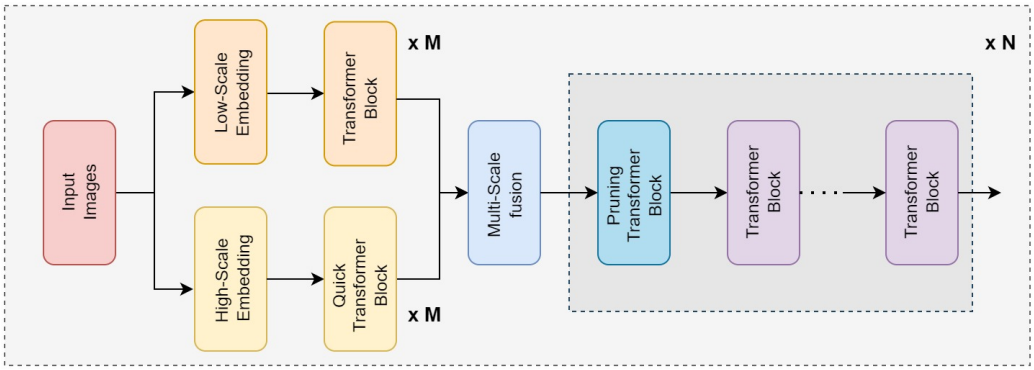

Figure 3: The overall architecture of the multi-scale transformer model.

two feature groups with identical embedding dimensions but varying token counts during feature embedding. The group with more tokens is termed as high-scale features, while the group with fewer tokens is termed low-scale features. We assign learnable position codes to each feature group. Before token pruning stage, we fuse the information from both feature groups to obtain more representative features. During the multi-scale feature fusion process, we upsample low-scale features using nearest interpolation and convolution methods to align them with high-scale features in terms of token count. To reduce parameter count and ensure effectiveness, we employ the LKA module [10] to transform the upsampled features. Finally, we add the upsampled low-scale features and high-scale features in the token dimension and integrate new location information into the resulting features using the PEG module [6]. The formulaic description of this process is as follows:

$$X = \mathrm{PEG}(X_h + \mathrm{LKA}(\mathrm{UP}(X_l))). \tag{5}$$

Where X_h and X_l respectively represent the high-scale features and the low-scale features, UP represents the nearest interpolation operation.

3.4.1 Reduce computation

Incorporating multi-scale features in visual transformers unavoidably increases computational demands. To mitigate this overhead while maintaining high accuracy, we propose a simple modification to the attention modules in the first blocks of the high-scale feature hierarchy. Specifically, we first downsample the high-scale features before processing them with the MHSA module. The downsampled tokens are then input into the MHSA module, and the resulting feature vectors are upsampled to the original scale, keeping the class token unchanged during the whole of the process. The formulaic description of this process is as follows:

$$X_h = X_h + \mathrm{UP}(\mathrm{MHSA}(\mathrm{DOWN}(X_h))). \tag{6}$$

Where X_h denotes the high-scale feature, DOWN represents a simple downsample operation, and UP refers to a nearest interpolation method. This update ensures that each block's multi-scale feature computational requirements are approximately equivalent to those of the original ViT.

4 Experiments

4.1 Implementation details

To maintain consistency, we trained all models in our experiment on the ImageNet-1k dataset [8], containing a training set of 12 million images and a test set of 50k images. To enhance performance, we incorporate Token Selection and Token Mergence modules into the 4^{th}, 7^{th}, and 10^{th} layers of DeiT-T, DeiT-S, and DeiT-B [22], as well as into the 5^{th}, 9^{th}, and 13^{th} layers of LV-ViT-S [12], adopting the same optimization strategy as the original paper. We employed high-scale features with 196 tokens and low-scale features with 49 tokens, each containing a class token. For a fair comparison with EViT [16], we gradually reduced the keep rate of attentive tokens from 1 to the target value utilizing a cosine schedule. Following the other methods' setting, we trained all models for 300

Table 1: Comparison with existing token pruning methods.

Model	Params(M)	FLOPs(G)	FLOPs↓(%)	Top-1(%)
DeiT-T [22]	5.7	1.3	0.0	72.2
DynamicViT [20]	5.9	0.9	30.8	71.2(-1.0)
SP-ViT [13]	5.7	0.9	30.8	72.1(-0.1)
Evo-ViT [30]	5.7	0.8	38.5	72.0(-0.2)
Ours-DeiT-T	5.8	0.8	38.5	**72.7(+0.5)**
DeiT-S [22]	22.1	4.6	0.0	79.8
ToMe [1]	22.1	2.7	41.3	79.4(-0.4)
DynamicViT [20]	22.8	2.9	37.0	79.3(-0.5)
A-ViT [31]	22.1	3.6	21.7	78.6(-1.2)
Evo-ViT [30]	22.1	3.0	34.8	79.4(-0.4)
EViT [16]	22.1	3.0	34.8	79.5(-0.3)
AS-ViT [17]	22.1	3.0	34.8	79.6(-0.2)
Ours-DeiT-S	22.2	3.1	32.6	**79.7(-0.1)**
DeiT-B [22]	86.6	17.5	0.0	81.8
DynamicViT [20]	-	11.2	36.0	81.3(-0.5)
Evo-ViT [30]	86.6	11.5	34.2	81.3(-0.5)
EViT [16]	86.6	11.5	34.2	81.3(-0.5)
AS-ViT [17]	88.6	11.2	36.0	81.4(-0.4)
Ours-DeiT-B	86.6	11.5	34.2	**81.5(-0.3)**

epochs on 8 NVIDIA RTX 3090 GPUs, and measured their throughput using a single NVIDIA RTX 3090 GPU with a 128 batch size.

4.2 Main results

4.2.1 Comparison with the-state-of-the-arts

As shown in Table 1, we provide a comparative analysis of our token pruning method against other state-of-the-art techniques on DeiT [22], reporting top-1 accuracy and FLOPs to assess performance. Our approach outperforms previous methods, delivering superior results while maintaining reasonable computational costs. Specifically, our method reduces the computational complexity of DeiT-T [22] by 35% while improving model accuracy by 0.5%. We also apply our method to LV-ViT [12], which is a deep-narrow architecture. As illustrated in Table 2, our approach enables LV-ViT [12] to achieve an optimal balance between accuracy and speed compared to other transformer models.

4.2.2 Comparison with existing methods on each keep rate

We evaluate the accuracy of our method and the other two [20, 16] at different keep rates, as depicted in Figure 5. The results show that our method outperforms the other methods at the same keep rates, with more significant improvements at lower keep rates. This is likely because our method merges the information of non-crucial tokens into crucial tokens, reducing information loss. This is in contrast to Dynamic-ViT [20], which directly discards non-crucial tokens, or EViT [16], which fuses them into a single token. Consequently, the lower the keep rate, the more tokens are merged and the greater the benefit of our method. Furthermore, we incorporate multi-scale features before token pruning, leading to improved performance compared to the baseline at high keep rates.

Table 2: Comparison with the state-of-the-art models.

Method	Params(M)	FLOPs(G)	Top-1(%)
ViT-B [9]	86.6	4.6	77.9
DeiT-S [22]	29.0	4.5	79.8
Swin-T [18]	50.0	4.6	81.3
T2T-ViT-14 [33]	21.5	4.8	81.5
CvT-21 [25]	31.5	7.1	82.5
DW-T [21]	30.0	5.2	82.0
Cross-ViT-S [5]	26.7	5.6	81.0
CoaT-Lite Small [29]	20.0	4.0	81.9
RegionViT-S [4]	30.3	5.3	82.6
LV-ViT-S [12]	26.2	6.6	83.3
DynamicViT-LV-S [20]	26.9	3.7	82.0
EViT-LV-S [16]	26.2	3.9	82.5
AS-LV-S [17]	26.2	3.9	82.6
Ours-LV-S	26.2	3.9	**82.8**

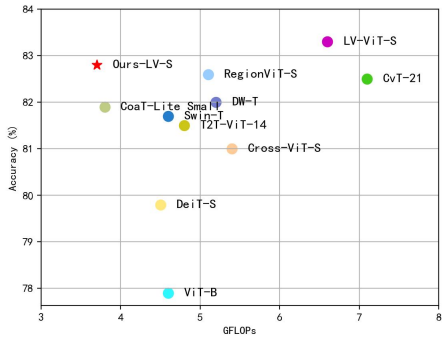

Figure 4: Comparison with the state-of-the-art models.

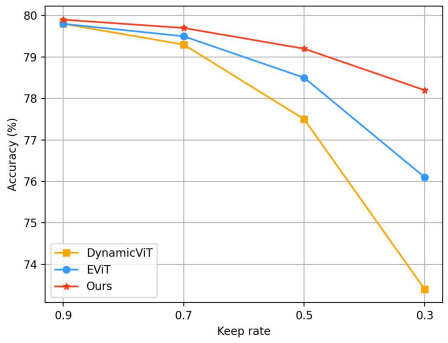

Figure 5: Comparisons with existing methods on each keep rate.

4.2.3 Visualization

Our method adopts a novel strategy to decrease computational costs by identifying and preserving crucial tokens while merging non-crucial ones to reduce redundancy. To evaluate the effectiveness of our approach, we present visualizations of each stage in Figure 6. The results show that our method successfully distinguishes crucial tokens from non-crucial ones, by concentrating tokens in positions that significantly impact classification, while merging background elements. For instance, in the image classified as a panda, our method assigns more tokens to the panda after each pruning layer, while continuously merging the background. This evidence depicts why the method is effective.

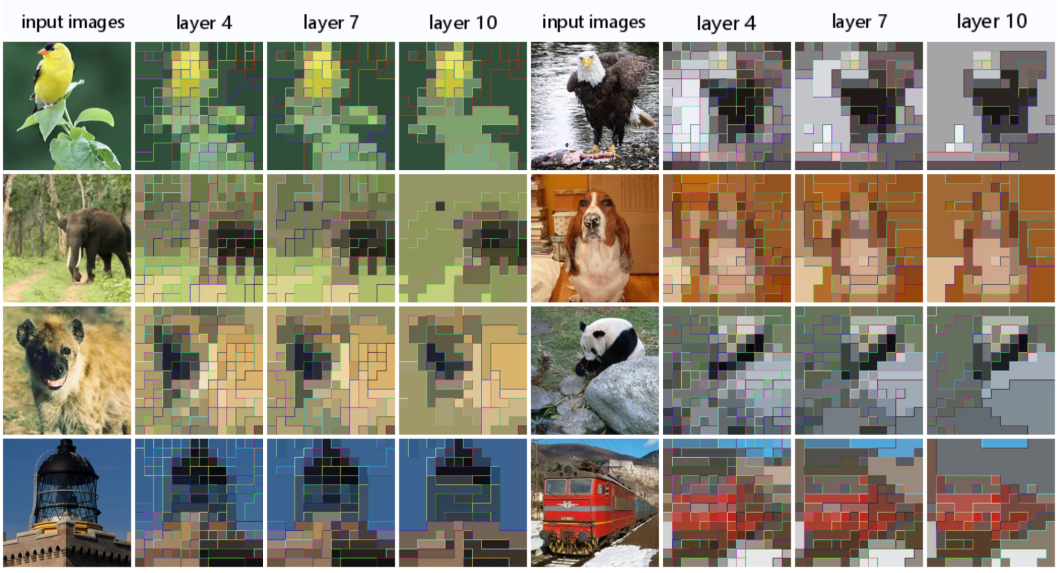

Figure 6: Visualization of the token pruning process on DeiT-S at a keep rate of 0.5. Using a color-coding scheme, tokens that have been merged together are represented with the same color, allowing for a clear and intuitive representation of the pruning process.

Table 3: The effectiveness of each module on EViT-DeiT-S with different keep rates.

Method	Top-1 (%)	FLOPs(G)
	DeiT-S/η=0.5	
baseline	78.5	2.3
+TM	79.1	2.3
+MF	79.2	2.4
	DeiT-S/η=0.7	
baseline	79.5	3.0
+TM	79.6	3.0
+MF	79.7	3.1

Table 4: Different similarity calculation methods on DeiT-S.

Method	Top-1 (%)	Throughput (img/s)
Random	78.8	2137
Attention Map	79.0	2195
L1 Distance	79.1	2033
L2 Distance	79.1	2181
Ours	79.2	2201

4.3 Ablation Analysis

4.3.1 Effectiveness of each module

We have integrated our approach into EViT [16] to evaluate the effectiveness of individual modules. The experimental results are shown in Table 3, where TM denotes token mergence and MF represents multi-scale feature fusion. Our analysis shows that incorporating TM significantly improves the baseline accuracy by preserving non-crucial token feature information. Furthermore, the inclusion of MF results in more comprehensive feature extraction and improvement in baseline accuracy.

4.3.2 Different similarity calculation methods

As shown in Table 4, we have compared several techniques for computing the similarity between crucial and non-crucial tokens on DeiT-S [22] when the keep rate is 0.5. These methods include: i) merging crucial and non-crucial tokens randomly; ii) assessing similarity using the cross scores of crucial and non-crucial tokens in the attention matrix; iii) employing Manhattan distance for similarity calculation; iv) employing Euclidean distance for similarity computation; and v) employing cosine similarity. Our observations indicate that, given similar throughput, our method typically attains a 0.1% enhancement in accuracy relative to alternative similarity calculation techniques. This implies that cosine similarity serves as an effective metric for determining similarity between crucial and

non-crucial tokens. Moreover, compared to other methods, cosine similarity has the relatively simple computation. In summary, our findings endorse the application of cosine similarity for computing similarity between crucial and non-crucial tokens, owing to its precision and computational efficiency.

5 Conclusion

In this study, we introduce an innovative token pruning module that is designed to reduce the impact on model accuracy during token pruning. By selectively identifying crucial tokens and merging non-crucial ones, we maintain the accuracy caused by token pruning. The integration of multi-scale features further increases model accuracy. The experiments results from DeiT [22] support the effectiveness of our proposed module. Moreover, our technique can be combined with the majority of existing token pruning methods to enhance their accuracy with almost no computational overhead. Our approach achieves an good balance between accuracy and speed. We envision our method being valuable for downstream tasks such as target detection, semantic segmentation, and beyond classification tasks.

References

[1] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *CoRR*, abs/2210.09461, 2022.

[2] Zhaowei Cai, Avinash Ravichandran, Paolo Favaro, Manchen Wang, Davide Modolo, Rahul Bhotika, Zhuowen Tu, and Stefano Soatto. Semi-supervised vision transformers at scale. In *NeurIPS*, 2022.

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV (1)*, volume 12346 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2020.

[4] Chun-Fu Chen, Rameswar Panda, and Quanfu Fan. Regionvit: Regional-to-local attention for vision transformers. In *ICLR*. OpenReview.net, 2022.

[5] Chun-Fu (Richard) Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *ICCV*, pages 347–356. IEEE, 2021.

[6] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021.

[7] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. UP-DETR: unsupervised pre-training for object detection with transformers. In *CVPR*, pages 1601–1610. Computer Vision Foundation / IEEE, 2021.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. IEEE Computer Society, 2009.

[9] 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. In *ICLR*, 2021.

[10] Meng-Hao Guo, Chengze Lu, Zheng-Ning Liu, Ming-Ming Cheng, and Shimin Hu. Visual attention network. *CoRR*, abs/2202.09741, 2022.

[11] Yifan Jiang, Shiyu Chang, and Zhangyang Wang. Transgan: Two pure transformers can make one strong gan, and that can scale up. In *NeurIPS*, pages 14745–14758, 2021.

[12] Zihang Jiang, Qibin Hou, Li Yuan, Daquan Zhou, Yujun Shi, Xiaojie Jin, Anran Wang, and Jiashi Feng. All tokens matter: Token labeling for training better vision transformers. In *NeurIPS*, pages 18590–18602, 2021.

[13] Zhenglun Kong, Peiyan Dong, Xiaolong Ma, Xin Meng, Wei Niu, Mengshu Sun, Xuan Shen, Geng Yuan, Bin Ren, Hao Tang, Minghai Qin, and Yanzhi Wang. Spvit: Enabling faster vision transformers via latency-aware soft token pruning. In *ECCV (11)*, volume 13671 of *Lecture Notes in Computer Science*, pages 620–640. Springer, 2022.

[14] Youngwan Lee, Jonghee Kim, Jeffrey Willette, and Sung Ju Hwang. Mpvit: Multi-path vision transformer for dense prediction. In *CVPR*, pages 7277–7286. IEEE, 2022.

[15] Tianyang Li, Jian Wang, and Tibing Zhang. L-DETR: A light-weight detector for end-to-end object detection with transformers. *IEEE Access*, 10:105685–105692, 2022.

[16] Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. *CoRR*, abs/2202.07800, 2022.

[17] Xiangcheng Liu, Tianyi Wu, and Guodong Guo. Adaptive sparse vit: Towards learnable adaptive token pruning by fully exploiting self-attention. *CoRR*, abs/2209.13802, 2022.

[18] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 9992–10002. IEEE, 2021.

[19] Lingchen Meng, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser-Nam Lim. Adavit: Adaptive vision transformers for efficient image recognition. In *CVPR*, pages 12299–12308. IEEE, 2022.

[20] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In *NeurIPS*, pages 13937–13949, 2021.

[21] Pengzhen Ren, Changlin Li, Guangrun Wang, Yun Xiao, Qing Du, Xiaodan Liang, and Xiaojun Chang. Beyond fixation: Dynamic window visual transformer. In *CVPR*, pages 11977–11987. IEEE, 2022.

[22] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 10347–10357. PMLR, 2021.

[23] Jeya Maria Jose Valanarasu, Poojan Oza, Ilker Hacihaliloglu, and Vishal M. Patel. Medical transformer: Gated axial-attention for medical image segmentation. In *MICCAI (1)*, volume 12901 of *Lecture Notes in Computer Science*, pages 36–46. Springer, 2021.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[25] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *ICCV*, pages 22–31. IEEE, 2021.

[26] Kai Yuanqing Xiao, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. Noise or signal: The role of image backgrounds in object recognition. In *ICLR*. OpenReview.net, 2021.

[27] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. In *NeurIPS*, pages 12077–12090, 2021.

[28] Tongkun Xu, Weihua Chen, Pichao Wang, Fan Wang, Hao Li, and Rong Jin. Cdtrans: Cross-domain transformer for unsupervised domain adaptation. In *ICLR*. OpenReview.net, 2022.

[29] Weijian Xu, Yifan Xu, Tyler A. Chang, and Zhuowen Tu. Co-scale conv-attentional image transformers. In *ICCV*, pages 9961–9970. IEEE, 2021.

[30] Yifan Xu, Zhijie Zhang, Mengdan Zhang, Kekai Sheng, Ke Li, Weiming Dong, Liqing Zhang, Changsheng Xu, and Xing Sun. Evo-vit: Slow-fast token evolution for dynamic vision transformer. In *AAAI*, pages 2964–2972. AAAI Press, 2022.

[31] Hongxu Yin, Arash Vahdat, Jose M. Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-vit: Adaptive tokens for efficient vision transformer. In *CVPR*, pages 10799–10808. IEEE, 2022.

[32] Zhongzhi Yu, Yonggan Fu, Sicheng Li, Chaojian Li, and Yingyan Lin. Mia-former: Efficient and robust vision transformers via multi-grained input-adaptation. In *AAAI*, pages 8962–8970. AAAI Press, 2022.

[33] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis E. H. Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *ICCV*, pages 538–547. IEEE, 2021.

[34] Bowen Zhang, Zhi Tian, Quan Tang, Xiangxiang Chu, Xiaolin Wei, Chunhua Shen, and Yifan Liu. Segvit: Semantic segmentation with plain vision transformers. In *NeurIPS*, 2022.