Removing Rows and Columns of Tokens in Vision Transformer enables Faster Dense Prediction without Retraining

Diwei Su[1], Cheng Fei[1], and Jianxu Luo[1](✉)

East China University of Science and Technology, Shanghai 200237, China
dvsu@mail.ecust.edu.cn, cfei_jarvis@163.com, jxluo@ecust.edu.cn

Abstract. In recent years, vision transformers based on self-attention mechanisms have demonstrated remarkable abilities in various tasks such as natural language processing, computer vision (CV), and multimodal applications. However, due to the high computational costs and the structural nature of images, the application of transformers to CV tasks faces challenges, particularly when handling ultra-high-resolution images. Recently, several token reduction methods have been proposed to improve the computational efficiency of transformers by reducing the number of tokens without the need for retraining. These methods primarily involve fusion based on matching or clustering. The former exhibits faster speed but suffers more accuracy loss compared to the latter. In this work, we propose a simple matching-based fusion method called Token Adapter, which achieves comparable accuracy to the clustering-based fusion method with faster speed and demonstrates higher potential in terms of robustness. Our method was applied to Segmenter, MaskDINO and SWAG, exhibiting promising performance on four tasks, including semantic segmentation, instance segmentation, panoptic segmentation, and image classification. Specifically, our method can be applied to Segmenter on ADE20k, providing 41% frames per second (FPS) acceleration while maintaining full performance without retraining or fine-tuning off-the-shelf weights. Our code will be released at `https://github.com/MilknoCandy/Token-Adapter`.

Keywords: Vision transformer · Token reduction · Dense prediction · Efficient transformer

1 Introduction

Over the past few years, the computer vision field has been significantly advanced by the emergence of the Transformer architecture [31]. Vision Transformers [9] with global attention mechanism achieve state-of-the-art results on a wide range of vision tasks. To obtain better performance, researchers have scaled up the number of parameters of vision transformers to billions [15] through techniques such as self-supervised pre-training [14, 29] methods and adding visual prior knowledge [5, 22], etc. However, scaling up transformers brings performance improvements as well as huge computational costs due to the quadratic complexity

(a) (b) (c)

Fig. 1: Visualizations of random pruning results from different methods on ADE20K. We randomly prune the same number of tokens using different methods. (a) Input Images. (b) Randomly remove tokens (c) Randomly remove specific rows and columns.

of self-attention mechanism. Vision transformers are extremely inefficient when processing high-resolution images. Designing efficient vision transformers has become a prominent research hotspot.

Some researchers have reduced the computational complexity to linear by optimizing the computation of self-attention mechanism [24, 30, 39, 42], while others are focusing on reducing the number of tokens to increase the inference efficiency of vision transformers [3, 10, 32, 44]. Despite significantly improving the efficiency of the vision transformer, most of these methods necessitate the retraining or fine-tuning of the optimized Transformer, demanding substantial computational resources.

There are also methods that do not require fine-tuning or retraining, such as Expediting [19] and ToMe (Token Merging) [2]. Expediting integrates local information to fuse and reconstruct tokens using a clustering algorithm, and ToMe fuses the corresponding tokens based on a bipartite matching algorithm. Previous methods either selected the most representative tokens based on their importance scores or employed clustering algorithms to obtain more representative tokens. The former approach involves calculating importance scores for all tokens, introducing significant computation, and necessitating operations such as padding to ensure uniform dimensions across different batches for batch processing. Although the latter method effectively compresses token sequences, parameter tuning for clustering algorithms poses a challenge, and the iterative calculation approach incurs additional computational costs.

In this work, we present Token Adapter to merge rows and columns of tokens rather than merge tokens sparsely for the following three reasons:

1. **Information Redundancy.** In vision tasks, multi-scale features naturally provide more information than single-scale features, resulting in spatial information redundancy in multi-scale image features.
2. **Image Structure.** As image data is derived from sampling real-world optical information, semantic information is contained within consecutive pixel

blocks rather than individual pixels. With the same number of missing pixels, removing entire rows or columns of pixels from an image could be less detrimental to the existing semantic information than the pruning partial pixels approach.

3. **Computational Efficiency.** To speed up vision transformers, one simple idea is that using fewer tokens can make them more efficient. Wang et al. [32] has observed that simple images can be accurately predicted with only a few tokens. This observation highlights the inconsistency in the number of tokens required to make accurate predictions for images of varying content complexity.

Fig. 1 shows the difference between our approach and the previous methods, i.e., the change in token selection strategy. It allows our method to compress token sequences efficiently and handle batch data directly without additional operations. Our experiments show that Token Adapter outperforms Expediting and ToMe on multiple segmentation datasets. To summarize, our main contributions can be outlined as follows: (i) We propose a simple yet powerful module that injects normal tokens into representative tokens while preserving unique tokens associated with dense prediction tasks; (ii) Our token reduction method preserves relative positional information among tokens, providing ViT-like models with substantial acceleration and slight accuracy loss in dense prediction tasks; (iii) Without the need for retraining or fine-tuning, our approach achieves performance comparable to clustering-based methods by employing matching-based techniques, demonstrating even better stability and speed.

2 Related Work

Efficient Vision Transformer. Some studies have focused on adapting the internal configuration of transformer models to improve their efficiency. These efforts include designing the architecture of the transformer [4, 18, 30, 35], modifying attention mechanisms [13, 21, 34, 38], approximating attention [6, 11, 12, 26], etc.

Token Reduction. Benefiting from the ability of Transformers to handle variable-length sequences, some recent work has attempted to optimize intermediate representations within transformers. Previous approaches could be categorized by their purpose into two groups: Pruning [16, 33, 36, 37] and Fusion [1, 23, 40, 41]. From a methodological perspective, these approaches can be divided into two categories: *recognition*-based and *non-recognition*-based methods. Recognition-based methods categorize all tokens into two groups, vital and minor, performing pruning or fusion operations on them separately based on specific rules. On the other hand, non-recognition-based methods employ clustering algorithms to cluster all tokens, reducing the number of tokens while preserving as much semantic information as possible.

Our Approach. We decompose token compression into row and column compression, providing a more efficient token reduction method. Additionally, we found that some rows or columns in tokens are unique and cannot be pruned

or merged. In our experiments, retaining these rows or columns yields better results than not retaining them.

3 Methodology

3.1 Overview

Designing for dense prediction tasks, our token adapter comprises two simple match-based approaches: the token injector and the token ejector. Token adapter utilize a subset of tokens for the purpose of representing another subset of tokens while maintaining the integrity of the original token set.

Pipeline. As shown in Fig. 2, two modules of the token adapter are inserted in the middle of consecutive transformer blocks. Given a feature map $X \in \mathbb{R}^{H \times W \times C}$, we compress rows and columns with reduction ratios r_h and r_w respectively, reducing the number of tokens by n_r, where $n_r = H \times W \times (r_h + r_w - r_h \times r_w)$. Here, r_h and r_w are settable parameters. Finally, we rearrange the compressed tokens to rebuild the original token sequence. The details are introduced in the next section.

3.2 Token Adapter

Token Injector. For sequence input $X_s \in \mathbb{R}^{B \times N \times C}$, we first reshape it to $X \in \mathbb{R}^{B \times H \times W \times C}$, where $N = H \times W$ represents the number of tokens.To identify which *rows* and *columns* to merge, importance scores are computed for each row and column. In images, areas with sharp pixel value changes usually correspond to inter-object or intra-object boundaries. Moreover, areas with larger pixel values compared to their surroundings are typically more significant.

In our experiments, we perform two steps of dimension reduction on features to obtain their importance scores. For the input feature $X \in \mathbb{R}^{B \times H \times W \times C}$, we first calculate the maximum value of each token vector (you can also calculate its sum value or range value instead) , thereby compressing the features into $X_{hw} \in \mathbb{R}^{B \times H \times W}$. Subsequently, the same procedure is applied to each row and column, yielding $S_h \in \mathbb{R}^{B \times H}$ and $S_w \in \mathbb{R}^{B \times W}$ as the importance scores. Please note that we employed a straightforward method to calculate the importance scores, yet we believe that a better importance assessment algorithm could yield better results.

Based on these scores, tokens are categorized into three groups: *representative* tokens, *unique* tokens, and *normal* tokens. Normal tokens are injected into representative tokens, while unique tokens remain intact. The compressed tokens are rearranged in their previous positions. A brief flow is delineated below:

1. Calculate the importance scores for each row and column.
2. Group all tokens into: *representative* tokens, *unique* tokens and *normal* tokens by importance scores.
3. Match the *most similar* representative token for each normal tokens.
4. Merge normal tokens with corresponding representative tokens.

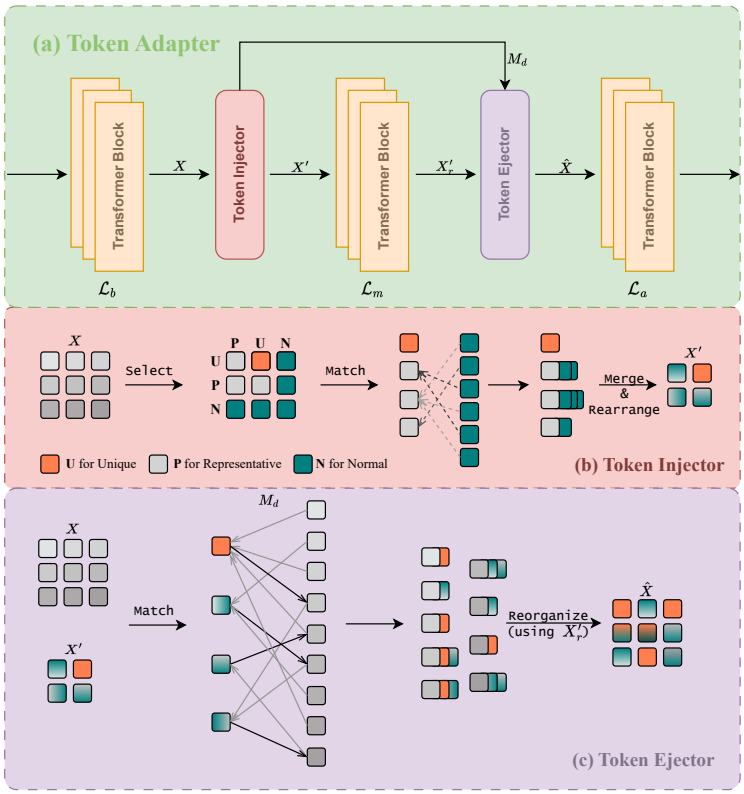


Fig. 2: Token Adapter. (a) We use Token Injector module and Token ejector module to wrap $\mathcal{L}_m$ Transformer blocks. (b) Token Injector mitigates the computational overhead of subsequent Transformer blocks by injecting normal tokens into representative tokens. (c) By leveraging token-to-token relationships, we represent the original sequence with compressed tokens.

5. Rearrange new representative tokens and unique tokens in their original order.

Within an image, consecutive pixel blocks form regions containing semantic information, and these associated semantic regions collectively constitute complete image objects. Merging entire rows or columns preserves the continuity of the original semantic regions while maintaining their relative positions, as depicted in Fig. 3. Our key thought involves representing each object through a refined set of representative and unique tokens. The results presented in Fig. 6 indicate that the retention of partial unique tokens generates better prediction accuracy.

Token Ejector. After compression, each representative token contains a blend of information from the original representative tokens and the merged normal tokens. Therefore, we exploit the correlation (e.g., the distance matrix

(a) (b) (c) (d)

Fig. 3: Token Selection Visualizations. Results obtained through the removal of specific rows and columns from the ADE20K validation dataset using Segmenter+ViT-L/16 equipped with Token Adapter ($r_h = r_w = 0.3$). (a) Input Images. (b) Remove specific rows and columns. (c) Input images in which each patch is filled with the mean color of its respective region. (d) Remove specific rows and columns from (c).

M_d) between the reduced sequence and the original sequence to restore the original length. Illustrated in Fig. 2, the primary steps are as follows:

1. Compute the distance matrix $M_d \in \mathbb{R}^{N \times N_r}$ between $X \in \mathbb{R}^{N \times C}$ and $X' \in \mathbb{R}^{N_r \times C}$, with $N_r = N - n_r$ denoting the length of reduced sequence.
2. Perform token matching between tokens in X and X' according to M_d.
3. Reconstruct the original sequence $\hat{X}$ by aggregating tokens in X'_r in accordance with the alignment results.

Note that the distance matrix is computed from the original sequence X_s and the reduced sequence X' that has not been refined by the subsequent transformer blocks. By following this approach, the reduced sequence can be efficiently reverted to its initial length.

4 Experiments

4.1 Setup

Dataset and evaluation metrics. We present task-specific metrics, including the mean Intersection-over-Union (mIoU) and average precision (AP), etc. Furthermore, we compare our method with others by analyzing model inference speed and floating-point operations (FLOPs). The datasets utilized for evaluation include ADE20K [43], PASCAL-Context [25], CityScapes [7], COCO [20], and ImageNet-1k [8], covering semantic segmentation, instance segmentation, panotic segmentation, object detection, and image classification tasks.

Implementation details. To assess the generalization of our method, we equipped two models with Token Adapter and conducted a series of experiments on four well-known datasets for dense prediction tasks. The proposed module is inserted into two classic models off-the-shelf with no re-training or fine-tuning required. All experiments are performed on a single NVIDIA GeForce RTX3090 GPU using Pytorch framework. Further details are presented in the following sections. Our code implementation is based on the open-source code of Expediting and ToMe.

4.2 Main Results

In this section, we delve into a comparative analysis of the proposed approach with other token reduction methods that do not require retraining and are applicable to dense prediction tasks. Since the number of GFLOPs does not provide a direct measure of model inference efficiency, our primary concern lies in accuracy metrics for dense prediction tasks and corresponding metrics for evaluating inference speed.

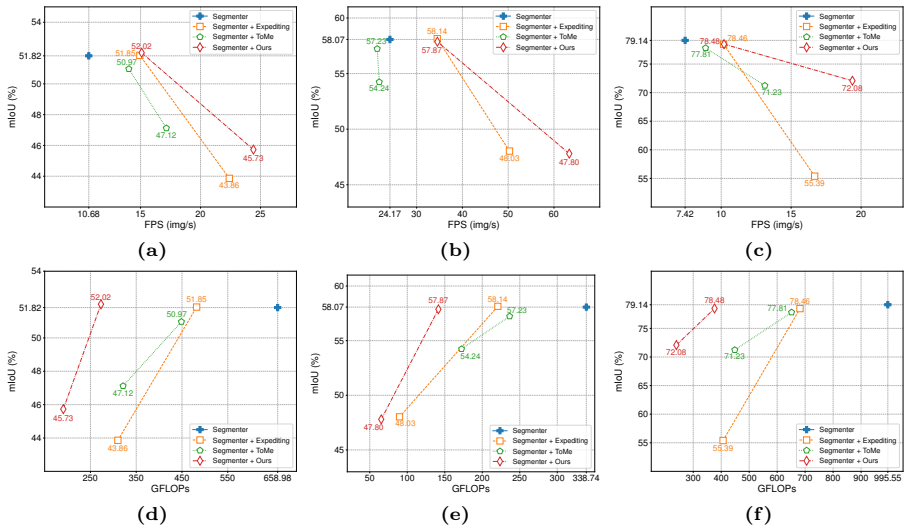


Fig. 4: Comparisons on Semantic Segmentation Tasks. A comprehensive comparison of our approach with Expediting and ToMe on ADE20K (1-st column), PACAL-Context (2-ed column), and Cityscapes (3-rd column) using Segmenter + ViT-L/16.

Comparisons on Semantic Segmentation Task. We use the ViT-like segmentation network model Segmenter [28] as our baseline for three benchmarks, including ADE20K, PASCAL-Context, and Cityscapes, reporting semantic segmentation results. We compare our approach with Expediting[1] [19] and

[1] `https://github.com/Expedit-LargeScale-Vision-Transformer`

Table 1: Comparisons on Semantic Segmentation Tasks. Quantitative comparisons of our method with other token reduction approaches that do not require retraining on semantic segmentation tasks.

Dataset	Method	mIoU(%)	FPS(img/s)	FLOPs(G)
ADE20K	Segmenter + ViT-L/16	51.82	10.68	658.98
	+ Token Merging	47.12	17.15	321.29
		50.97	14.02	448.7
	+ Expediting	43.86	22.4	309.96
		51.85	14.94	481.9
	+ Token Adapter (Ours)	45.73	**24.41**	190.74
		52.02	15.09	273.0
PASCAL-Context	Segmenter + ViT-L/16	58.07	24.17	338.74
	+ Token Merging	54.24	21.84	172.38
		57.23	21.39	236.5
	+ Expediting	48.03	50.28	89.66
		58.14	34.50	220.76
	+ Token Adapter (Ours)	47.80	**63.34**	65.29
		57.87	34.53	141.5
Cityscapes	Segmenter + ViT-L/16	79.14	7.42	995.55
	+ Token Merging	71.23	13.12	448.5
		77.81	8.88	651.5
	+ Expediting	55.39	16.68	406.69
		78.46	10.17	681.57
	+ Token Adapter (Ours)	72.08	**19.38**	239.65
		78.48	10.2	376.16

ToMe[2] [2], which are clustering-based and matching-based methods for token reduction that do not require retraining, respectively. The results are reported in Tab. 1. The two lines for each method in Tab. 1 represent the best result and the maximum speedup that can be obtained with the same token compression settings, respectively.

To ensure fairness and comprehensive performance comparisons among different methods, we fine-tuned the hyper-parameters of each method starting from the default settings mentioned in their papers. The results are depicted in Fig. 4. Under maximum acceleration, Token Adapter exhibits superior stability to Expediting (+6.11% mIoU in average) and achieves higher velocity compared to ToMe (+93.33% fps in average). Refer to our supplementary material for detailed parameter settings.

Comparisons on Instance/Panoptic Segmentation Task. To further explore the generalization capabilities of our method across various Transformer frameworks, we have selected the latest framework, MaskDINO [17]. It utilized

[2] `https://github.com/facebookresearch/ToMe`

Swin Transformer as its backbone network and has exhibited remarkable performance in tasks like instance segmentation and panoptic segmentation on COCO. We employed MaskDINO as our baseline model to compare the overall performance of our approach with Expediting on the COCO dataset, as presented in Tab. 2.

Table 2: Comparisons on Instance/Panoptic Segmentation Tasks. Inserting both the proposed algorithm and Expediting at different stages of MaskDINO + Swin-L, we compare their comprehensive performance on the COCO dataset.

Method	Stage	PQ(%)	Mask AP(%)	FPS(img/s)	FLOPs(G)
MaskDINO + Swin-L	-	58.3	52.3	6.11	936.86
+ Expediting	1	30.7	22.67	6.05	920.88
	2	33.1	23.56	6.04	918.68
	3	57.98	**51.91**	6.88	810.94
	4	58.29	52.23	6.14	917.03
+ Token Adapter (ours)	1	**55.91**	**49.23**	6.31	915.74
	2	**56.03**	**49.05**	6.22	916.11
	3	57.57	51.42	**6.96**	809.66
	4	**58.42**	**52.34**	6.19	916.38

Due to the window attention mechanism segmenting all tokens into windows, we reduce the tokens within each window instead of the entire token sequence. To ensure a fair comparison, we gradually increase $\mathcal{L}_b$ starting from the setting where Expediting balances efficiency and accuracy, to showcase the impact of applying this method at different stages of MaskDINO.

By observing Tab. 2, it is evident that our approach is more stable than Expediting. This could be attributed to the fact that we directly combine and reconstruct tokens through matching, whereas Expediting relies on clustering, which is sensitive to hyper-parameters for different scenarios. On top of that, with the integration of token adapter into the fourth stage of the backbone network of MaskDINO, we can observe a slight enhancement in performance (+0.1% PQ).

Comparisons on Image Classification Task. We applied three methods to SWAG[3] [27] for testing on the ImageNet-1k dataset. The results are shown in Tab. 3, where † indicates that the same fusion method is applied as Token Merging, i.e., layer-by-layer compression, and 32 indicates merging one row per layer. 'Size' indicates the resolution of the feature map. 'r' denotes the number of tokens compressed per layer.

Visualization of the token injector. We visualize the token injection process in Fig. 5. Initially, we divide all tokens into three groups: representative

[3] `https://github.com/facebookresearch/SWAG`

Table 3: Comparisons on ImageNet1k validation dataset

Method	Size	r	Acc(%)	FPS(img/s)	FLOPs(G)
SWAG + ViT-L/16	32×32	-	88.07	22.92	364.77
+ Token Adapter	23×23	-	87.43	32.28	270.65
+ Expediting	23×23	-	87.8	29.21	275.79
+ Token Adapter†	-	32	87.87	36.49	224.02
+ Token Merging	-	32	87.80	38.58	218.18
+ Token Adapter†	-	64	85.34	66.04	130.36
+ Token Merging	-	64	67.56	79.99	116.40

tokens, unique tokens, and normal tokens. Subsequently, we inject the normal tokens into the representative tokens. In the third row of Fig. 5, we use different colored borders to represent the matching results of normal tokens and representative tokens. The last row in the figure represents the results of token injection. By observing this, we can notice that token injection effectively highlights the crucial pixels in the image while preserving the original semantic information. More results and hyper-parameter settings can be found in the supplementary material.

4.3 Ablation Studies

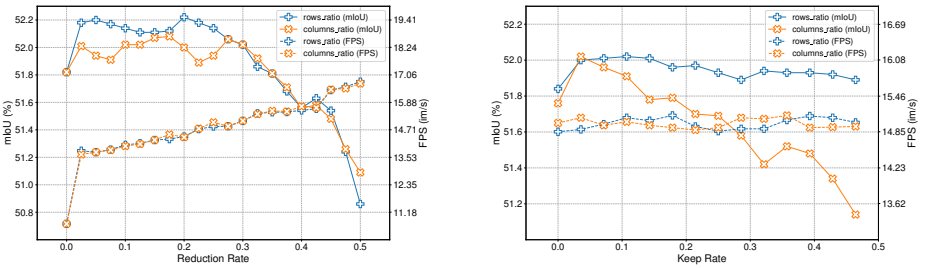
We performed ablation experiments for each critical choice within our approach on ADE20K semantic segmentation task. The results can be found in Tab. 4. By default, we employed the official weights provided by Segmenter[4] as a baseline model (Segmenter + ViT-L/16, mIoU: 51.82%, FPS: 10.68 img/s). The table cells marked in red and purple respectively denote the final choices for the token injector and token ejector.

Token Reduction Strategy. We initially conducted ablation experiments on the token selection method, and the results are presented in Tab. 4a. In this table, *random* signifies the stochastic selection, *alternating* represents the periodic selection, and *scores* indicates the selection based on computed scores derived from features or attention scores. *random* and *alternating* represent the two selection strategies of ToMe. Selecting tokens based on calculated importance scores derived from features presents promising performance.

Tab. 4b presents the influence of different evaluation metrics employed to calculate the distance among tokens. We utilize this distance matrix to perform token matching. Among these metrics, the cosine similarity distance metric demonstrates superior performance, hence we utilize cosine similarity to measure the distance between tokens. By comparing the pairwise distances between normal tokens and representative tokens, we can match each normal token with its closest representative token.

[4] `https://github.com/rstrudel/segmenter`

Fig. 5: Token Injection Visualizations. Visualization results from Segmenter + ViT-L/16 appling Token Injector on ADE20K. The first row depicts the input image. The second row illustrates the grouping of all tokens, where gray represents representative tokens, orange represents unique tokens, and teal represents normal tokens. The third row displays the matching results between normal tokens and representative tokens. Different pairs are indicated by boxes of varying colors. The final row showcases the injected results.



(a) compressed token **(b)** unique token

Fig. 6: Ablation studies on the ratio of rows and columns of tokens. (a) Ratio ablation of compressed rows or columns (r_h and r_w). (b) Ablation of retention ratios for unique token rows or columns.

To obtain a reduced token sequence, we employ several methods, as shown in Tab. 4d. *Keep* denotes the preservation of representative tokens and unique tokens without merging normal tokens. *Max pool* and *avg pool* refer to pooling-based methods for token fusion, while *bilinear* represents resizing token sequence through bilinear interpolation. *Average* indicates averaging tokens wihin each matched group to merge them. *Keep* and *average* do not rely on pre-defined priors to adjust the tokens, hence yielding better results. However, *keep* leads

Table 4: Token Adapter ablation experiments using Segmenter + ViT/L-16 on ADE20K validation dataset with official checkpoint and no retraining. r_h and r_w are set to 0.3. Segmenter + ViT/L-16 without Token Adapter achieves 51.82% mIoU at 10.68 img/s. We provide the mean Intersection-over-Union (mIoU) and model inference speed (img/s).

Selection	mIoU	FPS
random	51.38	**15.16**
alternating	51.43	14.96
scores (feat.)	**52.02**	15.09
scores (attn.)	51.13	15.04

(a) Token Selection.

Metric	mIoU	FPS
dot	51.80	15.09
cosine	**52.02**	**15.09**
manhat	51.81	9.21
eucl	51.86	15.09

(b) Distance Metric.

token	mIoU	FPS
X_p^u	51.81	14.85
X_p^m	51.90	14.82
X_r	51.87	**15.15**
X_r^m	**52.02**	15.09

(c) Feature Choice.

Method	mIoU	FPS
keep	50.67	**15.19**
max pool	48.15	14.97
avg pool	48.10	15.10
bilinear	50.08	15.12
average	**52.02**	15.09

(d) Merge Method.

Mode	mIoU	FPS
max	51.93	**15.13**
m-threshold	**52.02**	15.09

(e) Matching Mode.

to the loss of information from normal tokens, thus we ultimately opt for using *average* to merge the tokens.

Restoration Method. Considering dependency on the correlation between the reduced sequence and its original counterpart to restore the sequence length, it is imperative to contemplate the selection of features utilized in calculating this relationship matrix. Within Tab. 4c, the superscript u designates *unmerged* features, while m signifies *merged* features. The variable X_p represents representative tokens, while X_r pertains to reduced tokens.

To facilitate understanding of the meaning of each symbol, we next summarize the algorithmic procedures of the token adapter, and omit the calculation process. For $X \in \mathbb{R}^{N\times C}$, we first partition tokens into $X_p \in \mathbb{R}^{N_p\times C}$, $X_n \in \mathbb{R}^{N_n\times C}$ and $X_u \in \mathbb{R}^{N_u\times C}$. Here, we haven't merged X_n into X_p yet, so we can mark it as X_p^u. X_r is formed by combining X_p^u and X_u, representing the compressed sequence without merging. After merging operations, the new X_p is marked as X_p^m and the new X_r is marked as X_r^m. From the results, it shows that recovery using the merged and reduced token sequence X_r^m works best.

A comparison result is presented for the two matching modes applied to the token ejector in Tab. 4e. In Sec. 3.2, we introduced the token ejector, which matches the most similar tokens for each token in the original token sequence according to the distance matrix $M_d \in \mathbb{R}^{N\times N_r}$. Next, the token vectors from each matched group are averaged to reconstruct the token sequence to its original size. Mode *max* stands for matching the most similar token for each token of

the original token sequence, i.e., one-to-one matching. Mode *m-threshold*, on the other hand, involves filtering based on a set threshold and distance matrix after Mode *max* matching. The filtered matching results are combined with the results from Mode *max*, followed by executing the corresponding fusion process. The results indicate that the latter achieves more accurate recovery of the token sequence than the former.

We also conduct ablation experiments on the ratio of compressed rows, columns, and unique tokens, and the results are shown in Fig. 6. Although we determined the final parameter settings for our method based on the ablation experiments, we believe that there still remains a better solution. It will be discussed in the next section.

5 Discussion and Conclusion

5.1 Limitation

The experimental results mentioned earlier have already demonstrated the effectiveness of our method on dense prediction tasks for hierarchical transformer structures and ViT-like structures. However, there are still many aspects of our work that remain to be explored, which we will analyze one by one next.

Generalizability. While our method has been validated for effectiveness in multiple segmentation tasks, it has not been validated on other more complex vision tasks. Previous methods essentially involved sparse sampling of token sequences, resulting in compressed token sequences being in sequence form only. The compressed sequence features can only be processed by a sequence processing module such as Transformer. Nevertheless, it remains to be examined whether discrete sparse sequence features are more effective than dense features for image tasks. Motivated by this, we designed the token adapter, a method that performs sparse sampling on the rows and columns of image features. Compared to previous methods, our approach yields more compact sampled features, which may be more effective for image tasks.

Adaptability. Although our core idea is rather simple, the Token Adapter still takes time to tune the hyper-parameters. The current setting allows Token Adapter to be directly applied to existing models, but it also limits its further development. We chose not to retrain the model equipped with the token adapter to directly evaluate the effectiveness of our idea. In the future, we will continue to explore more efficient neural network acceleration schemes.

Our core contribution is to compress and reconstruct rows and columns of image features in a way that conforms to the structure of the image. When applied to vision models, Token Adapter only compresses and reconstructs image features and thus can be applied to most vision models. For large language models, the concept of a Token Adapter can still be effectively transferred. By decomposing text embeddings based on the contextual information, and then leveraging the connections between the decomposed parts to compress them, it should be able to improve the efficiency of large language models.

5.2 Conclusion

In this work, we propose Token Adapter to expedite vision transformers used in dense prediction tasks. By injecting normal tokens into representative tokens, we enhance the efficiency of vision transformers. Notably, our method can be applied in off-the-shelf models directly without additional retraining or fine-tuning. Tab. 2 reveals that when applied to the early stages of MaskDINO, the clustering-based Expediting method results in a significant decline in accuracy, and the matching-based Token Adapter method demonstrates superior stability. Clustering algorithms typically require fine-tuning of multiple hyper-parameters, which remain fixed once adjusted. Our approach is expected to leverage mechanisms such as cross-attention to transform fixed hyper-parameters into dynamically learnable parameters, thereby enhancing the method's generalization capability. Theoretically, the design of the compression strategy and reconstruction strategy in our algorithm allows the token adapter to be inserted in the middle of any module designed for image tasks.

In principle, Token Adapter bears resemblance to an unparameterized form of cross-attention mechanism, both blending information from one set of tokens into another set. Our thought lies in how we employ tokens to represent semantic entities within the image. Each entity in the image can be divided into distinct local semantic regions. By assigning several representative tokens to each local semantic region and injecting the remaining normal tokens into their corresponding tokens, we achieve a more refined fusion. After the fusion, the representative tokens representing different semantic regions will engage in information exchange within transformers, greatly enhancing computational efficiency and promoting diversity among tokens. We hope our research can offer a fresh perspective on image features and inspire the design of more efficient methods for extracting image features.

References

1. Bian, Z., Wang, Z., Han, W., Wang, K.: Muti-scale and token mergence: Make your vit more efficient. arXiv preprint arXiv:2306.04897 (2023)
2. Bolya, D., Fu, C.Y., Dai, X., Zhang, P., Feichtenhofer, C., Hoffman, J.: Token merging: Your vit but faster. In: The Eleventh International Conference on Learning Representations (2022)
3. Bonnaerens, M., Dambre, J.: Learned thresholds token merging and pruning for vision transformers. arXiv preprint arXiv:2307.10780 (2023)
4. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European conference on computer vision. pp. 213–229. Springer (2020)
5. Chen, Z., Duan, Y., Wang, W., He, J., Lu, T., Dai, J., Qiao, Y.: Vision transformer adapter for dense predictions. arXiv preprint arXiv:2205.08534 (2022)
6. Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., et al.: Rethinking attention with performers. arXiv preprint arXiv:2009.14794 (2020)

7. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3213–3223 (2016)
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
9. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
10. Fayyaz, M., Koohpayegani, S.A., Jafari, F.R., Sengupta, S., Joze, H.R.V., Sommerlade, E., Pirsiavash, H., Gall, J.: Adaptive token sampling for efficient vision transformers. In: European Conference on Computer Vision. pp. 396–414. Springer (2022)
11. Geng, Z., Guo, M.H., Chen, H., Li, X., Wei, K., Lin, Z.: Is attention better than matrix decomposition? arXiv preprint arXiv:2109.04553 (2021)
12. Guo, M.H., Liu, Z.N., Mu, T.J., Hu, S.M.: Beyond self-attention: External attention using two linear layers for visual tasks. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**(5), 5436–5447 (2022)
13. Hassani, A., Walton, S., Li, J., Li, S., Shi, H.: Neighborhood attention transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6185–6194 (2023)
14. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 16000–16009 (2022)
15. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. arXiv preprint arXiv:2304.02643 (2023)
16. Kong, Z., Dong, P., Ma, X., Meng, X., Sun, M., Niu, W., Shen, X., Yuan, G., Ren, B., Qin, M., et al.: Spvit: Enabling faster vision transformers via soft token pruning. arXiv preprint arXiv:2112.13890 (2021)
17. Li, F., Zhang, H., Xu, H., Liu, S., Zhang, L., Ni, L.M., Shum, H.Y.: Mask dino: Towards a unified transformer-based framework for object detection and segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3041–3050 (2023)
18. Li, Y., Yuan, G., Wen, Y., Hu, J., Evangelidis, G., Tulyakov, S., Wang, Y., Ren, J.: Efficientformer: Vision transformers at mobilenet speed. Advances in Neural Information Processing Systems **35**, 12934–12949 (2022)
19. Liang, W., Yuan, Y., Ding, H., Luo, X., Lin, W., Jia, D., Zhang, Z., Zhang, C., Hu, H.: Expediting large-scale vision transformer for dense prediction without fine-tuning. Advances in Neural Information Processing Systems **35**, 35462–35477 (2022)
20. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. pp. 740–755. Springer (2014)
21. Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., et al.: Swin transformer v2: Scaling up capacity and resolution. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12009–12019 (2022)

22. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 10012–10022 (2021)
23. Long, S., Zhao, Z., Pi, J., Wang, S., Wang, J.: Beyond attentive tokens: Incorporating token importance and diversity for efficient vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10334–10343 (2023)
24. Lu, J., Yao, J., Zhang, J., Zhu, X., Xu, H., Gao, W., Xu, C., Xiang, T., Zhang, L.: Soft: Softmax-free transformer with linear complexity. Advances in Neural Information Processing Systems **34**, 21297–21309 (2021)
25. Mottaghi, R., Chen, X., Liu, X., Cho, N.G., Lee, S.W., Fidler, S., Urtasun, R., Yuille, A.: The role of context for object detection and semantic segmentation in the wild. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 891–898 (2014)
26. Peng, H., Pappas, N., Yogatama, D., Schwartz, R., Smith, N.A., Kong, L.: Random feature attention. arXiv preprint arXiv:2103.02143 (2021)
27. Singh, M., Gustafson, L., Adcock, A., de Freitas Reis, V., Gedik, B., Kosaraju, R.P., Mahajan, D., Girshick, R., Dollár, P., Van Der Maaten, L.: Revisiting weakly supervised pre-training of visual perception models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 804–814 (2022)
28. Strudel, R., Garcia, R., Laptev, I., Schmid, C.: Segmenter: Transformer for semantic segmentation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 7262–7272 (2021)
29. Tian, K., Jiang, Y., Diao, Q., Lin, C., Wang, L., Yuan, Z.: Designing bert for convolutional networks: Sparse and hierarchical masked modeling. arXiv preprint arXiv:2301.03580 (2023)
30. Tolstikhin, I.O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al.: Mlp-mixer: An all-mlp architecture for vision. Advances in neural information processing systems **34**, 24261–24272 (2021)
31. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
32. Wang, Y., Huang, R., Song, S., Huang, Z., Huang, G.: Not all images are worth 16x16 words: Dynamic transformers for efficient image recognition. Advances in Neural Information Processing Systems **34**, 11960–11973 (2021)
33. Wang, Z., Luo, H., Wang, P., Ding, F., Wang, F., Li, H.: Vtc-lfc: Vision transformer compression with low-frequency components. Advances in Neural Information Processing Systems **35**, 13974–13988 (2022)
34. Yang, J., Li, C., Zhang, P., Dai, X., Xiao, B., Yuan, L., Gao, J.: Focal self-attention for local-global interactions in vision transformers. arXiv preprint arXiv:2107.00641 (2021)
35. Yang, R., Ma, H., Wu, J., Tang, Y., Xiao, X., Zheng, M., Li, X.: Scalablevit: Rethinking the context-oriented generalization of vision transformer. In: European Conference on Computer Vision. pp. 480–496. Springer (2022)
36. Ye, D., Lin, Y., Huang, Y., Sun, M.: Tr-bert: Dynamic token reduction for accelerating bert inference. arXiv preprint arXiv:2105.11618 (2021)
37. Yin, H., Vahdat, A., Alvarez, J.M., Mallya, A., Kautz, J., Molchanov, P.: A-vit: Adaptive tokens for efficient vision transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10809–10818 (2022)

38. Yu, Q., Xia, Y., Bai, Y., Lu, Y., Yuille, A.L., Shen, W.: Glance-and-gaze vision transformer. Advances in Neural Information Processing Systems **34**, 12992–13003 (2021)
39. Yu, W., Luo, M., Zhou, P., Si, C., Zhou, Y., Wang, X., Feng, J., Yan, S.: Metaformer is actually what you need for vision. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10819–10829 (2022)
40. Zeng, W., Jin, S., Liu, W., Qian, C., Luo, P., Ouyang, W., Wang, X.: Not all tokens are equal: Human-centric visual analysis via token clustering transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11101–11111 (2022)
41. Zeng, Z., Hawkins, C., Hong, M., Zhang, A., Pappas, N., Singh, V., Zheng, S.: Vcc: Scaling transformers to 128k tokens or more by prioritizing important tokens. arXiv preprint arXiv:2305.04241 (2023)
42. Zheng, M., Gao, P., Zhang, R., Li, K., Wang, X., Li, H., Dong, H.: End-to-end object detection with adaptive clustering transformer. arXiv preprint arXiv:2011.09315 (2020)
43. Zhou, B., Zhao, H., Puig, X., Xiao, T., Fidler, S., Barriuso, A., Torralba, A.: Semantic understanding of scenes through the ade20k dataset. International Journal of Computer Vision **127**, 302–321 (2019)
44. Ziwen, C., Patnaik, K., Zhai, S., Wan, A., Ren, Z., Schwing, A.G., Colburn, A., Fuxin, L.: Autofocusformer: Image segmentation off the grid. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18227–18236 (2023)