Supplementary material for *GTP-ViT: Efficient Vision Transformers via Graph-based Token Propagation*

A. Implementation optimizations

First, we note that both the spatial graph and semantic graph are sparse graphs, with graph sparsity $\frac{8}{N}$ and $\frac{M}{N}$, respectively. For ViT [3] and DeiT [10] models, the total number of image tokens N is usually 196, which indicates less than 5% non-trivial values in the two adjacency matrices. As a result, the mixed graph is also a sparse graph whose sparsity is no more than $\frac{8+M}{N}$ (less than 7% on average). Therefore, we can store the graph in sparse tensors [8] and perform sparse matrix multiplication to accelerate the graph propagation in Sections 3.2.2 and 3.2.3. Besides, for batched inputs that the sparse matrix multiplication does not support, we can use the scatter reduction operation to avoid the dense matrix multiplication. Second, the threshold T_i for sparsifying the semantic graph can be determined by finding the M^{th} largest value in the unsorted array $\mathcal{A}_i^{\text{semantic}}$ with a complexity $O(N)$ rather than sorting the whole array with a complexity $O(N \log N)$.

B. Experiment settings

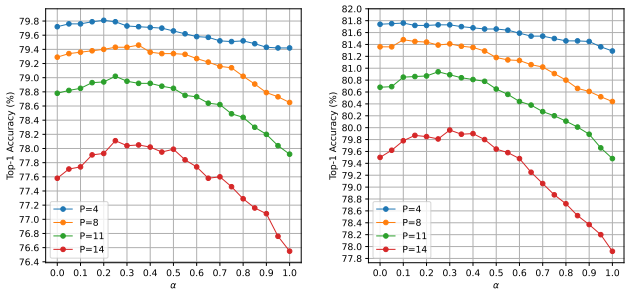
We provide the hyperparameter settings for the compared methods in Tables 1 and 2. These hyperparameters are used to control the reduced computational complexity for the backbone ViT, ensuring fair comparisons.

C. Additional experiments

C.1. Larger ViT backbones

In the main submission, we have validated GTP's effectiveness on small to medium-sized ViT backbones. Moreover, we employ GTP on two large-size ViT backbones: ViT-L [3] and EVA-L [4], which represent ViT backbones with and without the [CLS] token, respectively. Since ViT-L and EVA-L both have 24 layers, the maximum number of propagated tokens P per layer is limited to 8. We also reproduce the state-of-the-art ToMe [1] on these models and present the comparisons in Table 3. It is worth noting that ToMe consumes considerable GPU memory for computing the cosine similarity in each layer. We omit the evaluation on larger models (-H, -G) due to hardware constraints.

From Table 3, we point out that our GTP outperforms ToMe in both model performance and efficiency. Such performance difference is even more significant on ViT backbones without the [CLS] token. For instance, GTP achieves 85.4% top-1 accuracy at 212.0 image/second when taking EVA-L as the backbone, **surpassing ToMe's 80.1% top-1 accuracy by a significant 5.3%** at a similar inference speed.



(a) DeiT-S (b) DeiT-B

Figure 1. **Top-1 accuracy of GTP on ImageNet-1K [2] for various αs.** We evaluate the performance of GTP on both DeiT-S and DeiT-B [10] *without finetuning* w.r.t. different α. For fair comparisons, we employ the same graph type for propagation and set the attention sparsity static at 0.6 for DeiT-S and 0.5 for DeiT-B, respectively. The findings in this experiment are consistent across different settings.

C.2. Graph propagation hyperparameter α

In the graph summarization process $\boldsymbol{X}^s = \boldsymbol{X}^k + \alpha \hat{\mathcal{A}}^p \boldsymbol{X}^p$, we use α to control the magnitude of information broadcast from propagated tokens $\boldsymbol{X}^p$ to kept tokens $\boldsymbol{X}^k$. In this section, we investigate the performance of GTP with respect to different α and visualize the results in Figure 1. In general, as α increases within the range of $[0, 1]$, the corresponding accuracy first rises and then declines, reaching its peak between 0.2˜0.4 for DeiT-S and 0.1˜0.3 for DeiT-B. We explain this phenomenon from two aspects. First, when α is approaching 0, the propagated information becomes trivial, leading to a situation where the propagated tokens' information is not preserved. When $\alpha = 0$, this process merely prunes tokens. Secondly, as α increases, the propagated information gradually dominates the original information of the remaining

Backbone	Method	Hyperparameter	Approximate computational complexity			
			3.5 GMACs	3.0 GMACs	2.6 GMACs	2.3 GMACs
DeiT-S [10]	DynamicViT [9]	base keep ratio ρ	$\rho = 0.8$	$\rho = 0.7$	$\rho = 0.6$	$\rho = 0.5$
	EViT [7]	token keeping rate k	$k = 0.8$	$k = 0.7$	$k = 0.6$	$k = 0.5$
	Evo-ViT [11]	selection ratio p	$p = 0.7$	$p = 0.5$	$p = 0.4$	$p = 0.3$
	Tri-Level [6]	token keep ratio R_T	$R_T = 0.8$	$R_T = 0.7$	$R_T = 0.6$	$R_T = 0.5$
	ToMe [1]	token reduce r per layer	$r = 8$	$r = 11$	$r = 14$	$r = 16$
	ATS [5]	ATS block layers	layer 7 to 11	layer 6 to 11	layer 5 to 11	layer 3 to 11
	GTP (ours)	propagated tokens P per layer	$P = 8$	$P = 11$	$P = 14$	$P = 16$

Table 1. **Hyperparameter settings for the baseline methods, taking DeiT-S as the backbone.**

Backbone	Method	Hyperparameter	Approximate computational complexity				
			15.3 GMACs	13.1 GMACs	11.6 GMACs	9.8 GMACs	8.8 GMACs
DeiT-B [10]	DynamicViT [9]	base keep ratio ρ	$\rho = 0.9$	$\rho = 0.8$	$\rho = 0.7$	$\rho = 0.6$	$\rho = 0.5$
	EViT [7]	token keeping rate k	$k = 0.9$	$k = 0.8$	$k = 0.7$	$k = 0.6$	$k = 0.5$
	Evo-ViT [11]	selection ratio p	$p = 0.8$	$p = 0.7$	$p = 0.5$	$p = 0.4$	$p = 0.3$
	Tri-Level [6]	token keep ratio R_T	$R_T = 0.9$	$R_T = 0.8$	$R_T = 0.7$	$R_T = 0.6$	$R_T = 0.5$
	ToMe [1]	token reduce r per layer	$r = 4$	$r = 8$	$r = 11$	$r = 14$	$r = 16$
	ATS [5]	ATS block layers	layer 9 to 11	layer 8 to 11	layer 6 to 11	layer 3 to 11	layer 1 to 11
	GTP (ours)	propagated tokens P per layer	$P = 4$	$P = 8$	$P = 11$	$P = 14$	$P = 16$

Table 2. **Hyperparameter settings for the baseline methods, taking DeiT-B as the backbone.**

tokens, which results in an over-smoothing problem and subsequently hinders performance.

C.3. The number of graph neighbours

We study the influence of the number of semantic neighbours M on model performance and plot the accuracy in Figure 2. For fair comparisons, we apply GTP on DeiT-S and DeiT-B with static attention sparsity and alpha at 0.5 and 0.2, respectively. Figures 2(a) and 2(c) illustrate the results obtained by only employing the semantic graph for token propagation, with respect to different M. It can be observed that when the number of propagated tokens P is small (e.g., $P = 4$ or $P = 8$), increasing the semantic neighbours would first slightly improves the accuracy and then converges. However, when P becomes large (e.g., $P = 14$), increasing the semantic neighbours may lead to a performance drop. This can be attributed to the aggregation of redundant information, where one kept token encapsulates an excessive number of propagated tokens that may not be semantically close to it. Figures 2(b) and 2(d) show that integrating the semantic graph with the spatial graph stabilizes the trend of accuracy, indicating the significance of the spatial relationship in token summarization.

C.4. Computational complexity comparison

As stated in the Introduction section, ToMe [1] encounters a computational bottleneck in the pair-wise token matching process, whose computational complexity is proportional to the feature dimensions and the square of the number of tokens. Compared with ToMe, GTP demonstrates faster inference speed and accomplishes better information preservation results. In this section, we provide an in-depth analysis of the enhancements in computational efficiency achieved by GTP.

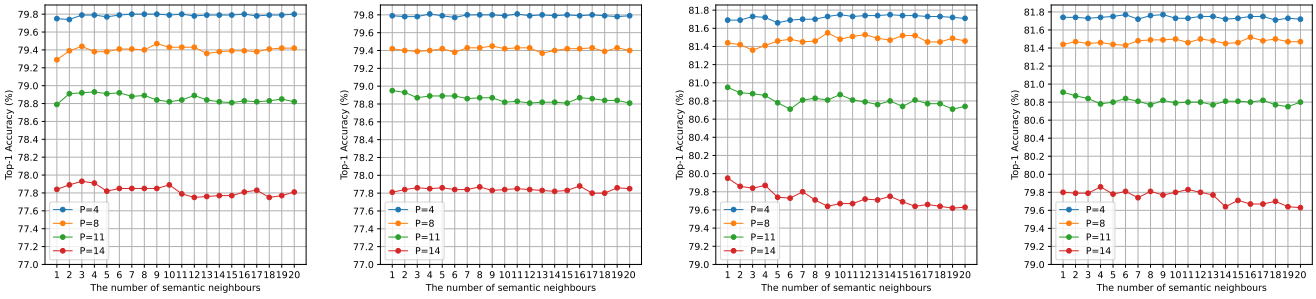
As a plug-and-play component, GTP inserts the token summarization module between the MHSA layer and the FFN layer in each ViT block, which behaves analogously to ToMe. Therefore, when the number of eliminated tokens is the same, the computational complexity of the backbone model for GTP and ToMe should be the same. Consequently, we only consider the additional computational costs (e.g., token matching, token selection and token propagation) introduced by the two models in this analysis. We list the denotations before the theoretical analysis as follows:

$$
\begin{aligned}
&N : \text{The total number of tokens in the backbone network.}\\
&N_l : \text{The number of remaining tokens in layer } l, \text{ where}\\
&\quad N_l = N - (l-1)M.\\
&M : \text{The number of eliminated tokens in each layer.}\\
&C : \text{The dimension of features.}\\
&L : \text{The total number of layers.}\\
&H : \text{The number of heads.}
\end{aligned}
\tag{1}
$$

For ToMe, the token matching processing first splits tokens into two sets and then calculates the cosine similarity between each pair of tokens from the two sets. The computational complexity for this process in layer l is $\frac{1}{4}N_l^2C$. Besides, ToMe merges M tokens in each layer, whose total computational complexity is MC in each layer. As a result, the total additional computational complexity G_{ToMe}

Backbone	Method	$P=0$		$P=1$		$P=2$		$P=3$		$P=4$		$P=5$		$P=6$		$P=7$		$P=8$	
		Acc. (%)	Speed (img/s)	Acc. (%)	Speed (img/s)	Acc. (%)	Speed (img/s)	Acc. (%)	Speed (img/s)	Acc. (%)	Speed (img/s)	Acc. (%)	Speed (img/s)	Acc. (%)	Speed (img/s)	Acc. (%)	Speed (img/s)	Acc. (%)	Speed (img/s)
ViT-L [10]	ToMe [1]	85.8	123.4	**85.8**	122.0	85.7	132.1	85.5	142.2	85.3	153.8	85.0	169.0	84.7	184.8	84.2	204.0	**83.7**	228.5
	GTP (ours)			**85.8**	**125.4**	**85.8**	**134.4**	**85.8**	**144.9**	**85.5**	**157.4**	**85.3**	**172.0**	**85.0**	**188.3**	**84.3**	**208.8**	**83.7**	**234.0**
EVA-L [4]	ToMe [1]	87.9	123.1	87.7	123.6	87.6	132.3	87.3	142.5	87.0	155.2	86.5	169.6	85.6	185.9	84.0	207.0	80.1	230.9
	GTP (ours)			**87.9**	**125.6**	**87.8**	**134.6**	**87.8**	**145.3**	**87.7**	**158.0**	**87.5**	**172.5**	**87.2**	**188.9**	**86.5**	**212.0**	**85.4**	**234.9**

Table 3. **Performance on larger ViT models.** We validate GTP's performance on two large-size ViT models, ViT-L [3] and EVA-L [4], where ViT-L employs the [CLS] token while EVA-L does not have the [CLS] token. Since both models have 24 layers, we can eliminate at most 8 tokens per layer. Bond font means better. GTP constantly outperform ToMe on large ViT models with and without the [CLS] token.



(a) GTP on DeiT-S with semantic graph (b) GTP on DeiT-S with mixed graph (c) GTP on DeiT-B with semantic graph (d) GTP on DeiT-B with mixed graph

Figure 2. **Top-1 accuracy of GTP with various numbers of semantic neighbours M.** We evaluate the performance of GTP on both DeiT-S and DeiT-B [10] *without finetuning* w.r.t. different numbers of semantically connected neighbours.

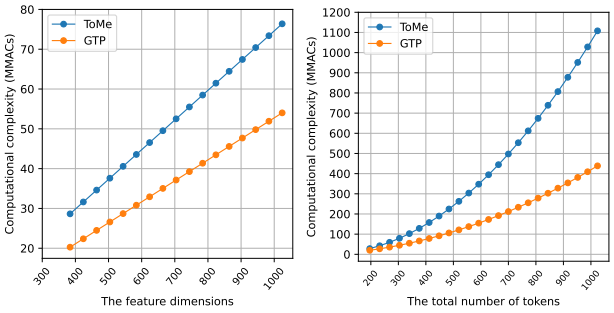
introduced by ToMe is calculated as

$$\begin{aligned} G_{\text{ToMe}} &= \sum_{l=1}^{L}(\frac{1}{4}N_l^2C + MC) \\ &= \frac{1}{4}C\sum_{l=1}^{L}N_l^2 + LMC \\ &= \frac{1}{4}C\sum_{l=1}^{L}(N-(l-1)M)^2 + LMC \\ &= \frac{1}{4}C\sum_{l=1}^{L}(N^2 - 2(l-1)NM + (l-1)^2M^2) \\ &\quad + LMC \\ &= \frac{1}{4}LN^2C + \frac{1}{4}(L-L^2)NMC \\ &\quad + (\frac{1}{12}L^3 + \frac{1}{12}L^2 + \frac{1}{8}L)M^2C + LMC \end{aligned} \tag{2}$$

We then calculate the computational complexity for GTP. GTP first constructs the semantic graph for an input image after the token embedding layer with a computational complexity N^2C. Next, it selects tokens with a computational complexity at most HN_l in layer l. And finally, the tokens are propagated with computational complexity at most $(N_l - M)MC$ in layer l. Consequently, the total additional computational complexity G_{GTP} of GTP is

$$\begin{aligned} G_{\text{GTP}} &= N^2C + \sum_{l=1}^{L}(HN_l + (N_l - M)MC) \\ &= N^2C + \sum_{l=1}^{L}(H(N-(l-1)M) \\ &\quad + (N-(l-1)M - M)MC) \\ &= N^2C + LHN + LMNC - \frac{1}{2}(L^2-L)HM \\ &\quad - \frac{1}{2}(L+L^2)M^2C. \end{aligned} \tag{3}$$

Given $N = 197, L = 12, H = 6, C = 384$ and $M = 8$ for DeiT-S, we can get $G_{\text{GTP}} \approx 20.1$MMACs, which is smaller than $G_{\text{ToMe}} \approx 28.3$MMACs. On DeiT-B where $N = 197, L = 12, M = 8, H = 12$ and $C = 768$, we observe that $G_{\text{GTP}} \approx 40.5$MMACs is much smaller than $G_{\text{ToMe}} \approx 57.3$MMACs. Figure 3 illustrates the additional computational complexity change with respect to the total number of tokens N and the feature dimensions C. It is obvious that our GTP introduces far less additional computational complexity than ToMe, and the difference signifies when N and C increase. It indicates the efficiency of GTP on large-size ViTs whose feature dimensions may exceed 1024, as well as on ViTs for dense prediction tasks where the number of tokens may be more than 1024.



(a) (b)

Figure 3. **Comparisons on the additional computational complexities introduced by ToMe [1] and our GTP.** We plot the computational complexity (measured in MMACs) with respect to the dimension of token features in (a) and the total number of tokens in (b).

References

[1] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. In *ICLR*, 2023. 1, 2, 3, 4

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 1

[3] 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. 1, 3

[4] Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva: Exploring the limits of masked visual representation learning at scale. In *CVPR*, 2023. 1, 3

[5] Mohsen Fayyaz, Soroush Abbasi Koohpayegani, Farnoush Rezaei Jafari, Sunando Sengupta, Hamid Reza Vaezi Joze, Eric Sommerlade, Hamed Pirsiavash, and Jürgen Gall. Adaptive token sampling for efficient vision transformers. In *ECCV*, 2022. 2

[6] Zhenglun Kong, Haoyu Ma, Geng Yuan, Mengshu Sun, Yanyue Xie, Peiyan Dong, Xin Meng, Xuan Shen, Hao Tang, Minghai Qin, et al. Peeling the onion: Hierarchical reduction of data redundancy for efficient vision transformer training. In *AAAI*, 2023. 2

[7] Youwei Liang, GE Chongjian, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Evit: Expediting vision transformers via token reorganizations. In *ICLR*, 2021. 2

[8] PyTorch. Torch.sparse, pytorch 2.0 documentation. https://pytorch.org/docs/stable/sparse.html, 2023. 1

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

[10] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICLR*, 2021. 1, 2, 3

[11] 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*, 2022. 2