Adaptive Token Sampling For Efficient Vision Transformers
Supplementary Document

Mohsen Fayyaz[1,6,*†] Soroush Abbasi Koohpayegani[2,*†] Farnoush Rezaei Jafari[3,4,*] Sunando Sengupta[1] Hamid Reza Vaezi Joze[5] Eric Sommerlade[1] Hamed Pirsiavash[2] Juergen Gall[6]

[1]Microsoft [2]University of California, Davis [3]Machine Learning Group, Technische Universität Berlin, [4]Berlin Institute for the Foundations of Learning and Data [5]Meta Reality Labs [6]University of Bonn

1 Runtime

Throughput: While ATS is a super-light module, there is still a small cost associated with I/O operations. For a DeiT-S network with a single ATS stage, the sampling overhead is about 1.5% of the overall computation which is negligible compared to the large savings due to the dropped tokens. To further elaborate on this, we have reported the throughput (images/s) of the DeiT-S model with/without our ATS module in Table 1. As it can be seen, the speed-up of our module is aligned with its GFLOPs reduction.

Batch Processing: While for most applications the inference is performed for a single image or video, ATS can also be used for inference with a mini-batch. To this end, we rearrange the tokens of each image so that the sampled tokens are in the lower indices. Then, we remove the last tokens completely to reduce the computation. This way, we only process m tokens, where $m = \max_i(K'_i + 1)$ over all images i of the mini-batch. In the worst case scenario (*e.g.* a very large minibatch), we will keep all $K+1$ first tokens after rearrangement. This will still reduce the computation by a factor of $\frac{N+1}{K+1}$. For example, using a mini-batch of size 512 on the ImageNet validation set, m is 129 in Stage 7 of the DeiT-S+ATS model, which is smaller than the total number of tokens (197). Therefore, we discard at least 68 tokens in stage 7 even in a mini-batch setting. Moreover, for the fully connected layers in a transformer block, which requires most of the computation [7], we can flatten the mini-batch dimension and forward only non-zero tokens of the whole mini-batch in parallel through the fully connected layers.

---

*Equal Contribution

†Work has been done during an internship at Microsoft

Table 1. Runtime comparison: We run the models on a single RTX6000 GPU (CUDA 11.0, PyTorch 1.8, image size: 224×224). We average the value of throughput over 20 runs. We add ATS to multiple stages of the DeiT-S model and fine-tune the network on the ImageNet dataset.

Model	Params (M)	GFLOPs	Throughput	Top-1
Deit-S [9]	22.05	4.6	1010	79.8
Deit-S+ATS	22.05	**2.9**	**1403**	79.7

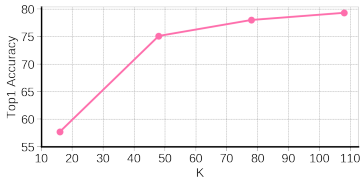

Fig. 1. Effect of K: We varied the value of K in the ATS module to study the effect of K on the top-1 accuracy. K=48 corresponds to 2 GFLOPs. The backbone model is DeiT-S pre-trained on ImageNet-1K.

2 The Effect of K

In Fig. 5 of the main paper, we varied the value of K to achieve different GFLOPs levels (Top-1 Accuracy vs. GFLOPs). In Fig. 1, we study the effect of varying K in the ATS module of the single-stage DeiTS+ATS model with fine-tuning. Interestingly, even sampling only 48 tokens (2 GFLOPs) achieves 75% accuracy.

3 ATS Integration Without Further Training

One of the most important aspects of our approach is that it can be added to any pre-trained off-the-shelf vision transformer. For example, our not fine-tuned multi-stage DeiT-S+ATS model (Fig. 5(c) in the paper) has only a 0.6% (Table 1 in the paper) top-1 accuracy drop while it has improved the efficiency by about 1.6 GFLOPs without any further training of the backbone model. We also observe the same performance on video data. As reported in Table 2, our not fine-tuned XViT+ATS model has only a 1.1% top-1 accuracy drop while it has improved the efficiency by about 329 GFLOPs without any further training of the backbone model. This capability of our ATS module roots back in its adaptive inverse transform sampling strategy. Our ATS module samples informative tokens based on their contributions to the classification token. Uninformative tokens that only slightly contribute to the final prediction receive lower attention weights for the classification token. Therefore, the output classification token will be only marginally affected by removing such redundant tokens. On the other hand, the redundant tokens are less similar to the informative tokens and receive lower attention weights for those tokens in the attention matrix. Consequently, they do not contribute much to the value of informative tokens and eliminating them does not change the way informative tokens are contributing to the output classification token.

Table 2. Our ATS module is added to XViT [2] pre-trained on Kinetics-600.

Model	Top-1	GFLOPs
XViT+ATS Not-Finetuned(16×)	83.4	**521**
XViT+ATS Finetuned(16×)	84.4	**521**
XViT(16×)	**84.5**	850

4 Attention Map Visualization

As shown in Fig. 2, the attention maps become more focused on the birds and less on the background at the later stages, which is aligned with our observations on the sampled tokens at each stage.

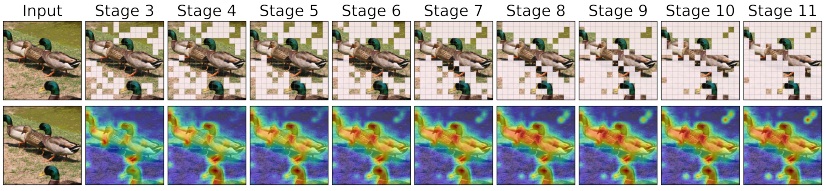

Fig. 2. Visualization of the sampled tokens and attention maps of a not fine-tuned multi-stage DeiT-S+ATS.

5 Implementation Details

In our experiments for image classification, we use the ImageNet [4] dataset with 1.28M training images and 1K classes. We evaluate our adaptive models, which are equipped with the ATS module, on 50K validation images of this dataset. In our experiments for action recognition, we use the Kinetics-400 [5] and Kinetics-600 [3] datasets containing short clips (typically 10 seconds long) sampled from YouTube. Kinetics-400 and Kinetics-600 consist of 400 and 600 classes, respectively. The versions of Kinetics-400 and Kinetics-600 used in this paper consist of approximately 261k and 457k clips, respectively. Note that these numbers are lower than the original datasets due to the removal of certain videos from YouTube. Our networks for image classification are trained on 8 NVIDIA Quadro RTX 6000 GPUs and for action recognition on 8 NVIDIA A100 GPUs.

5.1 DeiT + ATS

Training To fine-tune our adaptive models, we follow the DynamicViT [8] training settings. We use the DeiT model's pre-trained weights to initialize the backbones of our adaptive network and train it for 30 epochs using AdamW optimizer. The learning rate and batch size are set to 5e-4 and 8×96, respectively.

We use the cosine scheduler to train the networks. For both multi and single stage models, we set $K = 197$ during training.

Evaluation We use the same setup as [9] for evaluating our adaptive models. To evaluate the performance of our multi-stage DeiT-S+ATS model with different average GFLOPs levels of 3, 2.5, and 2, we set $K_n = \max(\lfloor \rho \times \#InputTokens_n \rceil, 8)$ in which ρ is set to 1, 0.87, 0.72, respectively, and n is the stage index. For the single-stage model, we set $K = 108$, 78, 48 to evaluate the model with different average GFLOPs levels of 3, 2.5, and 2.

5.2 CvT + ATS

We integrate our ATS module into the 1^{st} to 9^{th} blocks of the 3^{rd} stage of the CvT-13 [10] and CvT-21 [10] networks. For both CvT models, we do not use any convolutional projection layers in the transformer blocks of stage 3.

Training To train our adaptive models, we follow most of the CvT [10] network's training settings. We use the CvT model's pre-trained weights to initialize the backbones of our adaptive networks and train them for 30 epochs using AdamW optimizer. The learning rate and batch size are set to 1.5e-6 and 128, respectively. We use the cosine scheduler to train the networks.

Evaluation To evaluate our CvT+ATS model, we use the same setup as [10].

5.3 PS-ViT + ATS

Training To fine-tune our adaptive models, we follow the PS-ViT [11] training settings. We use the PS-ViT model's pre-trained weights to initialize the backbones of our adaptive network and train it for 30 epochs using AdamW optimizer. The learning rate and batch size are set to 5e-4 and 8×96, respectively. We use the cosine scheduler to train the networks.

Evaluation To evaluate our CvT+ATS model, we use the same setup as [11].

5.4 XViT + ATS

We integrate our ATS module into the stages 3 to 11 of the XViT [2] network.

Training To train our adaptive model, we follow most of the XViT [2] network's training settings. We use the XViT model's pre-trained weights to initialize the backbone of our adaptive network and train it for 10 epochs using SGD optimizer. The learning rate and batch size are set to 1.5e-6 and 64, respectively. We use the cosine scheduler to train the networks.

Evaluation To evaluate our XViT+ATS model, we use the same setup as [2].

5.5 TimeSformer + ATS

We integrate our ATS module into the stages 3 to 5 of the TimeSformer [1] network.

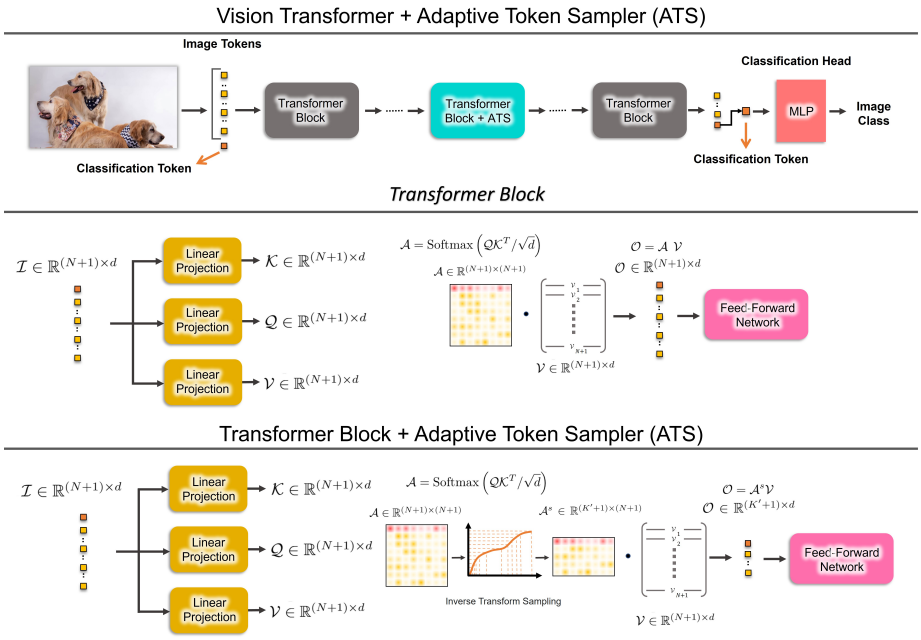

Fig. 3. The Adaptive Token Sampler (ATS) can be integrated into the self-attention layer of any transformer block of a vision transformer model (top). The ATS module takes at each stage a set of input tokens $\mathcal{I}$. The first token is considered as the classification token in each block of the vision transformer. The attention matrix $\mathcal{A}$ is then calculated by the dot product of the queries $\mathcal{Q}$ and keys $\mathcal{K}$, scaled by $\sqrt{d}$. Having selected the significant tokens, we then sample the corresponding attention weights (rows of the attention matrix $\mathcal{A}$) to get $\mathcal{A}^s$. Finally, we softly downsample the input tokens $\mathcal{I}$ to output tokens $\mathcal{O}$ using the dot product of $\mathcal{A}^s$ and $\mathcal{V}$. Next, we forward the output tokens $\mathcal{O}$ through a Feed-Forward Network (FFN) to get the output of the transformer block.

Training To train our adaptive model, we follow most of the TimeSformer [1] network's training settings. We use the TimeSformer model's pre-trained weights to initialize the backbones of our adaptive networks and train it for 5 epochs using SGD optimizer. The learning rate and batch size are set to 5e-6 and 32, respectively. We use the cosine scheduler to train the networks.
Evaluation To evaluate our TimeSformer-HR+ATS and TimeSformer-L+ATS models, we use the same setup as [1].

5.6 Integrating ATS into a Transformer Block

Unlike a standard transformer block in vision transformers, we assign a score to each token and use inverse transform sampling to prune the rows of the attention matrix $\mathcal{A}$ to get $\mathcal{A}^s$. Next, we get the output $\mathcal{O} = \mathcal{A}^s\mathcal{V}$ and forward it to the

Feed-Forward Network (FFN) of the transformer block. We visualize the details of our ATS module, which is integrated into a standard self-attention layer in Fig. 3.

6 Ablation

6.1 Score Assignment

In the main paper, we analyzed the impact of using different tokens to calculate the significance scores S. In all of our experiments, we suggested keeping the classification token since the loss is defined on this token and discarding it may negatively affect the performance. To represent the importance of this token experimentally, we sum over the attention weights of all tokens (rows of the attention matrix) to find the most significant tokens. We show this in Fig. 4 as Self-Attention Score (CLS Enforced). In contrast to our previous experiments, we allow ATS to remove the classification token when it is of low importance based on the significance scores S. We show the results of this experiment in Fig. 4 as Self-Attention Score (CLS Not Enforced). As it can be seen in Fig. 4, discarding the classification token reduces the top-1 accuracy.

Table 3. Comparison of the inverse transform sampling approach with the top-K selection. We finetune and test two different versions of the multi-stage DeiT-S+ATS model: with (1) top-K token selection and (2) inverse transform token sampling. We report the top-1 accuracy of both networks on the ImageNet validation set. For the model with the top-K selection approach, we set $K_n = \lfloor 0.865 \times \#InputTokens_n \rceil$ where n is the stage index. For example, $K_3 = 171$ in stage 3.

Method	Top-1 acc	GFLOPs
Top-K	78.9	2.9
Inverse Transform Sampling	**79.7**	2.9

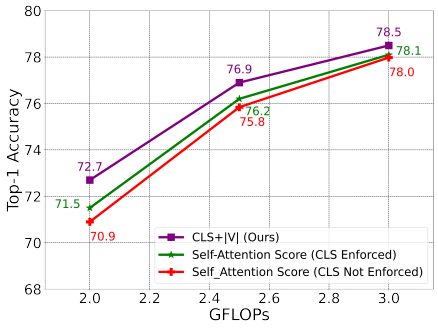

Fig. 4. Impact of allowing ATS to discard the classification token on the network's accuracy. The model is a single stage DeiT-S+ATS without finetuning.

6.2 Candidate Token Selection

As mentioned in the main paper, we employ the inverse transform sampling approach to softly downsample input tokens. We investigated this in Section 4 of the paper. To better analyze it, we also evaluate the performance of our trained multi-stage DeiT-S+ATS model when picking the top K tokens with the

highest significance scores $\mathcal{S}$. To this end, we trained our DeiT-S+ATS network with the top-K selection approach and compared it to our DeiT-S+ATS model with the inverse transform sampling method. As it can be seen in Table 3, our inverse transform sampling approach outperforms the top-K selection with and without training (Fig 5(a) in paper). As discussed earlier, our inverse transform sampling approach does not hardly discard all tokens with lower significance scores and hence provides a more diverse set of tokens for the following layers. This sampling strategy also helps the model to gain a better performance after training, thanks to a more diversified token selection.

6.3 ATS Placement

To evaluate the effect of our ATS module's location within a vision transformer model, we add it to different stages of the DeiT-S network and evaluate it on the ImageNet validation set without finetuning the model. To have a better comparison, we set the average computation costs of all experiments to 3 GFLOPs. As it can be seen in Table 4, integrating the ATS module into the first stage of the DeiT-S model results in a poor top-1 accuracy of 73.1%. On the other hand, integrating the ATS module into stage 3 results in a 78.5% top-1 accuracy. As mentioned before, earlier transformer blocks are more prone to predict noisier attention weights for the classification token. Therefore, integrating our ATS module into the first stage performs worse than incorporating it into the stage 3. Although the attention weights of the stage 6 are less noisy, we have to discard more tokens to reach the desired GFLOPs level of 3. For example in stages 0, 3, and 6, we set K to 130, 108, and 56, respectively. The highest accuracy is obtained when we integrate the ATS module into multiple stages of the DeiT-S model. This is because of the progressive token sampling that occurs in a multi-stage DeiT-S+ATS model. In other words, a multi-stage DeiT-S+ATS network can gradually decrease the GFLOPs by discarding fewer tokens in the earlier stages, while a single-stage DeiT-S+ATS model has to discard more tokens in the earlier stages to reach the same GFLOPs level. We also added the ATS module into all stages, yielding average GFLOPs of 2.6 and 76.9% top-1 accuracy.

Table 4. Evaluating the integration of the ATS module into different stages of DeiT-S [9].

Stage(s)	0	3	6	3-11
Top-1 Accuracy	73.1	78.5	77.4	**79.2**

6.4 Adding ATS to Models with Other Token Pruning Approaches

To better evaluate the performance of our adaptive token sampling approach, we also add our module to the state-of-the-art efficient vision transformer EViT-DeiT-S [6]. EViT [6] introduces a token reorganization method that first identifies the top-K important tokens by computing token attentiveness between the tokens and the classification token and then fuses less informative tokens. Interestingly, our ATS module can also be added to the EViT-DeiT-S model and further decrease the GFLOPs, as shown in Table 5. These results demonstrate the superiority of our adaptive token sampling approach compared to static token pruning methods. We integrate our ATS module into stages 4, 5, 7, 8, 10, and 11 of the EViT-DeiT-S backbone and fine-tune them for 10 epochs following our fine-tuning setups on the ImageNet dataset discussed earlier.

Table 5. Evaluating the EViT-DeiT-S [6] model's performance when integrating the ATS module into it with $K_n = \lfloor 0.7 \times \#InputTokens_n \rceil$ where n is the stage index.

Model	Top-1 acc	GFLOPs
EViT-DeiT-S (30 Epochs) [6]	79.5	3.0
EViT-DeiT-S (30 Epochs)+ATS	79.5	**2.5**
EViT-DeiT-S (100 Epochs) [6]	79.8	3.0
EViT-DeiT-S (100 Epochs)+ATS	79.8	**2.5**

7 More Visualizations

We show more visual results in Fig. 5. We select several images of the ImageNet validation set with various amounts of detail and complexity. We visualize the progressive token sampling procedure of our multi-stage DeiT-S+ATS model for the selected images. The number of output tokens of each ATS module in the multi-stage DeiT-S+ATS model is limited by the number of its input tokens, which is 197. Our adaptive model samples a higher number of tokens when the input images are more cluttered. We can also observe that the sampled tokens are more scattered in images with more details compared to more plain images.

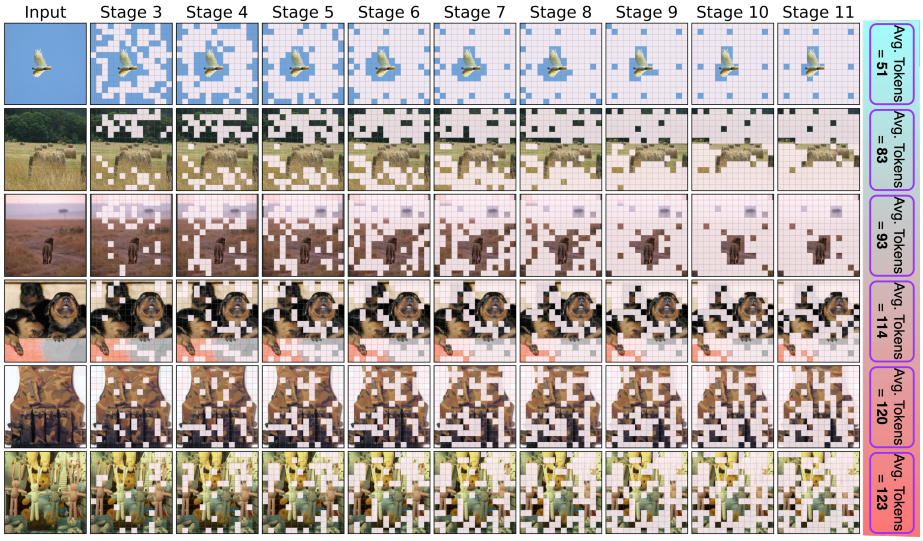

Fig. 5. Visualization of the gradual token sampling procedure in the multi-stage DeiT-S+ATS model. We integrate our ATS module into the stages 3 to 11 of the DeiT-S model. The tokens that are sampled at each stage of the network are shown for images that are ordered by their complexity (from low complexity to high complexity). We visualize the tokens, which are discarded, as masks over the input images. As it can be seen, a higher number of tokens are sampled for more cluttered images while a lower number of tokens are required when the images contain less details. Additionally, we can see that the sampled tokens are more focused and less scattered in images with less details.

References

1. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding. In: International Conference on Machine Learning (ICML) (2021) 4, 5
2. Bulat, A., Perez Rua, J.M., Sudhakaran, S., Martinez, B., Tzimiropoulos, G.: Space-time mixing attention for video transformer. In: Advances in Neural Information Processing Systems (NeurIPS) (2021) 3, 4
3. Carreira, J., Noland, E., Banki-Horvath, A., Hillier, C., Zisserman, A.: A short note about kinetics-600. In: arXiv preprint arXiv:1808.01340v1 (2018) 3
4. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2009) 3
5. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, A., Suleyman, M., Zisserman, A.: The kinetics human action video dataset. In: arXiv preprint arXiv:1705.06950 (2017) 3
6. Liang, Y., Ge, C., Tong, Z., Song, Y., Wang, J., Xie, P.: Not all patches are what you need: Expediting vision transformers via token reorganizations. In: International Conference on Learning Representations (ICLR) (2022) 8
7. Marin, D., Chang, J.H.R., Ranjan, A., Prabhu, A.K., Rastegari, M., Tuzel, O.: Token pooling in vision transformers. arXiv preprint arXiv:2110.03860 (2021) 1
8. Rao, Y., Zhao, W., Liu, B., Lu, J., Zhou, J., Hsieh, C.J.: Dynamicvit: Efficient vision transformers with dynamic token sparsification. In: Advances in Neural Information Processing Systems (NeurIPS) (2021) 3
9. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jegou, H.: Training data-efficient image transformers and distillation through attention. In: International Conference on Machine Learning (ICML) (2021) 2, 4, 7
10. Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L., Zhang, L.: Cvt: Introducing convolutions to vision transformers. In: IEEE/CVF International Conference on Computer Vision (ICCV) (2021) 4
11. Yue, X., Sun, S., Kuang, Z., Wei, M., Torr, P., Zhang, W., Lin, D.: Vision transformer with progressive sampling. In: IEEE/CVF International Conference on Computer Vision (ICCV) (2021) 4