# SlowFormer: Adversarial Attack on Compute and Energy Consumption of Efficient Vision Transformers

## Supplementary Material

In sections 1 and 2, we provide visualizations of our learnt patches and token dropping respectively. In Sec. 3, we provide additional details on our train and test settings.

## 1. Patch Visualization

In Fig. 1, we visualize the optimized patches for each of the three efficient methods. All patches are of size $64 \times 64$, contributing to 16 of the 196 tokens for the input image. Surprisingly, a rectangular region in the patch for A-ViT , corresponding to one token, is almost entirely black.

We optimize patches for A-ViT using different initializations and visualize them in Fig. 2. All patches achieve Attack Success close to $100\%$. Presence of multiple universal adversarial patches highlights the vulnerability of the current efficient methods.

We show the evolution of the patch as training progresses in Fig. 4. The patch is trained to attack A-ViT approach. We observe that the patch converges quickly, requiring less than an epoch for $100\%$ Attack Success. The patch at 1000 iterations (0.1 epoch) is similar to that at 10000 iterations (1 epoch) in terms of both appearance and attack performance.

## 2. Visualization of Token Dropping

In Fig. 3, we visualize dropped tokens in A-ViT-Small with and without our attack. Our attack significantly decreases the number of pruned tokens, resulting in more compute and energy consumption for the efficient transformer model.

## 3. Implementation Details

**ATS Details:** As in ATS [1], we replace layers 3 through 9 of ViT networks with the ATS block and set the maximum limit for the number of tokens sampled to 197 for each layer. We train the patch for 2 epochs with a learning rate of $0.4$ for ViT-Tiny and $lr = 0.2$ for ViT-Base and ViT-Small. We use a batch size of $1024$ and different loss coefficients for each layer of ATS. For DeiT-Tiny we use $[1.0, 0.2, 0.2, 0.2, 0.01, 0.01, 0.01]$, for DeiT-Small we use $[1.0, 0.2, 0.05, 0.01, 0.005, 0.005, 0.005]$, and for DeiT-Base we use $[2.0, 0.1, 0.02, 0.01, 0.005, 0.005, 0.005]$ The weights are vastly different at initial and final layers to account for the difference in loss magnitudes across layers.

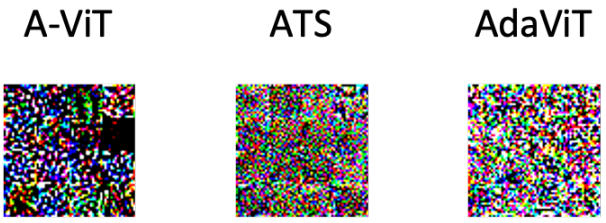


Figure 1. **Visualization of optimized patch:** We show the learnt universal patches for each of the three efficient methods that we attack.

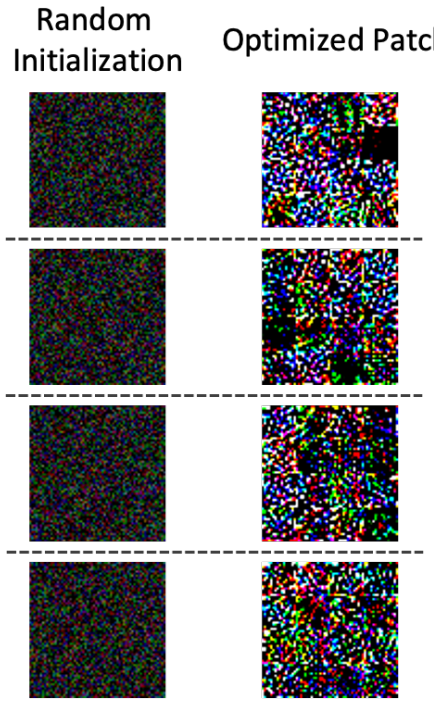


Figure 2. **Optimized patches With different initializations:** Here, we show the optimized patches for A-ViT . A different initialization is used to train each of these patches. All patches achieve Attack Success close to $100\%$. Presence of multiple universal adversarial patches highlights the vulnerability of the current efficient methods.

**A-ViT Details:** The patches are optimized for one epoch with a learning rate of $0.2$ and a batch size of $512$ ($128 \times 4$GPUs) using AdamW [2] optimizer. We optimize the patches for $4$ epochs for patch length 32 and below. For CIFAR-10 experiments, the images are resized from $32 \times 32$ to $256 \times 256$ and a $224 \times 224$ crop is used as the input. For the training of adversarial defense, we generate 5 patches per epoch of adversarial training and limit the number of iterations for patch generation to $500$. The learning rate for

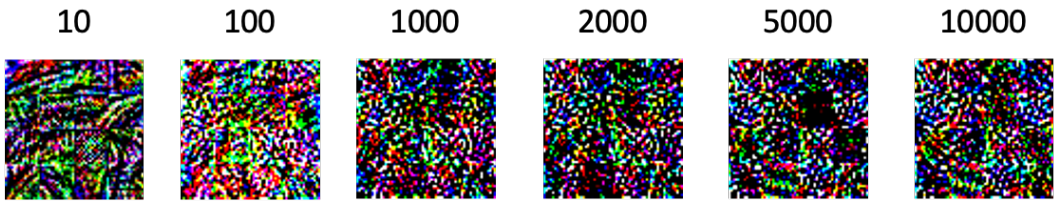


Figure 3. **Visualization of patch optimization:** We train our patch to attack A-ViT and display the patch at various stages of optimization. We observe that the patch converges quickly. The patch at 1000 iterations (0.1 epoch) is similar to that at 10000 iterations (1 epoch) in terms of both appearance and attack performance.

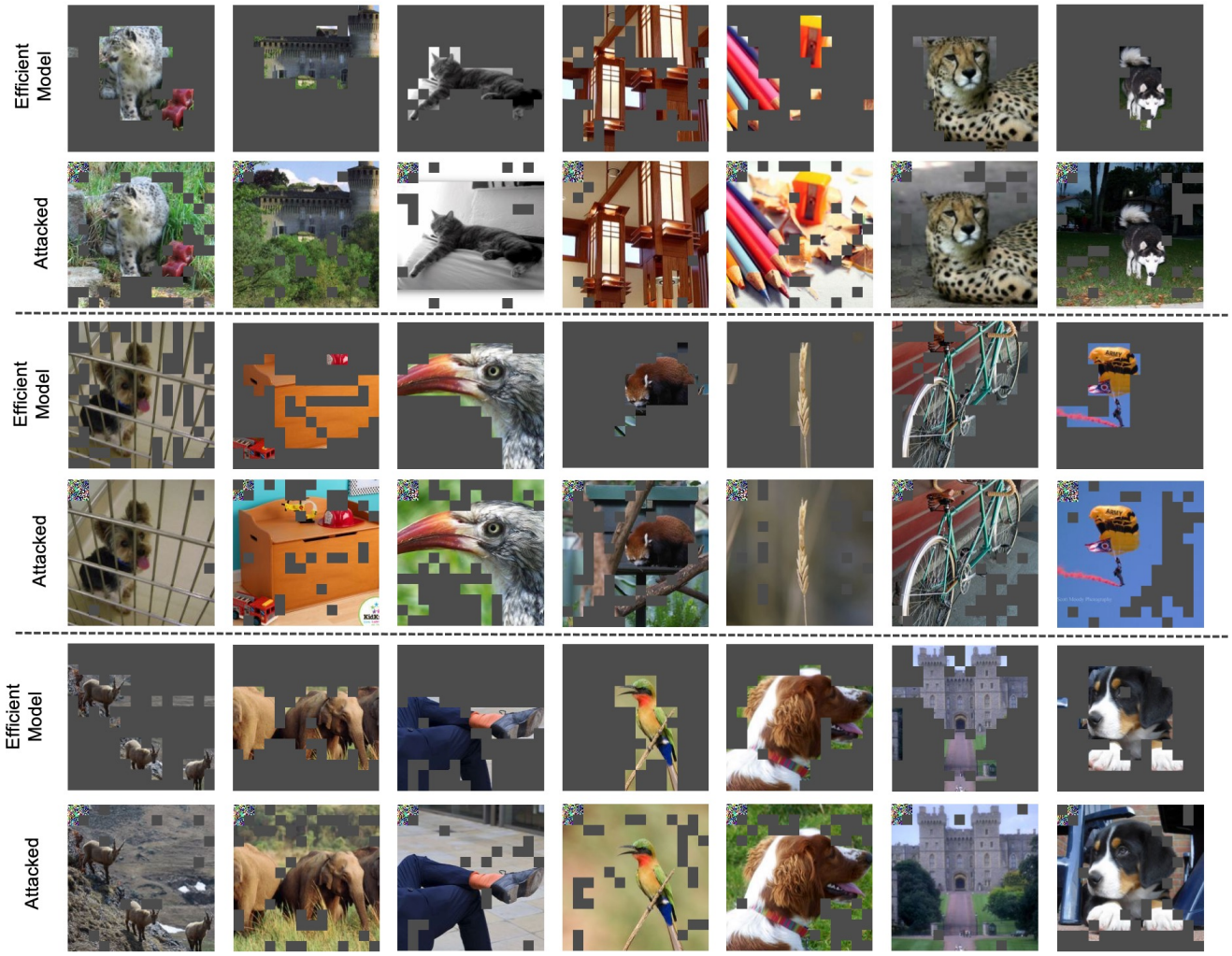


Figure 4. **Visualization of our Energy Attack on Vision Transformers:** Similar to Figure 2 of the main submission, we visualize the A-ViT-Small with and without our attack. We use patch size of 32 for the attack (on the top-left corner). We show pruned tokens at layer 8 of A-ViT-Small. Our attack can recover most of the pruned tokens, resulting in increased computation and power consumption.

patch optimization is increased to 0.8 for faster convergence.

**AdaViT Details:** We use a learning rate of 0.2 and a batch size of 128 with 4GPUs for patch optimization. We use AdamW [2] optimizer with no decay and train for 2 epochs with a patch size of 64 x 64. We train on the ImageNet-1k train dataset and evaluate it on the test set.

## References


[1] 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 *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XI*, pages 396–414. Springer, 2022. 1

[2] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. 2019. 1, 2