Patch Ranking: Token Pruning as Ranking Prediction for Efficient CLIP Supplementary Material

A. Additional Results Comparison

Comparison with CLS Attention In prior works, using CLS attention weight to rank the importance of patch tokens has been a prevalent method for enhancing the efficiency of Vision Transformers (ViTs). However, this approach is less effective for CLIP's ViT due to its dual-modality structure. Addressing this limitation, we introduce 'Patch Rank,' a novel framework tailored for CLIP's ViT. To assess the efficacy of Patch Rank, we conduct a comparative analysis with the CLS attention method across seven datasets, evaluating performance at keep rates ranging from 100% to 50%. Token pruning was executed at the first layer of CLIP's ViT to optimize computational savings. Importantly, neither method performs fine-tuning after token pruning. As shown in Figure 1, our Patch Ranking consistently demonstrates higher accuracy than CLS attention across all keep rates and datasets. Notably, our method shows a significant advantage over CLS attention, especially at lower keep rates (60% and 50%). This outcome indicates the ability of Patch Rank to precisely eliminate less informative patch tokens while minimizing the loss in accuracy, thereby affirming its effectiveness in the nature of CLIP's ViT.

B. Ablation study

Architecture of Predictor To construct our predictor, we selected three different architectures: (1) MLP, which consists of a 256-dimensional hidden layer, layer normalization, GLUE, and a 196-dimensional hidden layer; (2) Transformer, specifically a Transformer-encoder block; and (3) Mix-MLP, which is a single block configuration. To assess the performance of these architectures, we evaluated their top-100 matching rates and pruning effectiveness across various keep rates, from 80% to 50%. As depicted in Table 1, Mix-MLP emerges as the most effective, achieving the highest matching rate. Regarding the performance in token pruning, Mix-MLP demonstrates stable results across all datasets, and notably, it significantly outperforms the other architectures in the UCF101 dataset. This superiority of Mix-MLP can be attributed to its optimal capacity for learning and applying the Golden Ranking, coupled with its ability to avoid overfitting the training dataset.

Dataset	**Arch.**	**Matching rate**	**Predictor**				
			100	80	70	60	50
Caltech101	MLP	76.5	93.5	93.3	93.2	92.7	91.0
	Trans.	73.4	93.5	93.4	**93.3**	92.8	**91.2**
	Mix-MLP	**78.0**	93.5	**93.6**	93.2	**93.4**	91.0
OxfordPets	MLP	75.7	89.5	**89.2**	88.5	88.0	84.5
	Trans.	72.9	89.5	89.0	**89.0**	88.0	**85.5**
	Mix-MLP	**78.2**	89.5	88.6	88.4	**88.1**	83.5
Flowers102	MLP	69.2	70.5	69.5	**69.2**	67.4	**64.6**
	Trans.	56.2	70.5	**69.8**	69.0	67.5	60.9
	Mix-MLP	**71.9**	70.5	69.6	68.8	**67.9**	63.9
Food101	MLP	70.4	86.0	85.5	84.8	**83.7**	78.3
	Trans.	69.7	86.0	**85.7**	**85.0**	83.8	**78.5**
	Mix-MLP	**71.9**	86.0	85.1	84.2	82.8	75.8
FGVCAircraft	MLP	85.2	23.4	23.5	23.1	**23.1**	**22.8**
	Trans.	84.6	23.4	**23.6**	**23.6**	22.9	**22.8**
	Mix-MLP	**89.3**	23.4	23.2	23.2	22.9	22.4
DTD	MLP	**77.5**	45.1	44.6	44.3	43.7	41.9
	Trans.	62.1	45.1	44.5	**44.9**	**43.8**	42.1
	Mix-MLP	72.3	45.1	**45.0**	44.2	43.7	**43.0**
UCF101	MLP	77.5	67.0	61.8	58.2	53.6	43.2
	Trans.	51.2	67.0	62.2	60.0	53.5	43.5
	Mix-MLP	**79.7**	67.0	**66.6**	**65.9**	**65.2**	**60.9**

Table 1. Design Choices for the Predictor: This table explores three different architectures employed as predictors: Multilayer Perceptron (MLP), Transformer-encoder block (Trans.), and Mix-MLP. We evaluate these architectures based on their top-100 matching rates and classification accuracy across various keep rates, ranging from 100% to 50%. Token pruning is executed at the 4th layer of CLIP's ViT, aiming to assess the effectiveness of each architecture in maintaining accuracy while managing token redundancy.

Token Pruning Locations In our exploration of token pruning locations within CLIP's Vision Transformer, we conducted an in-depth analysis to determine the impact of varying pruning depths on model performance. This involved progressively pruning an equal number of patch tokens at different layers while maintaining a consistent keep rate of 60%. The results are shown in Table 2. It focuses on four distinct combinations of pruning locations, ranging from shallower to deeper layers within the

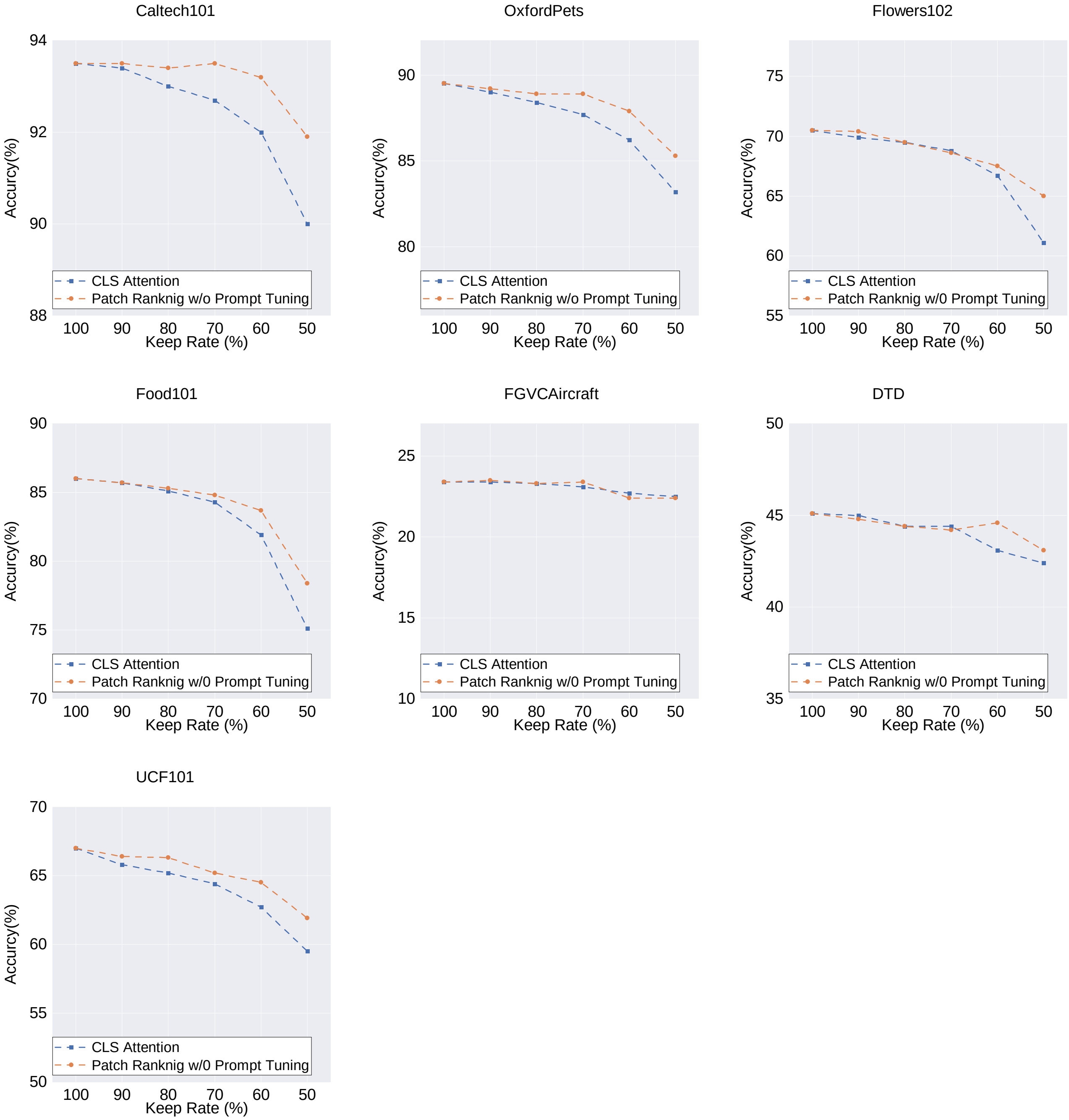

Figure 1. This figure compares the classification accuracy between the CLS attention method and our Patch Ranking approach, both without fine-tuning post-token pruning. CLS attention employs CLS attention weights to rank tokens, whereas Patch Ranking utilizes the Feature Preservation Score for this purpose. Token removal occurs at the first layer of CLIP's ViT. We present classification accuracy across different keep rates, ranging from 100% to 50%, highlighting the differential impact of each method on model performance as the number of pruned tokens increases.

network. Despite a slight margin favoring pruning patch tokens at deeper layers, the overall average performance across all datasets remains notably consistent. This suggests that our predictor can adapt to different layers within the network,

Pruning Locations	**Caltech101**	**OxfordPets**	**Flowers102**	**Food101**	**FGVCAircraft**	**DTD**	**UCF101**	**Avg.**
2, 3, 4, 5	94.4	92.3	94.3	82.0	37.9	67.6	81.7	78.6
4, 5, 6, 7	94.3	92.1	95.3	82.2	39.5	68.1	82.0	79.1
1, 3, 5, 7	94.8	91.6	94.4	82.0	39.0	67.8	81.8	78.9
4, 6, 8, 10	95.3	91.4	94.5	83.0	40.0	68.4	83.0	**79.2**

Table 2. Performance analysis across different pruning locations: In this experiment, we maintained a keep rate of 60% and progressively pruned equal quantities of patch tokens at four distinct layers within CLIP's ViT. We examined four different combinations of pruning locations to evaluate how varying the pruning layers within the network layers affects overall model performance.

accurately estimating rankings, and identifying redundant tokens across various depths. Specifically, the minimal variation in performance across different pruning configurations indicates that our approach maintains the predictor's ability regardless of the specific layers targeted for token reduction.