Supplementary Material

A. Attention Rollout Implementation

We provide additional implementation details for our method. Transformer blocks in ViT may apply multiple attention matrices through parallel heads, as well as residual connections. In these cases, we represent the attention matrix A_l as follows:

$$A'_l = \frac{1}{h}\sum_h A^h_l + I$$
$$A_l(m,n) = \frac{A'_l(m,n)}{\sum_k A'_l(m,k)} \tag{6}$$

The attention rollout $\tilde{A}$ is given by Eq. (1). In practice, instead of computing $\tilde{A}^T$ for Eq. (2) by multiplying matrices $\tilde{A}^T = (A_L \times ... \times A_2 \times A_1)^T = A_1^T \times A_1^T \times \cdots \times A_L^T$, we compute the importance vector s_0 recursively, using matrix by vector multiplications, reducing computations by a factor of N, as follows:

$$s_l = \begin{cases} A^T_{l+1} s_{l+1} & \text{if } l < L \\ s_L & \text{if } l = L \end{cases} \tag{7}$$

B. Additional Implementation Details

Pruning ViTDet. As discussed in previous works [1], token pruning in dense prediction tasks requires special considerations. To handle token pruning in ViTDet and maintain the spatial structure of the feature maps, we keep an intermediate cache with the most updated keys and values features for all layers. Our pruning then allows to save computations in all token-wise operations of the ViT and in the query to key product of the self-attention operation. We also note that in the windowed attention blocks, in order to allow batch processing, we compute the full attention computation as done in [2]. In addition, for the OD second stage, we maintain the feature map of the output layer and update after each frame the unpruned tokens. Note that while this approach requires some memory overhead, it is still $\sim 1/10$ cheaper in memory compared to [3]. In our experiments we used a threshold of 0.1 on the confidence score of predicted bounding boxes to define s_L. For Figure (2) of the main paper, 672×672 resolution, we used K values in the range $[64 - 768]$, while for Figure (6), 1024×1024 resolution, we used K values in the range $[75 - 2200]$.

Pruning ViViT. The spatial backbone of ViViT FE is followed by a small temporal transformer. As in [3], the temporal transformer is fine-tuned on the outputs of the pruned model in order to account for the change from the original model. For fine-tuning the temporal model on Kinetics-400 [4] we trained for 10 epochs with learning rate $2e-6$. For Epic kitchens [5] we traiedn for 1 epoch with a learning rate of $1e-5$. Importantly, pruning is applied only on the spatial backbone, for which no fine-tuning is applied.

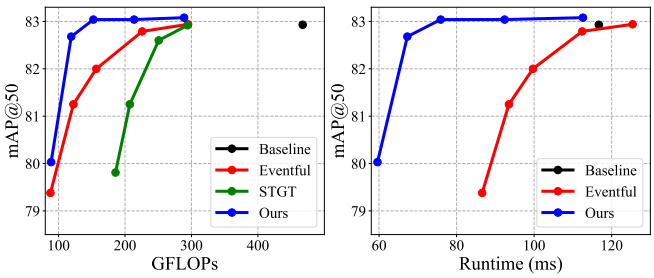

Fig. 6: Video object detection on ILSVRC 2015 ImageNet VID at 1024×1024 resolution.

Table 3: Effect of tracking window size r on ImageNet VID .

Window Size	GFLOPs	mAP50 (%) 1/1 Sampling	mAP50 (%) 1/4 Sampling
$r = 0$	46.44	81.7	77.2
$r = 1$	46.45	82.1	79.3
$r = 2$	46.47	82.1	79.5
$r = 3$	46.49	82.0	79.2

Runtime Measurements. Reported running times of VitDet are for the transformer backbone with the overheads related to rollout computation. For ViViT, runtime measurements represent the entire model, including the temporal transformer and rollout computation.

C. Additional Results on Video Object Detection

We follow common evaluation protocols on ImageNet VID and rescale input frames to either 1024×1024 or 672×672. We show results for 672×672 in Figure (2) in the manuscript, as more related works have been evaluated on this resolution. We show results for 1024×1024 in Figure (6). Our method reduces computation by 67% while improving the accuracy of the original ViTDet model. We provide additional qualitative results in Figure (7), visualizing s_0^t for both the original ViTDet and our method, as well as the pruned tokens.

D. Tracking Window Size

We provide an ablation study on the window size used for token tracking. Table 3 shows that $r = 1$ yields best results for the video clips in ImageNet VID. Larger r values should be used when increased motion is expected, as in our simulated 1/4 sampling. We note that tracking within larger windows is subject to noise and might degrade accuracy.

E. Temporal Token Tracking

Our method propagates importance scores between successive frames as defined in Eq (4). Intuitively, it can be re-

garded as a template matching in the token space. While more advanced tracking methods can be used, our motivation lies in applying them on tokens rather than on frame pixels, due to the following traits; first, as we want to relate tokens between successive frames, we seek to avoid additional propagation between tokens and pixels which might impact such relation. Second, tracking methods may require less computations when applied on tokens rather than pixels. We suggest that input tokens, derived using a linear projection on non-overlapping image patches, retain locality such that translation in the pixel space would be similarly depicted in the token space. In addition, the aforementioned projection in ViTs used in this work is $\mathbb{R}^{16\times16\times3} \rightarrow \mathbb{R}^{768}$, potentially preserving the amount of information. We note, however, that the above does not hold for subsequent token transformations. In particular, attention layers exchange information between tokens and hence do not preserve locality. Furthermore, tokens in successive frames undergo different transformations, making their latent representations incommensurate.

F. Token Pruning at Intermediate Layers

Our method estimates the importance of input tokens at frame t, z_0^t, with respect to predictions in frame $t-1$. At subsequent layers, attention rollout can also be used to propagate importance scores from s_0^t to s_l^t, $1 < l < L$. Following Eq (2), we get:

$$\begin{aligned} s_0^t &= A_{(l,1)}^T s_l^t \\ s_l^t &= (A_{(l,1)}^T)^{-1} s_0^t \end{aligned} \tag{8}$$

With Eq. (8), token pruning can be applied gradually across transformer blocks. In our experiments, we observed marginal improvement in complexity-accuracy tradeoffs, along with a runtime increase due to matrix inversion. We note that while such pruning in intermediate layers enables gradual loss of information, pruning at s_l^t is as informed as at s_0^t, as they both stem from the importance of s_L^{t-1}.

1. REFERENCES

[1] Y. Liu, M. Gehrig, N. Messikommer, M. Cannici, and D. Scaramuzza, "Revisiting token pruning for object detection and instance segmentation," 2023.

[2] S. Sarkar, G. Datta, S. Kundu, K. Zheng, C. Bhattacharyya, and P. A. Beerel, "Maskvd: Region masking for efficient video object detection," *arXiv:2407.12067*.

[3] M. Dutson, Y. Li, and M. Gupta, "Eventful transformers: Leveraging temporal redundancy in vision transformers," in *ICCV*. IEEE, 2023, pp. 16911–16923.

[4] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman, "The kinetics human action video dataset," *arXiv:1705.06950*, 2017.

[5] D. Damen, H. Doughty, G. M. Farinella, A. Furnari, E. Kazakos, J. Ma, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray, "Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100," *Int. J. Comput. Vis.*, vol. 130, pp. 33–55, 2022.

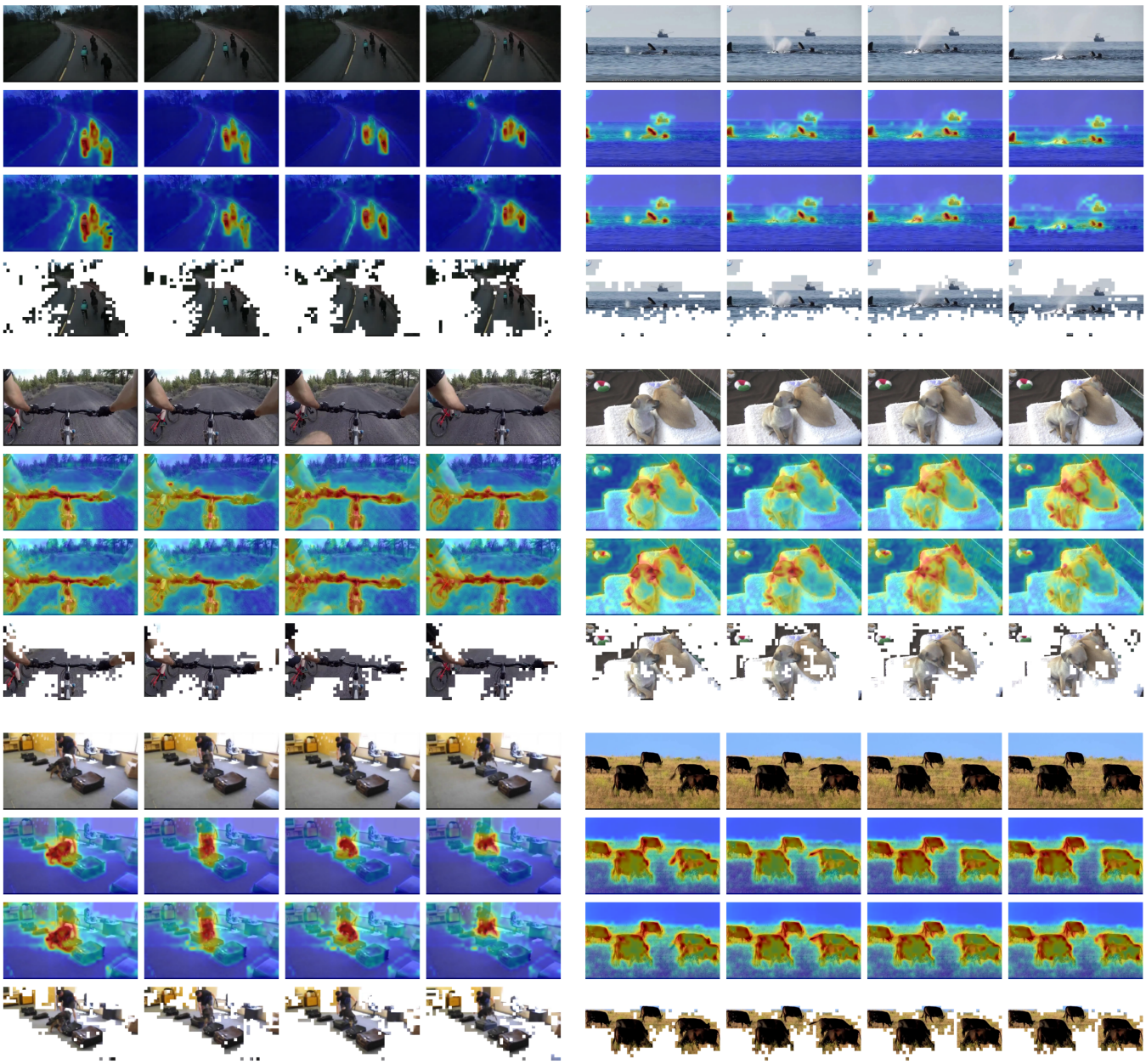

Fig. 7: Additional qualitative results on ImageNet VID. The four rows for each video clip show input frames x_t, s_0^t of the original model, s_0^t of our method and pruned tokens by our method, respectively.