Eventful Transformers: Leveraging Temporal Redundancy in Vision Transformers Supplementary Material

Matthew Dutson, Yin Li, and Mohit Gupta
University of Wisconsin–Madison
{dutson,yin.li,mgupta37}@wisc.edu

We use capital letters (*e.g.*, Figure A) to refer to the supplementary material and numbers (*e.g.*, Figure 1) to refer to the main paper.

In Section A, we provide further discussion on token selection policies, optimizations to the query-key product, and the ViViT temporal model. In Section B, we present additional experiments: action recognition on Kinetics-400, an evaluation of a threshold policy, and an ablation of the gate position. In Section C, we provide low-level details for the experiments in the main paper. In Section D, we include tables of results for the experiments in the main paper.

A. Further Discussion

The ViViT temporal sub-model. Recall that, for ViViT action recognition, we fine-tune the non-Eventful temporal model on the outputs of the Eventful spatial model. We now provide some intuition as to why this is necessary to preserve the prediction accuracy.

The outputs of an Eventful Transformer are approximations of the "correct" outputs (those of the original, non-Eventful Transformer). In the case of the ViViT spatial model, individual outputs are fairly close to the correct values. However, the *pattern of temporal changes* between outputs may be quite different from the original model. Token gates reduce the number of updated tokens on each frame, but each update tends to be larger (a single update may contain accumulated changes from several time steps). Given the nature of the prediction task – action recognition on highly dynamic videos – the temporal sub-model is sensitive to the pattern of temporal changes. Fine-tuning allows us to correct for the shifts in these temporal changes that result from using an Eventful spatial model.

Compatibility with spatial redundancy methods. We now provide further discussion regarding the compatibility of our method with spatial redundancy approaches. Abstractly, we can think of spatial redundancy methods as summarizing a set of tokens $\boldsymbol{x} \in \mathbb{R}^{N \times D}$ using a reduced set of tokens $\hat{\boldsymbol{x}} \in \mathbb{R}^{M \times D}$. The simple method in our experiments summarizes tokens using uniform pooling; however, we could also use adaptive pruning or merging.

Assume we apply a gate to the reduced tokens $\hat{\boldsymbol{x}}$. The gate assumes that the definitions of its input tokens are relatively stable. This assumption clearly holds for non-reduced or uniformly pooled tokens. However, we need to be careful when applying arbitrary reductions to $\boldsymbol{x}$.

For example, say we have an image containing a region of blue sky. An adaptive token merging method might combine all sky-colored tokens from $\boldsymbol{x}$ into a single token in $\hat{\boldsymbol{x}}$. Assume that on frame $t = 1$, the first token in $\hat{\boldsymbol{x}}$ represents the sky. Ideally, on frame $t = 2$, the first token in $\hat{\boldsymbol{x}}$ should again represent the sky. Note that this is not a strict constraint – our gating logic can deal with non-consistent definitions for a few tokens. However, if the definitions for all tokens in $\hat{\boldsymbol{x}}$ completely change between frames, then the gate will not be able to keep up (*i.e.*, the number of tokens with significant changes will exceed the policy r-value).

B. Additional Experiments

Video action recognition on Kinetics-400. We evaluate our method on the Kinetics-400 action recognition dataset [2]. Kinetics-400 contains over 300k video clips, each annotated with one of 400 action categories. We evaluate top-1 accuracy. We use the same ViViT model architecture as in our EPIC-Kitchens experiments; the only difference is the input size (224×224 rather than 320×320).

As in our EPIC-Kitchens experiments, we fine-tune the non-Eventful temporal model on the outputs of the Eventful spatial model. We fine-tune three variants of the model with r =24, 48, and 96 (out of a maximum of 197 tokens). We train for 10 epochs on a subset of the training set containing 39729 videos. We use the AdamW optimizer [4] with a learning rate of 2×10^{-6}, weight decay of 0.05, and a batch size of 16 videos. We add 50% dropout before the final classification layer.

Table A shows our results. The accuracy-compute

Table A. **Kinetics-400 video action recognition.** Results for Kinetics-400 action recognition using the ViViT model. We report the total TFlops per video (spatial + temporal sub-models).

Variant	r	Accuracy (%)	TFlops
Base model	–	79.06	3.360
Temporal	96	77.62	1.814
Temporal	48	75.88	1.016
Temporal	24	75.16	0.618

Table B. **A threshold policy.** Results for a threshold policy with the 1024-resolution ViTDet model. The policy selects tokens where the error $\boldsymbol{e}$ exceeds a threshold h.

Variant	h	mAP50 (%)	GFlops
Base model	–	82.93	467.4
Temporal	0.2	83.00	431.8
Temporal	1.0	82.75	294.1
Temporal	5.0	78.11	133.5

tradeoff is generally consistent with our results on EPIC-Kitchens. For example, with $r = 96$, we sacrifice 1.48% accuracy for a speedup of approximately 2x.

A threshold policy. We evaluate the ViTDet object detection model with a threshold policy. The threshold policy selects all tokens where the L2 norm of $\boldsymbol{e}$ exceeds a threshold h. We test h = 0.2, 1.0, and 5.0. See Table B for results. The accuracy-compute tradeoff for the threshold policy is generally worse than for the top-r policy. For example, compare threshold h = 5.0 with r = 512 in Table C. This is likely due to the use of a constant threshold for all gates (we would ideally use a unique threshold for each gate).

C. Experiment Details

Fine-tuning ViTDet for VID. We initialize our model using COCO [3] pre-trained weights, and then trained on a combination of the ImageNet VID and ImageNet DET datasets, following common protocols in [1, 5]. We select images from the DET dataset that are of of the same 30 classes as in the VID dataset. The training uses a batch size of 8, a maximum input resolution of 1024×1024, an initial learning rate of 10^{-4}, and a weight decay of 0.1. We use the AdamW optimizer [4] with linear warmup for a total of 5 epochs, with 10x learning rate decay from the 3rd epoch.

Fine-tuning the ViViT temporal model. We fine-tune the temporal sub-model for 5 epochs. We use the AdamW optimizer [4] with a learning rate of 10^{-5}, weight decay of 0.05, and a batch size of 8 videos. We add 50% dropout before the final classification layer.

Arithmetic precision. We compute the product $A\boldsymbol{v}$ at half precision in the global self-attention operators of the Eventful model. Using half precision reduces the model's computational cost and memory footprint and has a negligible effect on accuracy. When evaluating runtimes, we also compute $A\boldsymbol{v}$ at half precision in the base model (this ensures a fair comparison).

Runtime experiments. For ViTDet, we evaluate CPU runtimes using one random video from VID (ID 00023010, containing 242 frames). On the GPU, we use 5 random videos. For ViViT, we evaluate CPU runtimes using 5 random videos from EPIC-Kitchens. On the GPU, we use 100 random videos. We use a consistent random seed across all experiment runs.

Operation counting. Our GFlop counts include the following types of operations: linear transforms, matrix multiplications, einsum operations (used in relative position embeddings), and additions. We count a multiply-accumulate as a single operation. In Eventful Transformers, we additionally count operations required for updating the gate (additions and subtractions) and the extra additions in the sparse attention-value update. We only report operations in the Transformer backbones (*e.g.*, we do not count anything in the object detection head).

D. Result Tables

In this section, we provide tables of results for experiments in the main paper. Table C corresponds to Figures 7 and 8, and Table D corresponds to Figure 9. Table E shows spatial redundancy results for the 672-resolution ViTDet model (the 1024-resolution results are in Table 1).

References

[1] Yihong Chen, Yue Cao, Han Hu, and Liwei Wang. Memory enhanced global-local aggregation for video object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[2] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The Kinetics human action video dataset. arXiv, 2017. 1

[3] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–755, Cham, 2014. Springer International Publishing. 2

[4] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 1, 2

[5] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. Flow-guided feature aggregation for video object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 408–417, 2017. 2

Table C. **Video object detection results.** Results for video object detection on VID using the ViTDet model. This table corresponds to Figures 7 and 8 in the main paper.

Size	Variant	r	mAP50 (%)	GFlops
1024	Base model	–	82.93	467.4
1024	Our method	2048	82.94	294.9
1024	Our method	1536	82.79	225.9
1024	Our method	1024	82.00	156.8
1024	Our method	768	81.25	122.3
1024	Our method	512	79.38	87.8
1024	Our method	256	73.29	53.3
1024	Token-wise only	2048	82.97	294.1
1024	Token-wise only	1536	82.93	250.7
1024	Token-wise only	1024	82.58	207.3
1024	Token-wise only	768	82.08	185.7
1024	Token-wise only	512	81.11	164.0
1024	Token-wise only	256	76.60	142.3
1024	STGT	2048	82.92	294.1
1024	STGT	1536	82.60	250.7
1024	STGT	1024	81.25	207.3
1024	STGT	768	79.81	185.7
1024	STGT	512	76.70	164.0
1024	STGT	256	68.73	142.3
672	Base model	–	82.28	174.5
672	Our method	1024	82.23	115.1
672	Our method	768	82.21	87.9
672	Our method	512	81.84	60.7
672	Our method	384	81.43	47.1
672	Our method	256	80.16	33.5
672	Our method	128	75.19	19.9
672	Token-wise only	1024	82.28	111.9
672	Token-wise only	768	82.25	90.2
672	Token-wise only	512	82.01	68.5
672	Token-wise only	384	81.64	57.7
672	Token-wise only	256	80.76	46.8
672	Token-wise only	128	76.96	36.0
672	STGT	1024	82.28	111.9
672	STGT	768	81.95	90.2
672	STGT	512	80.45	68.5
672	STGT	384	78.71	57.7
672	STGT	256	75.57	46.8
672	STGT	128	68.13	36.0

Table D. **Video action recognition results.** Results for video action recognition on EPIC-Kitchens using the ViViT model. This table corresponds to Figure 9 in the main paper.

Variant	Tuned r	Tested r	Accuracy (%)	TFlops
Base model	–	–	67.14	7.12
Temporal	200	280	66.77	5.49
Temporal	200	240	66.53	4.77
Temporal	200	200	66.02	4.05
Temporal	200	160	64.72	3.33
Temporal	200	120	62.23	2.62
Temporal	100	140	65.52	2.98
Temporal	100	120	64.51	2.62
Temporal	100	100	62.91	2.26
Temporal	100	80	60.76	1.90
Temporal	100	60	59.13	1.54
Temporal	50	70	61.27	1.72
Temporal	50	60	60.60	1.54
Temporal	50	50	59.91	1.36
Temporal	50	40	58.90	1.18
Temporal	50	30	58.05	1.00

Table E. **Adding spatial redundancy to 672-resolution ViTDet.** Results for adding spatial redundancy to the 672-resolution ViTDet model. 1024-resolution results are in the main paper.

Variant	r	mAP50 (%)	GFlops
Base model	–	82.28	174.5
Spatial	–	79.86	159.7
Spatiotemporal	1024	79.85	98.2
Spatiotemporal	768	79.81	75.5
Spatiotemporal	512	79.47	52.8
Spatiotemporal	384	79.02	41.4
Spatiotemporal	256	77.90	29.8
Spatiotemporal	128	73.40	18.0