Latency Matters: Real-Time Action Forecasting Transformer (Supplementary)

Harshayu Girase[1,2‡*] Nakul Agarwal[1] Chiho Choi[1*†] Karttikeya Mangalam[2‡§]

[1]Honda Research Institute USA [2]UC Berkeley

1. Datasets

In this work, we explore the widely used EPIC-KITCHENS dataset [1], an unscripted dataset with nearly 20x more action classes and $10-100$x more observed sequences than other action datasets such as Breakfast dataset [6] and 50Salads [7] dataset.

We benchmark the EPIC-KITCHENS dataset on both the EPIC-55 and EPIC-100 anticipation splits and use the same training/testing split as in prior works [3, 4, 10, 11]. The EPIC-55 dataset consists of 39,600 segments split up from raw, unscripted videos of humans performing 2,513 actions in the kitchen. The EPIC-100 dataset contains close to 90,000 segments with 3,806 actions. For EK55, we use the Top-1 accuracy metric which was the primary metric used by most works on this dataset. For EK100, we use the Top-5 Action Recall metric to compare performance as done in most prior works for this dataset.

We also report results on the EGTEA+ Gaze dataset [8] which contains 106 actions from over 10,000 segments. Note that we use the same training/testing split (5-fold cross validation average) as reported in prior baselines/works [3]. We report with anticipation horizon, $t_f = 1$ so it is the same for all our datasets in the offline setting.

2. Implementation Details

2.1. MViT Backbone

We perform our own feature extraction using the MViT=B [2, 9] backbone. For the K400+IN1K model, we use the 16x4 model pre=trained for short-term action recognition on the Kinetics-400 dataset. The 16x4 model uses 16 frames sampled 4 frames apart at a 30fps. This translates to each clip being of length 2 seconds sampled at 8fps. We sample directly from the RGB frames provided in the EPIC-KITCHENS-100 dataset and do not use flow or object features that few other works such as [3, 4]. Our K700 model uses a different 32x3 pre-trained MViT on the K700 dataset. This model uses 32 frames sampled 3 frames apart at a 30fps; while the performance is better due to the heavier model architecture, the inference time is slower. This trade-off is explored in detail in the main paper.

2.2. Data Details

We find that proper data pre-processing is critical to model performance. We use a center crop of 224x224 and normalize the images after cropping to $mean = [0.45, 0.45, 0.45]$ and $STD = [0.25, 0.25, 0.25]$ following the scheme in MViT [2]. Note that techniques such as image augmentations, random cropping, multi-crop evaluation would boost performance during both inference and training. However, we do not perform augmentations on our data due to GPU limitations making our training recipe much faster in practice; training end-to-end with augmentations however would further increase our performance as well. For example [4] uses 3-crop testing which significantly improves performance. Our proposed method does **not** require end-to-end training and is suitable for faster experimentation iterations and those with GPU constraints. We use an observed sequence length of 16-18 seconds which is longer than previous works by 6 seconds but half as long as MemViT's model [11]. This however helps our model to be lower latency than SOTA model MemViT.

2.3. Training Hyperparameters

We train our model with the AdamW optimizer with momentum 0.8 and weight decay of 0.001. We use a batch size of 512 and AdamW optimizer for training. We train for 75 epochs using a cosine scheduler with a warm-up of 30 epochs. Our base learning rate is $1e$-4 and end learning rate is $8e$-7. We use a dropout of 0.25 for the transformer and a dropout of 0.1 for both the inputs and MLP feature/action heads.

3. Results

3.1. Latency Evaluation

In the main paper, latency values are calculated over an average of many samples rather than a single instance.

* Work done during Harshayu's internship at Honda Research Institute with Chiho Choi's supervision

† Now at Samsung Semiconductor US

‡ Equal technical contribution

§ Corresponding Author

Figure 1. This figure shows qualitative examples in which RAFTformer predicts the next action correctly, but AVT [4] is incorrect. Note that we show Top-5 predictions. The frames and their corresponding labels show the past video and corresponding actions (from 16s ago to the present).

3.2. Additional Baselines

In Tables 1 and 2 we present results on EK55 dataset (Top-1 Accuracy), using flow and object features as input. Note that this is the standard offline setting. We show that even with these modalities (which is not used for our final model), we outperform prior models. Note that the flow and object features are the same used in [3]. All performance improvements seen in these tables are due to the RAFTformer Head architecturer (Anticipation Tokens and Shuffled Causal Masking), since the backbone input features are the exact same (frame-wise TSN features [3]).

Figure 2. Two examples of failure cases of RAFTformer. We can see that there are some cases where even RAFTformer has a lot of trouble understanding context and what exactly is going on. In the first image, a tap is not clearly visible which is why RAFTformer may fail to predict actions that involve a tap.

Model	Top-5 Recall
RULSTM [3]	7.8
AVT [4]	8.7
RAFTformer	**9.5**

Table 1. Results of our model compared to others on flow features provided by [3]. Note that all models used the same input features, but we still outperform prior SOTA. Results are evaluated using the same train/val split [3, 4] which we refer to as the offline setting ($t_f = 1s$)

Model	Top-5 Recall
RULSTM [3]	7.2
AVT [4]	6.7
RAFTformer	**7.5**

Table 2. Results of our model compared to others on object features provided by [3]. Note that all models used the same input features, but we still outperform prior SOTA. Results are evaluated using the same train/val split [3,4] which we refer to as the offline setting ($t_f = 1s$).

3.3. Shuffling Probability Ablation

Shuffling ablation is shown in Figure 3, where we observe see the U-curve trend between performance and shuffling probability. Additionally, we also find that using just the reverse order (so fixed order but in reverse, no shuffling) in input sequence yields no change in T5R. Note that under a fixed π^* (such as reversed), APE learns the required temporal positioning information, thus preserving T5R. With a randomly sampled and changing $\pi^* \sim \pi$, the additional self-supervised task of predicting unseen clip features boosts model performance over a fixed π. Specifically, we find 0.3 to be optimal gaining 0.7 T5R points over doing no shuffling at all.

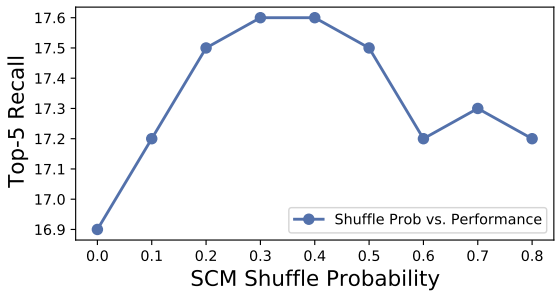

Figure 3. **Effect of SCM shuffling**. We ablate the effect of shuffling probabiltiy on the final T5R performance on EK100.

3.4. Qualitative Results

In figure 1, we visualize multiple examples where AVT has incorrect predictions, but RAFTformer is able to correctly predict the next action. In figure 2, we show cases where RAFTformer has incorrect predictions.

4. Limitations

In this work we focus solely on the short-term anticipation setting. Our model as is would not work directly for rolling-out multiple steps into the future which could be interesting for longer-term planning [5]. Furthermore, reasoning about longer-term horizons raises more questions about different types of multimodality that require additional modeling. Another limitation is that our model is tested on egocentric videos of a single human acting in an

environment. While this is indeed applicable for human-robot interaction and robot understanding, testing in even more complex interactive environments would be an interesting future direction.

References

[1] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *ECCV*, pages 720–736, 2018. 1

[2] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. *arXiv preprint arXiv:2104.11227*, 2021. 1

[3] Antonino Furnari and Giovanni Maria Farinella. Rolling-unrolling lstms for action anticipation from first-person video. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):4021–4036, 2020. 1, 2, 3

[4] Rohit Girdhar and Kristen Grauman. Anticipative video transformer. In *ICCV*, pages 13505–13515, 2021. 1, 2, 3

[5] Dayoung Gong, Joonseok Lee, Manjin Kim, Seong Jong Ha, and Minsu Cho. Future transformer for long-term action anticipation. In *CVPR*, pages 3052–3061, 2022. 3

[6] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *CVPR*, pages 780–787, 2014. 1

[7] Colin Lea, René Vidal, and Gregory D Hager. Learning convolutional action primitives for fine-grained action recognition. In *ICRA*, pages 1642–1649. IEEE, 2016. 1

[8] Yin Li, Miao Liu, and Jame Rehg. In the eye of the beholder: Gaze and actions in first person video. *IEEE transactions on pattern analysis and machine intelligence*, 2021. 1

[9] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Improved multiscale vision transformers for classification and detection. *arXiv preprint arXiv:2112.01526*, 2021. 1

[10] Fadime Sener, Dipika Singhania, and Angela Yao. Temporal aggregate representations for long-range video understanding. In *ECCV*, pages 154–171. Springer, 2020. 1

[11] Chao-Yuan Wu, Yanghao Li, Karttikeya Mangalam, Haoqi Fan, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Memvit: Memory-augmented multiscale vision transformer for efficient long-term video recognition. *arXiv preprint arXiv:2201.08383*, 2022. 1