Latency Matters: Real-Time Action Forecasting Transformer

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

[*] denotes equal technical contribution

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

Abstract

We present RAFTformer, a real-time action forecasting transformer for latency-aware real-world action forecasting. RAFTformer is a two-stage fully transformer based architecture comprising of a video transformer backbone that operates on high resolution, short-range clips, and a head transformer encoder that temporally aggregates information from multiple short-range clips to span a long-term horizon. Additionally, we propose a novel self-supervised shuffled causal masking scheme as a model level augmentation to improve forecasting fidelity. Finally, we also propose a novel real-time evaluation setting for action forecasting that directly couples model inference latency to overall forecasting performance and brings forth a hitherto overlooked trade-off between latency and action forecasting performance. Our parsimonious network design facilitates RAFTformer inference latency to be $\mathbf{9}\times$ *smaller than prior works at the same forecasting accuracy. Owing to its two-staged design, RAFTformer uses* $\mathbf{94}\%$ *less training compute and* $\mathbf{90}\%$ *lesser training parameters to outperform prior state-of-the-art baselines by* $\mathbf{4.9}$ *points on EGTEA Gaze+ and by* $\mathbf{1.4}$ *points on EPIC-Kitchens-100 validation set, as measured by Top-5 recall (T5R) in the offline setting. In the real-time setting, RAFTformer outperforms prior works by an even greater margin of upto* $\mathbf{4.4}$ *T5R points on the EPIC-Kitchens-100 dataset. Project Webpage:* https://karttikeya.github.io/publication/RAFTformer/.

1. Introduction

Latency matters. It is a crucial system design consideration for countless applications that operate in real-time from hardware design [65], network engineering [63], and satellite communications [30] to capital trading [32], human vision [59] and COVID transmission patterns [54]. However, it has not been a center stage design consideration in modern computer vision systems of the past decade [11,45]. Modern vision system design has largely focused on the correctness of systems rather than the latency of the predictions. While vision-based forecasting systems are often meant for embodied real-time deployment on autonomous agents like self-driving cars and robots, they are evaluated in an *offline* setting where inference latency is neglected (Figure 1). Interestingly, recent neural network architectures have adopted FLOPs as a *proxy* for latency as a *second* axis for model design. While a sufficient fidelity metric for offline after-the-fact applications like automatic content recognition, latency often comes second to correctness, even for real-time systems such as forecasting models.

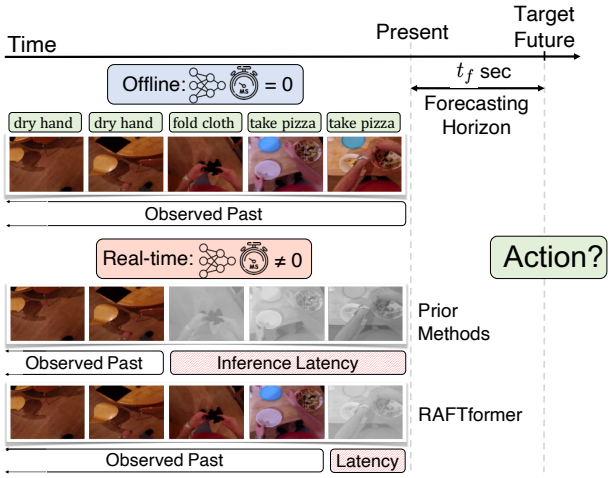

Figure 1. **Action Forecasting** is the task of predicting actions that will happen after a pre-determined time span, say t_f seconds, into the future. Prior works consider an offline evaluation setting that ignores the model inference latency. We propose a latency-aware real-time evaluation setting where the model is required to *finish* forecasting t_f seconds before the target time. We present RAFTformer, a fast action anticipation transformer that outperforms prior works both in offline & real-time setting while forecasting actions in real-time ($\geq$ 25 FPS).

Forecasting empowers reactive planning [17]. An autonomous system present in rich human environments inevitably needs to understand human actions around it for smooth task planning and execution. Autonomous agent planning critically depends on anticipating the future of the scene in various forms such as trajectory prediction [22, 23, 57, 58], action forecasting [19, 25, 80] or future scene

[*] Work done during Harshayu's internship at HRI with Chiho Choi's supervision who is now at Samsung Seminconductor US
Karttikeya Mangalam is the corresponding author

segmentation [8] and anticipating the future is an activity humans subconsciously do for day-to-day tasks [60]. And while vision-based forecasting systems are often meant for embodied real-time deployment on autonomous agents like autonomous cars and robots, they are evaluated in an *offline* setting where inference latency is neglected (Figure 1).

In this work, we propose a *real-time* evaluation setting (Figure 1) that closely mimics the real-world deployment for a forecasting system. Suppose that in a real-time system, the design specifications require the forecasting system outputs t_f seconds in advance of the event to be able to plan and use the forecasts effectively. In current offline settings, the forecasting system begins the inference t_f seconds in advance of the event ('Present' in Figure 1) and the model latency is ignored (or assumed to be 0) such that the predictions are available instantly. However, in our proposed *real-time* setting, the model is required to start inference in advance of 'Present' so that the outputs are available with a horizon of t_f seconds, meeting the design specification.

We observe that in the real-time setting, the prior works fare quite poorly because of their slow model inference latency (Table 3). A large latency implies that the model has to start inference further in the past and has to rely on older video data to make forecasts with the benefit of more expressiveness (Figure 2). A smaller latency means the model can enjoy more recent video data but has limited capacity. Simply said, models that are only evaluated in the offline setting may fare poorly in the real-time deployment setting due to their latency agnostic design (Figure 2).

We present, RAFTformer, a real-time action forecasting transformer that uses a two-stage transformer encoder-based network for lightning fast forecasts in inference. RAFTformer uses a shuffled casual masking scheme based feature prediction loss for learning strong temporal cues that transfer to feature prediction. Further, RAFTformer uses specialized anticipation tokens for learning to predict action at multiple temporal horizons that improve model reasoning capabilities of short-term action forecasting as well. Finally, the model is explicitly designed for real-time embodied deployments that allows inference up to an order of magnitude faster than prior state-of-the-art methods. In summary, our contributions are three-fold,

First, we propose Real-time Action Forecasting Transformer (RAFTformer), a real-time action forecasting transformer with latency at least $9\times$ smaller than prior state-of-the-art action forecasting methods. RAFTformer uses specialized anticipation tokens and a novel shuffled casual masking-based self-supervision loss that allows it to outperform prior work while maintaining low latency with a reduction of 94% in GPU training time and 90% in the number of trainable parameters compares to prior works. To the best of our knowledge, our work is the first to achieve action anticipation in real-time (*i.e.* 25 fps).

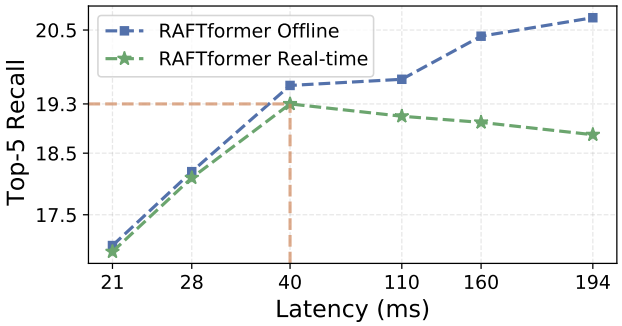

Figure 2. **Evaluation Performance *vs*. Latency.** Bigger models perform better in latency agnostic offline settings. In the real-time evaluation setting, we observe that, beyond a limit, bigger models with higher latency cause a drop in forecasting performance. In practical deployment, there exists a trade-off between latency and high-fidelity forecasts. See §4.3.1 for details.

Second, we propose a latency-aware real-time evaluation setting (Figure 1) that better mimics practical deployment settings for embodied forecasting systems. Real-time evaluation demonstrates a clear trade-off between inference latency and model forecasting fidelity, paving the path for the development of latency-aware forecasting models in the future (also see [20]).

Third, Through extensive experiments, we show that RAFTformer outperforms prior state-of-the-art methods by 4.9 points on the EGTEA Gaze+ dataset, by 1.4 points on the EPIC-Kitchens-100 dataset according to the Top-5 Recall metric and by a relative margin of 5.3% on the top-1 accuracy metric on EPIC-Kitchens-55 dataset.

2. Related Work

Action Anticipation. Over the last few years, action anticipation has seen significant advances following the promising results in video recognition [2,4,12,15,24,38,40,49,55,61,80,87] and video segmentation [27,43,48,76,81]. While earlier works [1, 56, 68] use CNN-based methods for video action anticipation, many follow-up works transitioned into using recurrent sequence-based networks [19, 22, 64, 66, 71]. Using multiple spatial and temporal scales was another key idea explored in several anticipation works [15, 70, 70, 77, 80, 86]. Masking-based self-supervision has proven to be a new frontier for both image [5, 14, 33, 35, 62] and video [31, 74] representation learning. This concept has also been explored in the context of video anticipation [25] which differs from prior works that explore temporal consistency [16, 37, 42, 79, 82], inter-frame predictability [28, 29, 36], and cross-modal correspondence [3, 44, 73]. In our work we propose a novel generalized self-supervision scheme which is a crucial part of our model's ability to generalize and outperform SOTA.

Transformers for Video Anticipation. With the rise of transformers for sequence tasks, recent works have explored various image and video-based transformers [25, 26, 26, 78, 80, 85] for anticipating actions. [25] proposes a ViT-based [11] spatial transformer backbone with a transformer decoder head to anticipate the next action with a $1s$ horizon. [25] predicts actions in the same latent space as their feature decodings causing conflation between feature reconstruction and future prediction. We propose specialized anticipation tokens to address this problem. Furthermore, [25] uses a recurrent decoder and frame-level operation, which is inefficient for both training compute and inference latency. MeMViT [80] proposes another fully transformer model which extends MViT [12] for better long-range modeling using a novel caching mechanism. However, their model is trained to only predict the next action and thus does not explicitly learn to model the future or evolving scene dynamics. Furthermore, their model is less efficient than RAFTformer in terms of both training and inference time due to its larger model and input size and complete end-to-end training. RAFTformer uses a fixed pre-trained video network and only trains the RAFTformer network. Recently [26] propose a transformer-based model for long-term action anticipation. They focus on a sequential prediction rather than next-action prediction and do not focus on inference latency.

Real-Time Systems. Latency matters, especially for real-time applications. Autonomous agents need to reason about nearby agents and take real-time decisions. This vision has led to great progress in the development of real-time systems in the related fields of semantic segmentation [13], video object segmentation [76], object detection [7, 51, 53, 67], multi-object tracking [41]. While some progress has also been made in activity understanding [72, 84], it is limited to recognition and detection. Keeping this in mind, some recent works in action anticipation have also started focusing on efficiency and memory footprint and report training time, inference time, and trainable parameters [80]. Although a step in the right direction, these works have high inference times in comparison to their desired anticipation horizon. Our proposed method is significantly faster during both training and inference, has fewer trainable parameters and a smaller memory footprint, and still outperforms state-of-the-art methods.

3. Real-Time Action Forecasting Transformer

In this section, we first present the problem formulation (§3.1), followed by the architectural details of the video backbone for feature extraction (§3.2) and the RAFTformer model architecture (§3.3) including anticipation tokens (§3.3.1), shuffled causal masking (§3.3.2) and permutation encodings (§3.3.3), finishing with an outline of the loss functions used for training the model (§3.5).

3.1. Problem Formulation

Given an observed video starting from time $T = 0$ and of arbitrary length t, $V_{0,t} = [F_0, ..., F_t]$ where F_i denotes the frame at time i, the task is to predict the future action t_f seconds in the future, *i.e.*, action A_{t+t_f} at time $T = t + t_f$.

3.2. Pre-trained Video Backbone

In contrast to state-of-the-art models [25, 80] that train end-to-end, we demonstrate that a two-stage training process is both efficient and produces high-fidelity forecasts. First, we split the full duration of the past video $V_{0,t}$ into sub-clips $V = [C_0, C_1, ..., C_N]$ in a sliding window fashion. Each clip, C_i, is independently processed with the short-term video backbone like MViT [12] to extract clip-level features. Previous works, such as [25], have employed image backbones for comparable undertakings, while we contend that video backbones offer richer spatio-temporal features, thus benefiting the downstream task. Further, a video backbone also allows lower latency since the produced clip embeddings already contain fused features for several images at once.

Our two-stage design also allows for hierarchical temporal processing of the long-form past video. The short-term recognition backbone model operates on short, high-resolution clips. These extracted features are then used as input to the head network to process longer-range lower resolution features that capture key information from each clip. This is a crucial design consideration to lower the inference latency in RAFTformer while still capturing longer-range temporal dependencies compared to [19, 25].

3.3. RAFTformer Model Network

We propose RAFTformer as a transformer encoder model. The transformer encoder has the advantage of being able to effectively learn across clip dependencies by attending over independently extracted clip features without facing memory bottlenecks such as faced in LSTM encoders [34] that have been used in some prior works [19,70]. However, to make the encoder effective for action forecasting on pre-trained features, we propose several changes to the training process to allow two-staged training to work as well as end-to-end training for action forecasting.

3.3.1 Anticipation Tokens

The extracted clip embeddings $[C_0 \cdots C_{N-1}]$ form the first part of the input to the action anticipation head. For the second part, we propose to train learnable 'anticipation tokens' that can aggregate global context and later can be decoded into future predictions. This design choice stands in contrast to prior works like [25] where the output of the anticipation token is implicitly designed to be in the same latent space as the output of their image-feature tokens. In

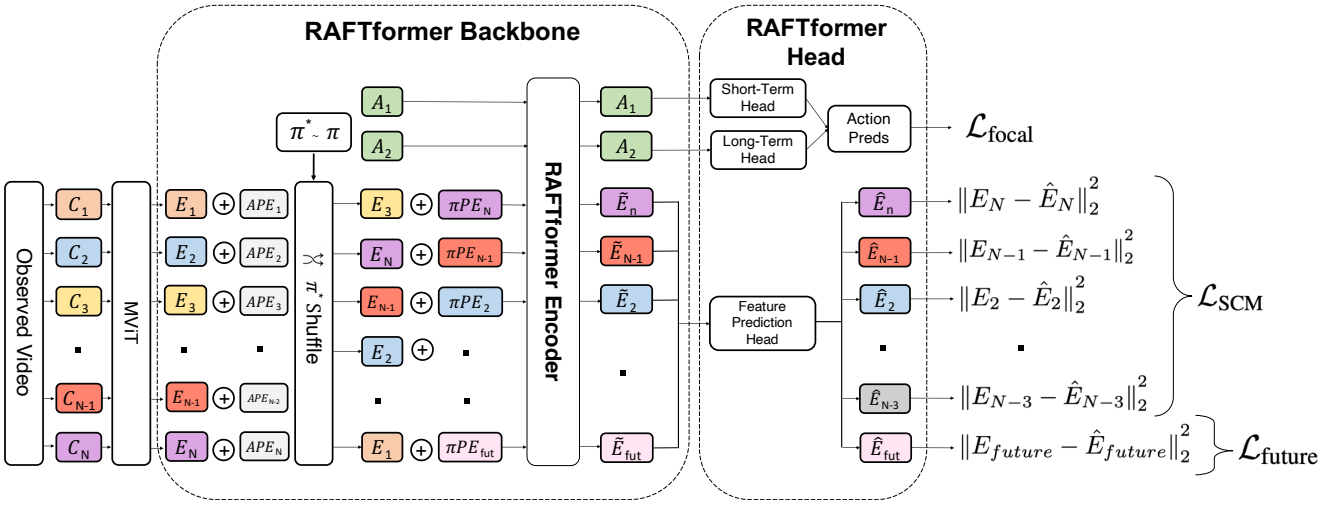

Figure 3. **RAFTformer** is a real-time action anticipation transformer architecture comprised of two stages. The first stage is a pre-trained short-term backbone that produces individual clip embeddings independently of other clips (§3.2). In the second stage (§3.3), absolute position encodings are added and the resulting sequence is shuffled with a sampled permutation $\pi^* \sim \pi$. The permutation encodings (§3.3.3) are then added to the shuffled sequence and after concatenation with anticipation tokens, the sequence is processed with the RAFTformer encoder. The output tokens are decoded via short & long-term action anticipation heads (§3.4), as well as the feature prediction head trained with self-supervised loss ($\mathcal{L}_{\text{SCM}}$), future feature prediction loss ($\mathcal{L}_{\text{future}}$) and action forecasting loss ($\mathcal{L}_{\text{focal}}$) (§3.5).

contrast to prior works, we find that temporal aggregation such as mean pooling for forecasting leads to subpar performance (see Table 5). Hence, we use the output tokens corresponding to $[C_0 \cdots C_N]$ solely for self-supervised feature loss rather than action anticipation.

Instead, for action anticipation, we propose to train learnable 'anticipation tokens' to learn useful global context from the clip tokens. The output of the anticipation token is no longer restricted to be in the same latent space as the output of clip-feature tokens like in [25] and can better capture the information needed to anticipate the next action. Further, we propose to use multiple anticipation tokens to generate additional supervision, with each token attending to different past video lengths and producing forecasts for different time horizons in the future.

3.3.2 Self-Supervision via Shuffled Feature Prediction

Prior works explored using the self-supervision task of predicting frame or clip features and using an MSE loss [25]. We propose using a generalized form of masking-based self-supervision [33] based on predicting shuffled future features. Predicting future clip features has a two-fold benefit: (A) It encourages causally learning the observed sequence incentivizing the model to grasp underlying scene dynamics and (B) Prediction in the latent feature space allows reasoning semantically about the future without wasting modeling capacity with low pixel level scene details.

Different from loss functions proposed in prior works [25], we propose an improved auto-regressive scheme for self-supervised learning by future feature prediction. Rather than sequentially predicting missing clip features in order to use a causal attention mask, we propose to use a model-level augmentation of the attention masking scheme where some of the attention weights are not used from the original causal mask.

Auto-regressive $[1,2,3,4]$	Shuffle $\pi_1 : [4,1,3,2]$	Another Shuffle $\pi_2 : [3,4,2,1]$
$1 \mapsto 2$	$4 \mapsto 1$	$3 \mapsto 4$
$2 \mapsto 3$	$1 \mapsto 3$	$4 \mapsto 2$
$3 \mapsto 4$	$3 \mapsto 2$	$2 \mapsto 1$

Table 1. **Encoding the input permutation π.** Shuffling the input sequence arbitrarily changes the successor of each token.

Random Masking. Construction of a sparse augmented attention mask is non-trivial. Simply generating a random mask such as in MAE [33] **fails** to isolate information within the intended partitioning due to multi-hop message passing caused by repeated application of the same mask. For example, in Figure 4, the attention mask prohibits token 1 to access token 2 and 3 (shown by 'white'). However, after two layers, token 1 in layer 2 can access information in token 2 in layer 0 (shown by red border), indirectly through token 4 in later 1 because of multi-hop message passing. This temporal information leakage can cause self-supervision to fail. For example, self-supervising token 1 with token 2 feature prediction would collapse in setup of Figure 4. To prevent this, we propose a simple yet effective solution.

Shuffled Causal Masking. We know that vanilla causal masks ensure that information available to each token is invariant under repeated applications of the attention mask, *i.e.*, multi-hop message passing. Since multi-hop message passing is token permutation invariant, any permutation of the tokens from the causal mask will preserve the invariance of accessible information under multiple hops. Thus, this allows a general framework for structured randomized mask construction without temporal leakage. We notice that a row-wise permutation of the attention mask is equivalent to the same permutation applied to the sequence itself. Hence, the same effect as properly constructed random masks can simply be achieved by shuffling the input token sequence itself. This allows generalizing vanilla autoregressive prediction order to arbitrary sequence permutations without any multi-hop information leakage through the layers. Thus, rather than predicting clip features sequentially from 0 to $N-1$ like [25], our proposed scheme allows for *exponentially* more variations.

Referring to Figure 3, we first add absolute position embeddings (APE) to each of the clip features before shuffling them. This allows the transformer to leverage the information about the *actual* temporal order of each clip in the video. Without the absolute position embeddings, the transformer is input permutation equivariant, which is not a desirable property for action forecasting. However, while APE is sufficient for positional information in vanilla autoregressive ordering, it is not enough for prediction under our proposed Shuffled Causal Masking (SCM) scheme. In autoregressive ordering, for any specific token, the next token in the sequence corresponds to a fixed position and hence the self-supervised training process can subtly learn this bias via next-token feature supervision. In contrast, under SCM the temporal position of the next token varies depending on the shuffling permutation. Hence, self-supervised training cannot learn to perform effective feature prediction without information about the shuffling permutation.

3.3.3 Permutation Position Encoding

Naïve Encodings. For an L length sequence, there exist $L!$ possible permutations. Encoding each permutation (π) by itself is clearly intractable. First, we observe that encoding π^* as a set of $O(L)$ embeddings which are shared among different π is more efficient. A follow-up solution would be to instead encode each predecessor $\mapsto$ successor relationship as an encoding and have $L-1$ of such embeddings together encode π^*. This reduces to the total number of required embeddings to $L(L-1)$ from $L!$. Each π^* now shares every one of its L embeddings with other π, but considered as a set, the L embeddings uniquely encode π^*. However, even L^2 becomes intractable for large L.

Permutation Position Encodings (πPE) provides an elegant solution for encoding π^* requiring only L total encodings to be learned. πPE encodes the predecessor $\mapsto$ successor relationships but further simplifies by noting that adding the encoding to the token itself makes the predecessor information redundant. The token itself is the predecessor and has the positional information available from APE. Hence, we simply encode the successor positional information, and that in combination with APE uniquely encodes π^*. Hence, we design πPE to be the encoding of the *original* temporal position of the successor in the permuted sequence. So for the π_1 permutation in Table 1, we would add πPE[1] to token 4, πPE[3] to token 1 and πPE[2] to token 3.

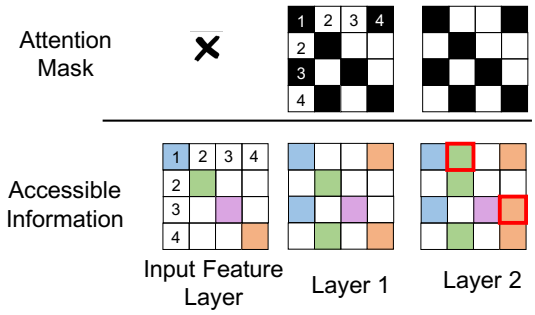

Figure 4. **Random Masking Self-Supervision** Illustration of how naive random masking (white denotes masked out) **fails** for self-supervised feature prediction task. We can see that in Layer 1 the tokens can only access information according to the provided mask. However, if the same mask is used in Layer 2, there is information leakage across tokens which is undesirable.

3.4. Overall Mechanism

The past video is split into clips, and the clip embeddings $[C_0 \cdots C_{N-1}]$ are extracted using a video backbone (Figure 3). Anticipation tokens are concatenated to the extracted sequence (§3.3.1), followed by the addition of absolute position encoding to the concatenated sequence. Now, the clip embedding is shuffled according to a randomly chosen permutation π^* to obtain $[C_{\pi^*[0]} \cdots C_{\pi^*[N-1]}]$. The sampled permutation π^* is then encoded by adding the successor's (§3.3.3) embedding to each token. In Figure 3, $\pi^*[1] = 3$ and $\pi^*[2] = N$, hence πPE$[\pi^*[1+1]]$ is added to the token $\pi^*[1]$ *i.e.*, πPE$[N]$ is added to E_3, which is used to self-supervise the feature at position 1 in the shuffled sequence, *i.e.*, token 3 (§3.3.2). The shuffled input sequence, in addition to the anticipation tokens, is now propagated through the transformer encoder using a causal attention masking scheme and the output is decoded with the head networks.

Head Networks We propose using three MLP heads on top of the transformer encoder network (Figure 3). Two MLP heads decode the anticipation tokens into the predicted future action distribution. The third MLP head upsamples the encoded tokens to the original representation space of the input tokens to allow self-supervision loss.

Split	Method	Addl. Modality	Init	Epic Boxes	Top-5 Recall			Parameters	GPU	Inference
					Verb	Noun	Action	($\times 10^6$)	Hours	Latency (ms)
Val	TempAgg [70]	None	IN1K		24.2	29.8	13.0	-	-	-
	RULSTM [19]	None	IN1K		-	-	13.3	-	-	-
	RULSTM [19]	Obj+Flow	IN1K	✓	30.8	27.8	14.0	-	-	-
	TempAgg [70]	Obj+Flow+ROI	IN1K	✓	23.2	31.4	14.7	-	-	-
	AVT [25]	None	IN21K		30.2	31.7	14.9	378	-	420
	AVT+ [25]	Obj	IN21K	✓	28.2	32.0	15.9	-	-	-
	TSN-AVT+ [25]	Obj	IN21K	✓	31.8	25.5	14.8	-	-	-
	MeMVit [80]	None	K400		32.8	33.2	15.1	59	-	160
	MeMVit [80]	None	K700		32.2	37.0	17.7	212	368	350
	RAFTformer	None	K400 + IN1K		33.3	35.5	17.6	**26**	**23**	**40**
	RAFTformer	None	K700		33.7	37.1	18.0	**26**	27	110
	RAFTformer-2B	None	K700 + IN1K		**33.8**	**37.9**	**19.1**	52	50	160
Test	RULSTM [19]	Obj+Flow	IN1K	✓	25.3	26.7	11.2	-	-	-
	TBN [83]	Obj+Flow	IN1K	✓	21.5	26.8	11.0	-	-	-
	AVT+ [25]	Obj+Flow	IN21K	✓	25.6	28.8	12.6	-	-	-
	Abstract Goal [69]	Obj+Flow	IN1K	✓	31.4	30.1	14.3	-	-	-
	AFFT [85]	Obj+Flow	-	✓	20.7	31.8	14.9	-	-	-
	RAFTformer	None	K400 + IN1K		27.3	32.8	14.0	26	23	40
	RAFTformer	None	K700		27.4	34.0	14.7	26	27	110
	RAFTformer-2B	None	K700 + IN1K		**30.1**	**34.1**	**15.4**	52	50	160

Table 2. **Offline Evaluation Results** on the EK-100 dataset for $t_f = 1$ second horizon. Latency is measured over the entire model including the backbone & head networks on a single 16G Tesla V100 GPU. Methods that use other modalities (+RGB) are deemphasized.

3.5. Loss Functions

We observe that the ground truth action distributions are often long-tailed, and propose to use the the focal loss [52], for supervising the future action prediction distributions.

$$\mathcal{L}_{\text{focal}} = \sum_{A_i \in A} \sum_{i=0}^{n} -(1 - p_{A_i})^{\gamma} \log(p_{A_i})$$

where p_i is the predicted probability for the correct class for the ith example, A_i represents predictions from a specific anticipation token and γ is the focusing parameter where $\gamma = 0$ is cross entropy loss. Increasing γ results in an increased penalty for hard, misclassified examples.

For self-supervision using shuffled causal masking to predict next-token embeddings, we use an expected ℓ_2 loss over both past tokens, with the expectation being oversampled permutations $\pi^* \sim \pi$.

$$\mathcal{L}_{\text{SCM}} = \mathop{\mathbb{E}}_{\pi^* \sim \pi} \sum_{j=0}^{N-1} \|E_{\pi^*[j]} - \hat{E}_{\pi^*[j]}\|_2^2$$

where $E_{\pi^*[j]}$ and $\hat{E}_{\pi^*[j]}$ denotes the original and the predicted clip embedding at position j after permuting with π^*. For future token prediction, we use a simple ℓ_2 loss:

$$\mathcal{L}_{\text{future}} = \|E_{\text{future}} - \hat{E}_{\text{future}}\|_2^2$$

where E_{future} is the MViT embedding for the next clip (after t_f). Finally, the overall loss is simply a weighted sum,

$$\mathcal{L} = \mathcal{L}_{\text{focal}} + \lambda_1 \mathcal{L}_{\text{SCM}} + \lambda_2 \mathcal{L}_{\text{future}}$$

4. Experiments

Datasets & Metrics. We use the widely benchmarked EPIC-KITCHENS dataset [9], an unscripted set with nearly 20× more action classes and 10-100× more observed sequences than other action datasets like 50 Salads [47] and Breakfast [46], used in prior work [19]. We use both the EPIC-55 & EPIC-100 anticipation splits and the EGTEA+ Gaze dataset [50]. Dataset details in supplementary.

Since human decision-making is inherently multimodal, we propose an approach to predict multiple reasonable future action forecasts. In addition to top-1 accuracy, we report top-5 recall following prior works [19, 25, 80]. We use the baseline data splits and report metrics on both the validation and test set. In addition, we report the number of trainable model parameters (M), the total compute spent for training the model to convergence on a Tesla V100 GPU (in hours), and the inference latency (in milliseconds).

4.1. Evaluation Setting

Offline evaluation is the setting where the model inference latency is ignored, or in other words, assumed to be zero. All prior works [19, 25, 80] consider this setting. Referring to Figure 1, prior works assume access to all past video frames up till the present moment $T = t$. Using this information, the model then predicts that action with a t_f second horizon at $T = t + t_f$. However, the prediction would actually be produced at time $T = t + t_l$ where t_l is the model inference latency. This is not practically useful

Model	Init	Latency (t_l ms)	Inference Start Time Stamp	Inference End Time Stamp	Target Time Stamp	Top-5 Recall		
						Verb	Noun	Action
AVT [25]	IN21K	$t_{\text{avt}} = 420$	T	$T + t_{\text{avt}}$	$T + 1$	30.2	31.7	14.9
RAFTformer	K400 + IN1k	$t_{\text{ours}} = 40$	$T + t_{\text{avt}} - t_{\text{ours}}$	$T + t_{\text{avt}}$	$T + 1$	34.1	38.2	**19.3** (+4.4)
MemViT [80]	K400	$t_{\text{vit}} = 160$	T	$T + t_{\text{vit}}$	$T + 1$	32.8	33.2	15.1
RAFTformer	K400 + IN1k	$t_{\text{ours}} = 40$	$T + t_{\text{vit}} - t_{\text{ours}}$	$T + t_{\text{vit}}$	$T + 1$	33.8	37.1	**18.1** (+3.0)
MemViT [80]	K700	$t_{\text{vit}} = 350$	T	$T + t_{\text{vit}}$	$T + 1$	32.2	37.0	17.7
RAFTformer	K400 + IN1k	$t_{\text{ours}} = 40$	$T + t_{\text{vit}} - t_{\text{ours}}$	$T + t_{\text{vit}}$	$T + 1$	33.7	37.9	**19.0** (+1.3)

Table 3. **Real-time Evaluation Results** for benchmarking action forecasting methods in a practical setting (§4.1). Each comparison is performed between a pair of models where their start time ('Inference Start') times have been adjusted (Figure 1) by their latency so that they produce the forecasting output for the 'Target Time' *simultaneously* ('Inference End'). For prior works [25, 80], start & end times are kept the same as their original offline settings (Table 2) to avoid any training recipe change. Faster models can pragmatically utilize recent frames while slower models must rely on higher fidelity prediction from older frames. Latency measured on a single 16G Tesla V100 GPU.

Model	Init	Top-1 Acc
RULSTM [19]	TSN/IN1k	13.1
ActionBanks [70]	TSN/IN1k	12.3
AVT-h [25]	TSN/IN1k	13.1
RAFTformer	TSN/IN1k	**13.8**

Table 4. **Offline Evaluation** results on the Epic-Kitchens-55.

Model	Setting				Top-5 Recall
	AT	FL	SCM	2×AT	
Mean pooling					15.2
RAFTformer	✓				16.1 (+0.9)
	✓	✓			16.8 (+0.7)
	✓	✓	✓		17.4 (+0.6)
	✓	✓	✓	✓	**17.6** (+0.2)

Table 5. **EK100 Ablation Study.** FL is feature loss (no SCM), SCM is Shuffled Causal Masking (§3.3.2), AT (2×) are single (double) anticipation tokens (§3.3.1).

since often, we require time horizon t_f to meaningfully use the predicted future outcomes. Further, this does not account for absurdities where a large model might even have latency $t_l > t_f$, in which case the model is predicting an action that has already happened by the time it predicts it! A complex model can have great offline performance but its inference time would make it unusable in practice.

Real-time evaluation. To remedy this impractical situation, we consider the real-time evaluation setting. In this setting, the model is required to finish inference with at least t_f seconds horizon before the target time (Fig. 1). Hence, the model is allowed access to past video data only for $T < t - t_l$. This setting even allows for large models where $t_l > t_f$ since predictions would still be produced with t_f seconds before the target time. Unlike other recognition scenarios where latency is a mere annoyance, in the case of future forecasting, models are often used in real-time rather than offline. The offline setting subtly oversteps the prediction horizon t_f, by receiving forecasts with a margin much less than the postulated t_f horizon. By coupling model latency to the past video data observed, the real-time setting incentivizes the development of efficient forecasting methods that utilize the recent data better than relying on slow large models that cannot miss the recent frames, which arguably are the most crucial for near future prediction.

4.2. EPIC-Kitchens-100

Network Details. For feature extraction, we fix the pre-trained 16x4 (K400+IN1K) and 32x3 (K700) MViT-B backbones [12, 49] trained for recognition on the EK100 dataset. Each clip is embedded as a 768-dimensional feature vector. Unless mentioned otherwise, the action forecasting experiments use a $t_f = 1$ second horizon. RAFTformer encoder is a lightweight 4 layer, 4 head transformer encoder using post-normalization and ReLU activations. A linear projection from up-projects $768 \rightarrow 1024$ for the transformer. We only shuffle during training with a probability of 0.3. The output of the encoder is down projected from $1024 \rightarrow 768$ channels. Both short-term and long-term action prediction heads are fast & lightweight MLPs ($1024 \rightarrow 2048 \rightarrow 3806$). We set $\lambda_1 = \lambda_2 = 14$. For other details, please see supplementary.

Offline Evaluation. In Table 2 we first report RAFTformer results against prior works, including previous SOTA MeMViT [80]. Results that use additional modalities like Object (Obj), Flow (Flow), Object Region of Interest (ROI), or Epic Kitchen boxes are de-emphasized. Using only RGB, RAFTformer (K700) outperforms the AVT RGB model in action T5R by 3.1 points and the AVT RGB+Obj model by 2.1 points. RAFTformer slightly outperforms MeMViT action T5R while predicting actions **8.75**× faster with **8.2**× lesser trainable parameters and a **16**× faster training. We also combine two separate RAFTformer models to train a two-backbone model (RAFTformer-2B) that achieves state-of-the-art by with a 1.4% T5R increase. In the second section, we report our submitted results to the EK100 test server. We observe that RAFTformer general-

Model	Init	Modality	Top-5 Recall
DMR [75]	-	RGB	38.1
ATSN [9]	TSN/IN1k	RGB+Flow	31.6
MCE [18]	TSN/IN1k	RGB+Flow	43.8
TCN [6]	-	RGB	47.1
FN [10]	VGG-16	RGB	42.7
ED [21]	VGG-16/TS	RGB+Flow	54.6
RULSTM [19]	TSN/IN1k	RGB+Obj+Flow	58.6
RAFTformer	TSN/IN1k	RGB	**63.5**

Table 6. **EGTEA Gaze+.** Under same initialization, RAFTformer outperforms prior methods without additional modalities.

izes well to the test set, improving upon the AVT+ multi-modal ensemble model [25] by 2.8 T5R points.

4.3. Additional Datasets

EPIC-Kitchens 55. We also compare RAFTformer to prior baselines on the EK55 dataset in Table 4 on top-1 accuracy. We evaluate against other models using the same initialization for a fair comparison. We observe that RAFTformer outperforms prior benchmarks by about 0.7% using the exact same backbone features. This shows the superiority of our proposed shuffle causal masking (§3.3.2) & anticipation token prediction (§3.3.1) in a controlled setting.

Real-time Evaluation. In table 3, we fix the target - start time difference to be what the prior models such as, AVT [25] or MeMViT [80] were designed for, to avoid any unintended change in their training recipes. Hence, we shift RAFTformer start time so that the output forecasts are produced simultaneously for the pair.

We observe that RAFTformer outperforms prior methods by an even *larger margin* in the real-time evaluation setting than than the offline setting. RAFTformer effectively utilizes its compute to produce the maximum forecasting effect at the least latency cost. This allows RAFTformer to exploit more recent information that slower models miss because of higher inference latencies ($t_{\text{ours}} < \{t_{\text{avt}}, t_{\text{vit}}\}$). This also shows the disproportional effect of inference latency on forecasting performance. While RAFTFormer K400 + IN1K has a similar Top-5 Recall as MeMViT K700 in the offline setting, in the practical real-time evaluation (Table 3), RAFTformer outperforms MeMViT by 1.3% by leveraging its faster inference latency (40 *vs*. 350 ms).

4.3.1 Latency *vs*. Offline & Real-time Forecasting

In offline evaluation, latency is ignored and all models have data access parity. As scaling laws [39] would predict, bigger models (with higher latency) have stronger forecasting performance (Blue curve in Fig. 2). In the real-time setting, a higher latency implies access to less recent data (Green curve in Fig. 2) and hence, an interesting trade-off materializes. At first, as latency (and model size) increases, the performance improves. This is because the benefit of having a more expressive model outweighs the cost of access to older video data. However, as the latency increases further, the real-time performance starts decreasing. Now the harm of not having access to recent data outweighs the benefit of a more expressive model. For our setting, the optimal RAFTformer model has a latency of 40 ms with an offline and real-time performance of 19.6 and 19.3 Top-5 recall respectively. A similar trade-off would exist for any combination of dataset, model, and deployment hardware regimes. Plots are for RAFTformer models of varying backbone parameter count for $t_f = 0.58$ second prediction on EK100.

4.3.2 Ablations

We ablate RAFTformer thoroughly in Table 5. In contrast to prior works [25,80] that pool information or use the same embedding for feature supervision and action prediction, we train explicit anticipation tokens (§3.3.1) to perform forecasting, improving T5R by 1.1. Anticipation tokens specialize in predicting the next action instead of using features that multi-task between reconstructing clip features and future action prediction [25]. Self-supervision via feature loss (FL) further improves by 0.7 T5R. Further, our SCM technique effectively regularizes FL that improves 0.6 T5R. Finally, an additional anticipation token (2×AT) for supervision further improves 0.2 T5R. Additional longer-horizon anticipation task helps in learning transferable global video features to the main anticipation task.

EGTEA Gaze+. We also evaluate the EGTEA Gaze+ dataset in Table 6 to showcase RAFTformer performance on new settings outside of EPIC-Kitchens. Again, we use the setting of anticipating 1s into the future and use the T5R metric. While prior methods use many input modalities including as object, flow, and RGB, we significantly outperform the prior state-of-the-art in this setting [19] by 4.9%. Note that we use pre-extracted TSN RGB features as provided by [19] for direct comparison. Our very strong performance on another egocentric video dataset with complex human tasks further validates the promise of RAFTformer.

5. Conclusion

We propose Real-Time Action Forecasting Transformer (RAFTformer), a parsimonious two-stage fully transformer-based architecture consisting of a short-range video transformer backbone for feature extraction and a long-range head transformer encoder for long-term temporal aggregation across multiple clips. We also introduce a real-time evaluation setting for action forecasting models that directly penalizes high latency and closely mimics the real-world deployment scenario for forecasting models. RAFTformer outperforms prior state-of-the-art methods by significant margins across several action forecasting benchmarks in the offline setting, and by an even larger margin of upto 4.4 T5R, in the real-time setting. RAFTformer achieves 9× lower inference latency at the same forecasting fidelity, using 16× less training compute and 10× lesser trainable parameters than prior baselines.

References

[1] Yazan Abu Farha, Alexander Richard, and Juergen Gall. When will you do what?-anticipating temporal occurrences of activities. In *CVPR*, pages 5343–5352, 2018. 2

[2] Georges Aoude, Joshua Joseph, Nicholas Roy, and Jonathan How. Mobile agent trajectory prediction using bayesian nonparametric reachability trees. In *Infotech@ Aerospace 2011*, page 1512. 2011. 2

[3] Relja Arandjelovic and Andrew Zisserman. Look, listen and learn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 609–617, 2017. 2

[4] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6836–6846, 2021. 2

[5] Yuki M Asano, Christian Rupprecht, and Andrea Vedaldi. A critical analysis of self-supervision, or what we can learn from a single image. *arXiv preprint arXiv:1904.13132*, 2019. 2

[6] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018. 8

[7] Sayantan Chatterjee, Faheem H Zunjani, and Gora C Nandi. Real-time object detection and recognition on low-compute humanoid robots using deep learning. In *2020 6th International Conference on Control, Automation and Robotics (ICCAR)*, pages 202–208. IEEE, 2020. 3

[8] Hsu-kuang Chiu, Ehsan Adeli, and Juan Carlos Niebles. Segmenting the future. *IEEE Robotics and Automation Letters*, 5(3):4202–4209, 2020. 2

[9] 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. 6, 8

[10] Roeland De Geest and Tinne Tuytelaars. Modeling temporal structure with lstm for online action detection. In *WACV*, pages 1549–1557. IEEE, 2018. 8

[11] Alexey Dosovitskiy et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 3

[12] Haoqi Fan et al. Multiscale vision transformers. *arXiv preprint arXiv:2104.11227*, 2021. 2, 3, 7

[13] Mingyuan Fan, Shenqi Lai, Junshi Huang, Xiaoming Wei, Zhenhua Chai, Junfeng Luo, and Xiaolin Wei. Rethinking bisenet for real-time semantic segmentation. In *CVPR*, pages 9716–9725, 2021. 3

[14] Christoph Feichtenhofer, Haoqi Fan, Yanghao Li, and Kaiming He. Masked autoencoders as spatiotemporal learners. *arXiv preprint arXiv:2205.09113*, 2022. 2

[15] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, pages 6202–6211, 2019. 2

[16] Basura Fernando, Hakan Bilen, Efstratios Gavves, and Stephen Gould. Self-supervised video representation learning with odd-one-out networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3636–3645, 2017. 2

[17] R James Firby. An investigation into reactive planning in complex domains. In *AAAI*, volume 87, pages 202–206, 1987. 1

[18] Antonino Furnari, Sebastiano Battiato, and Giovanni Maria Farinella. Leveraging uncertainty to rethink loss functions and evaluation measures for egocentric action anticipation. In *ECCV*, pages 0–0, 2018. 8

[19] Antonino Furnari and Giovanni Maria Farinella. Rolling-unrolling lstms for action anticipation from first-person video. *TPAMI*, 2020. 1, 2, 3, 6, 7, 8

[20] Antonino Furnari and Giovanni Maria Farinella. Towards streaming egocentric action anticipation. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 1250–1257. IEEE, 2022. 2

[21] Jiyang Gao, Zhenheng Yang, and Ram Nevatia. Red: Reinforced encoder-decoder networks for action anticipation. *arXiv preprint arXiv:1707.04818*, 2017. 8

[22] Harshayu Girase, Haiming Gang, Srikanth Malla, Jiachen Li, Akira Kanehara, Karttikeya Mangalam, and Chiho Choi. Loki: Long term and key intentions for trajectory prediction. In *ICCV*, pages 9803–9812, 2021. 1, 2

[23] Harshayu Girase, Jerrick Hoang, Sai Yalamanchi, and Micol Marchetti-Bowick. Physically feasible vehicle trajectory prediction. *arXiv preprint arXiv:2104.14679*, 2021. 1

[24] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *CVPR*, pages 244–253, 2019. 2

[25] Rohit Girdhar and Kristen Grauman. Anticipative video transformer. In *ICCV*, pages 13505–13515, 2021. 1, 2, 3, 4, 5, 6, 7, 8

[26] 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

[27] Matthias Grundmann, Vivek Kwatra, Mei Han, and Irfan Essa. Efficient hierarchical graph-based video segmentation. In *2010 ieee computer society conference on computer vision and pattern recognition*, pages 2141–2148. IEEE, 2010. 2

[28] Tengda Han, Weidi Xie, and Andrew Zisserman. Video representation learning by dense predictive coding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019. 2

[29] Tengda Han, Weidi Xie, and Andrew Zisserman. Memory-augmented dense predictive coding for video representation learning. In *European conference on computer vision*, pages 312–329. Springer, 2020. 2

[30] Mark Handley. Delay is not an option: Low latency routing in space. In *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*, pages 85–91, 2018. 1

[31] Harish Haresamudram, Apoorva Beedu, Varun Agrawal, Patrick L Grady, Irfan Essa, Judy Hoffman, and Thomas Plötz. Masked reconstruction based self-supervision for human activity recognition. In *Proceedings of the 2020 inter-*

national symposium on wearable computers, pages 45–49, 2020. 2

[32] Joel Hasbrouck and Gideon Saar. Low-latency trading. *Journal of Financial Markets*, 16(4):646–679, 2013. 1

[33] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, pages 16000–16009, 2022. 2, 4

[34] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 3

[35] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020. 2

[36] Dinesh Jayaraman and Kristen Grauman. Learning image representations tied to ego-motion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1413–1421, 2015. 2

[37] Dinesh Jayaraman and Kristen Grauman. Slow and steady feature analysis: higher order temporal coherence in video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3852–3861, 2016. 2

[38] Eugen Käfer, Christoph Hermes, Christian Wöhler, Helge Ritter, and Franz Kummert. Recognition of situation classes at road intersections. In *2010 IEEE International Conference on Robotics and Automation*, pages 3960–3965. IEEE, 2010. 2

[39] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. 8

[40] Shian-Ru Ke, Hoang Le Uyen Thuc, Yong-Jin Lee, Jenq-Neng Hwang, Jang-Hee Yoo, and Kyoung-Ho Choi. A review on video-based human activity recognition. *Computers*, 2(2):88–131, 2013. 2

[41] Chanho Kim, Li Fuxin, Mazen Alotaibi, and James M Rehg. Discriminative appearance modeling with multi-track pooling for real-time multi-object tracking. In *CVPR*, pages 9553–9562, 2021. 3

[42] Dahun Kim, Donghyeon Cho, and In So Kweon. Self-supervised video representation learning with space-time cubic puzzles. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 8545–8552, 2019. 2

[43] Irena Koprinska and Sergio Carrato. Temporal video segmentation: A survey. *Signal processing: Image communication*, 16(5):477–500, 2001. 2

[44] Bruno Korbar, Du Tran, and Lorenzo Torresani. Cooperative learning of audio and video models from self-supervised synchronization. *Advances in Neural Information Processing Systems*, 31, 2018. 2

[45] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. 2017. 1

[46] 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. 6

[47] Colin Lea, René Vidal, and Gregory D Hager. Learning convolutional action primitives for fine-grained action recognition. In *ICRA*, pages 1642–1649. IEEE, 2016. 6

[48] Jiangtong Li, Wentao Wang, Junjie Chen, Li Niu, Jianlou Si, Chen Qian, and Liqing Zhang. Video semantic segmentation via sparse temporal transformer. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 59–68, 2021. 2

[49] Yanghao Li et al. Improved multiscale vision transformers for classification and detection. *arXiv preprint arXiv:2112.01526*, 2021. 2, 7

[50] 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. 6

[51] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017. 3

[52] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017. 6

[53] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37. Springer, 2016. 3

[54] Zhihua Liu, Pierre Magal, Ousmane Seydi, and Glenn Webb. A covid-19 epidemic model with latency period. *Infectious Disease Modelling*, 5:323–337, 2020. 1

[55] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3202–3211, 2022. 2

[56] Tahmida Mahmud, Mahmudul Hasan, and Amit K Roy-Chowdhury. Joint prediction of activity labels and starting times in untrimmed videos. In *ICCV*, pages 5773–5782, 2017. 2

[57] Karttikeya Mangalam, Yang An, Harshayu Girase, and Jitendra Malik. From goals, waypoints & paths to long term human trajectory forecasting. *arXiv preprint arXiv:2012.01526*, 2020. 1

[58] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It is not the journey but the destination: Endpoint conditioned trajectory prediction. In *ECCV*, pages 759–776. Springer, 2020. 1

[59] RJW Mansfield. Latency functions in human vision. *Vision research*, 13(12):2219–2234, 1973. 1

[60] Peter McLeod. Visual reaction time and high-speed ball games. *Perception*, 16(1):49–59, 1987. 2

[61] Daniel Neimark, Omri Bar, Maya Zohar, and Dotan Asselmann. Video transformer network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3163–3172, 2021. 2

[62] Kriti Ohri and Mukesh Kumar. Review on self-supervised image recognition using deep neural networks. *Knowledge-Based Systems*, 224:107090, 2021. 2

[63] Venkata N Padmanabhan and Jeffrey C Mogul. Improving http latency. *Computer Networks and ISDN Systems*, 28(1-2):25–35, 1995. 1

[64] Bo Pang, Kaiwen Zha, Hanwen Cao, Chen Shi, and Cewu Lu. Deep rnn framework for visual sequential applications. In *CVPR*, pages 423–432, 2019. 2

[65] David A Patterson. Latency lags bandwith. *Communications of the ACM*, 47(10):71–75, 2004. 1

[66] Amir Rasouli, Iuliia Kotseruba, and John K Tsotsos. Pedestrian action anticipation using contextual feature fusion in stacked rnns. *arXiv preprint arXiv:2005.06582*, 2020. 2

[67] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, 2016. 3

[68] Cristian Rodriguez, Basura Fernando, and Hongdong Li. Action anticipation by predicting future dynamic images. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018. 2

[69] Debaditya Roy and Basura Fernando. Predicting the next action by modeling the abstract goal. *arXiv preprint arXiv:2209.05044*, 2022. 6

[70] Fadime Sener, Dipika Singhania, and Angela Yao. Temporal aggregate representations for long-range video understanding. In *ECCV*, pages 154–171. Springer, 2020. 2, 3, 6, 7

[71] Yuge Shi, Basura Fernando, and Richard Hartley. Action anticipation with rbf kernelized feature mapping rnn. In *ECCV*, pages 301–317, 2018. 2

[72] Gurkirt Singh, Suman Saha, Michael Sapienza, Philip HS Torr, and Fabio Cuzzolin. Online real-time multiple spatiotemporal action localisation and prediction. In *ICCV*, pages 3637–3646, 2017. 3

[73] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7464–7473, 2019. 2

[74] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *arXiv preprint arXiv:2203.12602*, 2022. 2

[75] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Anticipating visual representations from unlabeled video. In *CVPR*, pages 98–106, 2016. 8

[76] Haochen Wang, Xiaolong Jiang, Haibing Ren, Yao Hu, and Song Bai. Swiftnet: Real-time video object segmentation. In *CVPR*, pages 1296–1305, 2021. 2, 3

[77] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE transactions on pattern analysis and machine intelligence*, 41(11):2740–2755, 2018. 2

[78] Wen Wang, Xiaojiang Peng, Yanzhou Su, Yu Qiao, and Jian Cheng. Ttpp: Temporal transformer with progressive prediction for efficient action anticipation. *Neurocomputing*, 438:270–279, 2021. 3

[79] Donglai Wei, Joseph J Lim, Andrew Zisserman, and William T Freeman. Learning and using the arrow of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8052–8060, 2018. 2

[80] Chao-Yuan Wu et al. Memvit: Memory-augmented multiscale vision transformer for efficient long-term video recognition. *arXiv:2201.08383*. 1, 2, 3, 6, 7, 8

[81] Yu-Syuan Xu, Tsu-Jui Fu, Hsuan-Kung Yang, and Chun-Yi Lee. Dynamic video segmentation network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6556–6565, 2018. 2

[82] Ceyuan Yang, Yinghao Xu, Bo Dai, and Bolei Zhou. Video representation learning with visual tempo consistency. *arXiv preprint arXiv:2006.15489*, 2020. 2

[83] Olga Zatsarynna, Yazan Abu Farha, and Juergen Gall. Multimodal temporal convolutional network for anticipating actions in egocentric videos. In *CVPR*, pages 2249–2258, 2021. 6

[84] Bowen Zhang, Limin Wang, Zhe Wang, Yu Qiao, and Hanli Wang. Real-time action recognition with enhanced motion vector cnns. In *CVPR*, pages 2718–2726, 2016. 3

[85] Zeyun Zhong, David Schneider, Michael Voit, Rainer Stiefelhagen, and Jürgen Beyerer. Anticipative feature fusion transformer for multi-modal action anticipation. *arXiv preprint arXiv:2210.12649*, 2022. 3, 6

[86] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos. In *ECCV*, pages 803–818, 2018. 2

[87] Xizhou Zhu, Yuwen Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2349–2358, 2017. 2