Multi-Modal Temporal Convolutional Network for Anticipating Actions in Egocentric Videos

Olga Zatsarynna, Yazan Abu Farha and Juergen Gall
University of Bonn
Germany
{zatsarynna, abufarha, gall}@iai.uni-bonn.de

Abstract

Anticipating human actions is an important task that needs to be addressed for the development of reliable intelligent agents, such as self-driving cars or robot assistants. While the ability to make future predictions with high accuracy is crucial for designing the anticipation approaches, the speed at which the inference is performed is not less important. Methods that are accurate but not sufficiently fast would introduce a high latency into the decision process. Thus, this will increase the reaction time of the underlying system. This poses a problem for domains such as autonomous driving, where the reaction time is crucial. In this work, we propose a simple and effective multi-modal architecture based on temporal convolutions. Our approach stacks a hierarchy of temporal convolutional layers and does not rely on recurrent layers to ensure a fast prediction. We further introduce a multi-modal fusion mechanism that captures the pairwise interactions between RGB, flow, and object modalities. Results on two large-scale datasets of egocentric videos, EPIC-Kitchens-55 and EPIC-Kitchens-100, show that our approach achieves comparable performance to the state-of-the-art approaches while being significantly faster.

1. Introduction

Anticipating future events is of great importance for intelligent agents. There are many real-world scenarios, where apart from recognizing what is happening in the current moment, one also needs to make predictions about the future. For example, autonomous driving systems need to anticipate pedestrians movement to avoid collisions. Another field of application is assistive robotics, where the ability of robots to anticipate future human activities allows for smoother and more productive interactions. In our work, we focus on human activity anticipation since it is a challenging yet crucial task for an intelligent system to be deemed as such.

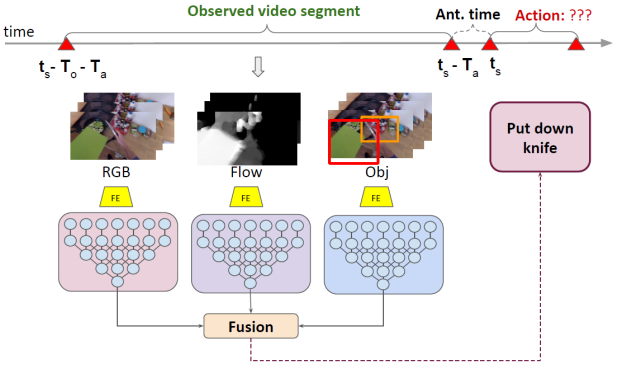

Figure 1. The short-term action anticipation task predicts the next unobserved action T_a seconds before it occurs. To address this task, we propose a multi-modal approach based on temporal convolutional networks that achieves state-of-the-art results while being faster than traditional RNN-based approaches.

In recent years, the number of works addressing the task of action anticipation has experienced a substantial increase. Generally, one could subdivide these works into two categories based on the time horizon of anticipation they tackle. While some works try to anticipate several actions into the future (long-term anticipation) [1, 11, 15, 25], others aim at anticipating only the next action at a fixed anticipation time based on the recent observations preceding it (short-term anticipation) [14, 20, 23, 45]. In our work, we deal with the second setting as illustrated in Figure 1.

Initially, this task has been addressed by predicting representations of the future frames and anticipating the actions by training a classifier on them [45]. Such approaches, while being successful on videos shot from the third-person view, do not perform well on egocentric videos captured from the first-person view. Egocentric action anticipation has been addressed in the work of Furnari *et al.* [14], who introduced a multi-modal LSTM-based [19] encoder-decoder network. Recently, several new methods have been

proposed [9, 28, 38, 48] that show improved performance in the egocentric action anticipation.

While these methods demonstrate better performance in the egocentric short-term action anticipation, little attention has been paid to the effectiveness of the training and inference procedures of the methods. Many of the above mentioned works, in particular [9, 14, 48], use recurrent layers for performing temporal sequence modelling. However, in recent years it has been repeatedly indicated [3, 16, 27, 44] that for many sequence processing tasks convolution-based methods are much faster than canonical recurrent layers such as LSTMs, and still show a similar or even superior performance. Convolution-based architectures are also easier to train, since they do not possess notorious drawbacks of recurrent layers such as vanishing gradients and inability to model long-term dependencies. On a different note, some of the action anticipation works [9, 28] also gather and incorporate additional data or annotations into training, which is a costly and labor intensive process.

In our work, we propose a model that addresses the previously mentioned limitations. We introduce a multimodal network based on a hierarchy of temporal convolutions. Our network consists of three parallel branches where each branch operates on features extracted from RGB, optical flow or object modalities. To fuse these modalities, we introduce a multi-modal fusion mechanism that captures both mutual and pairwise interactions between the different branches. In contrast to previous approaches [9, 28], our model does not require any additional data or annotations. We evaluate our approach on two large-scale datasets of egocentric videos: EPIC-Kitchens-55 [6, 7] and EPIC-Kitchens-100 [8]. We show that our model achieves comparable results to the state-of-the-art while being at least two times faster during both training and inference.

2. Related Work

2.1. Action Anticipation in Videos

There are several lines of work in the area of action anticipation that differ in the time horizon of predictions. Some approaches focus on long-term predictions. That is, given a subset of observed actions, they aim to predict multiple actions into the future or even all subsequent actions. In the work by Abu Farha *et al*. [1], two approaches for long-term anticipation have been proposed, based on RNN and CNN. The RNN-based method predicts labels and lengths of the upcoming actions by feeding its predicted values back to the network for future predictions. The CNN-based approach, unlike the previous one, predicts all future actions in a single step. It makes predictions by encoding both its input and output in a matrix form. To avoid intermediate computations and accumulation of errors, Ke *et al*. [25] introduced a time-conditioned method that anticipates long-term actions in one shot. Gammulle *et al*. [15] proposed a network that models long-term relationships within the input sequence with the help of a neural memory module, whose refined output is then used to make action predictions. In [11], a sequence-to-sequence model is used to predict future activities and their durations. Additionally, the authors leveraged cycle consistency over time by predicting past actions on the basis of the future predictions made by the network. In contrast to these approaches, we focus on short-term action anticipation from egocentric videos.

For the task of short-term action anticipation, the goal is to forecast an action several seconds prior to its occurrence. State-of-the-art methods usually take the most recent observations into account and predict actions up to several seconds into the future. For example, Vondrick *et al*. [45] proposed a mixture of regression networks to learn a representation of a frame one second in the future based on the frame at the current time step. Then, to predict the action, they categorize the predicted representations with a classifier network. Gao *et al*. [23] further extended the previous idea and introduced an encoder-decoder network that anticipates a sequence of future representations based on an observed sequence of representations, instead of just a single representation. In the work of Jain *et al*. [20], the authors also used an encoder-decoder network, albeit with several modalities and a loss that exponentially increases with time to prevent overfitting and encourage early anticipation. Different from previous works, Damen *et al*. [6] leveraged an action recognition network based on TSN [46] for the task of action anticipation. During training, the network receives the observed segment preceding the action of interest as input, while the corresponding label is set to the category of the action that needs to be predicted. Miech *et al*. [33] proposed to predict future actions by averaging predictions of two complementary modules: predictive and transitional, where the predictive model directly anticipates the upcoming action, and the transitional model is constrained to at first output the current action and then use the acquired information to anticipate the future. In [14], the RU-LSTM network was introduced, that consists of two LSTM networks. The first LSTM summarizes the past observations whereas the second one predicts the future actions. Camporese *et al*. [5] further proposed to extend the RU-LSTM with label smoothing to mitigate over-confident predictions and make their system more uncertainty-aware. Sener *et al*. [38] proposed a framework based on non-local blocks [47], that aggregates multi-scale features from the video by computing interaction between recent and distant observations. The resulting features are then used to anticipate both short-term and long-term actions. Liu *et al*. [28] explicitly incorporate intentional hand movement as an anticipatory representation of actions. They jointly model and predict hand trajectories, interaction hotspots and labels of

future actions. Dessalene *et al*. [10] proposed to use a Graph Convolutional Network (GCN) to model long-term temporal semantic relations between actions based on contact information. They use the constructed graph representations along with appearance features to make anticipation about the future actions. In contrast to these approaches, our approach relies on temporal convolutions to capture dependencies in the input sequence and predict the future action.

2.2. Anticipation of other Modalities

While in our work we address the anticipation of human activities, there are numerous efforts that address the anticipation of other modalities. Anticipation of human trajectories and motion is a popular task that has been addressed in many works [2, 12, 26, 49]. Another line of work deals with predictions of future human poses [13, 18, 21, 31]. Also, many approaches have been proposed for prediction of future semantic segmentation maps of images [4, 22, 30, 34] or even semantic instance segmentation maps [29]. A more difficult problem of future frame prediction has also been explored in [32, 36, 41]. Some works have also addressed the task of generating sentences for describing future frames or upcoming steps in recipes [39].

3. Proposed Approach

We introduce a multi-modal temporal convolutional network for the task of action anticipation. We start by defining the task of action anticipation in Section 3.1. Then, we discuss the video processing procedure in Section 3.2. Finally, we introduce our uni-modal anticipation branch in Section 3.3 and discuss the multi-modal fusion strategy in Section 3.4.

3.1. The Anticipation Task

We adopt the problem definition of action anticipation from [6]. Let T_a be the *anticipation time*, *i.e*. how many seconds in advance an action is predicted, and T_o be the *observation time*, *i.e*. the length of the observed video segment that precedes the action of interest. For a given action video segment $A = [t_s, t_e]$, let t_s and t_e denote times of action start and end respectively. Then, the goal of the action anticipation task is to predict the action label of A by observing a video segment of length T_o preceding the action start time t_s by the anticipation time of T_a seconds, that is $[t_s - (T_a + T_o), t_s - T_a]$. The action anticipation task is illustrated in Figure 1.

In our work, the anticipation time T_a is one second. *I.e*. an action is anticipated one second before its occurrence. The observation time T_o is set to 5.25 seconds. We will show in the experiments the effect of varying the length of the observed video segment on the performance.

3.2. Video Processing

We process the observed video segments in the same way as proposed in [14]. Let the observed video segment be denoted by V. As mentioned previously, we set the length of V to 5.25 seconds (*i.e*. $T_o = 5.25$). During processing, we break V down into snippets that have a duration of $\alpha = 0.25$ seconds, which results in a total of $N = 21$ snippets $\{V_1, V_2, \ldots, V_N\}$. Since the anticipation time is equal to 1.0 second, the resulting video snippets are located at $\{6.0, 5.75, \ldots, 1.0\}$ seconds before the action of interest, where the location is defined by the last frame of the snippet. Each snippet contains several video frames $V_i = \{I_i^1, I_i^2, \ldots, I_i^m\}$, where the exact number of frames m depends on the frame rate of the videos. From each snippet, we sample one or several frames, depending on the modality, which are then used to extract features.

In our model, we use three types of modalities: RGB, optical flow and object features. For extracting the RGB features, each snippet is represented by its last frame I_i^m. For optical flow, the last 5 frames $\{I_i^{m-4}, \ldots, I_i^m\}$ of horizontal and vertical flow are used. To get object features, as for the RGB features, the last frame of each video snippet is used, that is I_i^m. Having collected the frames, we extract RGB and optical flow features using TBN [24] pre-trained for action recognition, while for object features we use the representation proposed by [14]. We elaborate on the feature extraction procedure in the implementation details.

3.3. Uni-modal Branch

Our proposed branch for uni-modal action anticipation is inspired by the temporal convolutional network (TCN) proposed in [27]. Similar to [27], the uni-modal branch stacks several layers of temporal convolutional residual blocks. An overview of the uni-modal branch is illustrated in Figure 2.

The first layer of the proposed branch is a one dimensional convolution with kernel size one, that adjusts the number of channels in the features of the input sequence to match the number of the features maps in the network. After that, the embedded sequence is processed by the residual block layers, that contain dilated one dimensional convolutional filters. These blocks are applied to the input hierarchically, meaning that each layer processes the output of the previous one.

All blocks follow the structure depicted in Figure 2. It consists of a temporal convolution with kernel size K and C convolutional filters. In our work, we set the kernel size $K = 3$ and the number of filters $C = 1024$. Depending on the layer number l, the convolutional filters within the residual blocks have different dilation factors. We increase the dilation factor for the convolutional kernels linearly with the number of layers (*i.e*. 1, 2, 3, 4).

Also, within each block, we normalize the output of the convolution using batch normalization [42] and apply

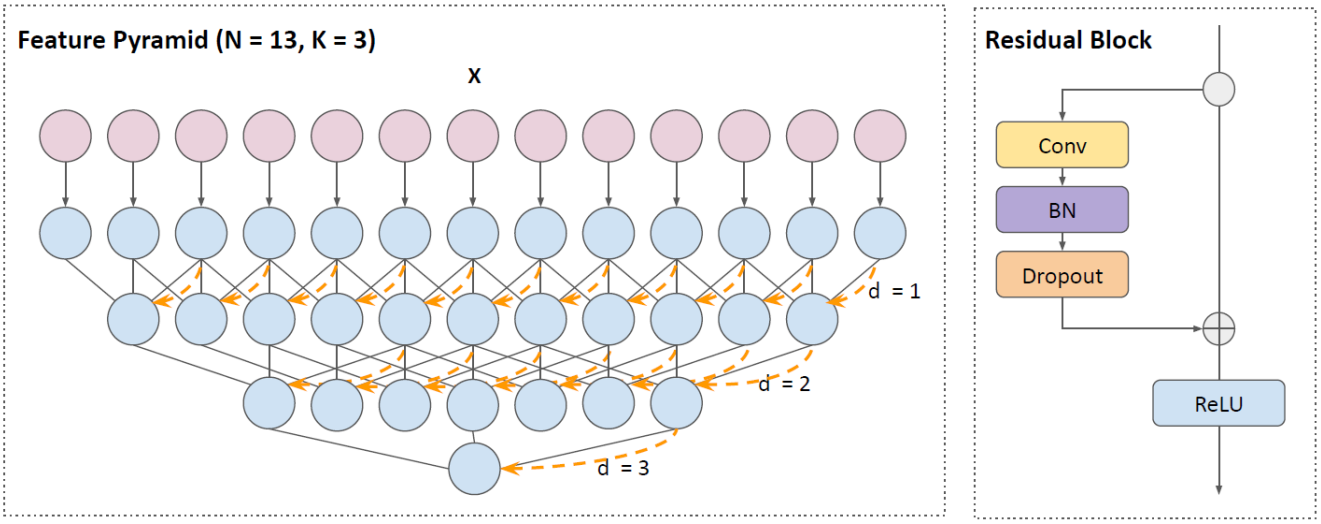

Figure 2. Architectural elements of our uni-modal branch. *(Left)* Our branch consists of a set of dilated temporal convolutions (with kernel size K) that process the input sequence (of length N) iteratively allowing our model to learn hierarchical feature representations. The increasing dilation factor allows to increase the receptive field of the model without increasing its depth. *(Right)* Overview of the convolutional block with the residual connection. Since the length of the input and output sequences for the residual blocks differ, we only consider the most recent elements of the input sequence for residual connections.

dropout [40,43] to avoid overfitting. We also apply dropout to the input of the first dimension-adjustment layer. Similar to [3], we used a spatial dropout, that is at each training step a whole channel across all time steps is zeroed out. To facilitate the gradient flow, we further introduce residual connections into the blocks that are followed by ReLU non-linearity. Since the temporal length of the output of the convolutional layer is less than the input sequence, we only use the most recent elements of the input sequence for the residual connections. Formally, the output at the l^{th} level is computed as follows:

$$\tilde{Z}_l = BN(W_l * Z_{l-1} + b_l)$$
$$\hat{Z}_l = Dropout(\tilde{Z}_l)$$
$$Z_l = ReLU(\hat{Z}_l + Z_{l-1}^{[N_{l-1}-N_l+1,\ldots,N_{l-1}]}),$$

where Z_l is the output of layer l, $*$ denotes the convolution operator with convolutional filters parametrized by a weight matrix $W_l \in \mathbb{R}^{K\times C\times C}$ and a bias vector $b_l \in \mathbb{R}^C$, $Z_{l-1}^{[N_{l-1}-N_l+1,\ldots,N_{l-1}]}$ is the sub-sequence of the most recent N_l elements of the sequence Z_{l-1}, where N_l denotes length of the sequence at level l.

Overall, the branch contains L layers of the previously discussed blocks. We set $L = 4$ for our model. Given an input sequence, we pass it through the hierarchy of the residual block layers until the whole sequence is summarized in a single feature vector F at the bottom of the pyramid. Based on the final feature vector F, we perform action anticipation using a fully-connected layer. In addition to the action classification layer, similar to [38], we also add two more layers for verb and noun classification that solve the auxiliary classification tasks to help anticipation.

3.4. Multi-Modal Fusion

Our approach fuses three modalities for anticipating the future action: RGB, flow, and object modalities. Figure 3 illustrates the proposed multi-modal fusion strategy. We fuse the branches by constructing a mutual multi-modal feature and use it for performing the final future prediction. To do so, we at first separately pre-train modality-specific branches for action anticipation. Then, we extract features F^{mod} from each modality by taking the output of the last convolutional block from the pre-trained branches.

We construct a cross-branch multi-modal feature by combining pairwise and mutual embeddings of the features computed by individual branches. To compute a pairwise embedding, we at first apply corresponding fully-connected layers to the pairwise concatenations of the features from the three branches. After that, these intermediate representations are merged by another feed-forward layer. A mutual embedding is constructed by projecting the concatenation of the features from the three branches with a fully-connected layer. The output dimension of both pairwise and mutual embeddings is 1024. Finally, the two computed embeddings are combined by taking their element-wise sum. Based on the resulting feature, three parallel fully-connected classification layers predict action, verb,

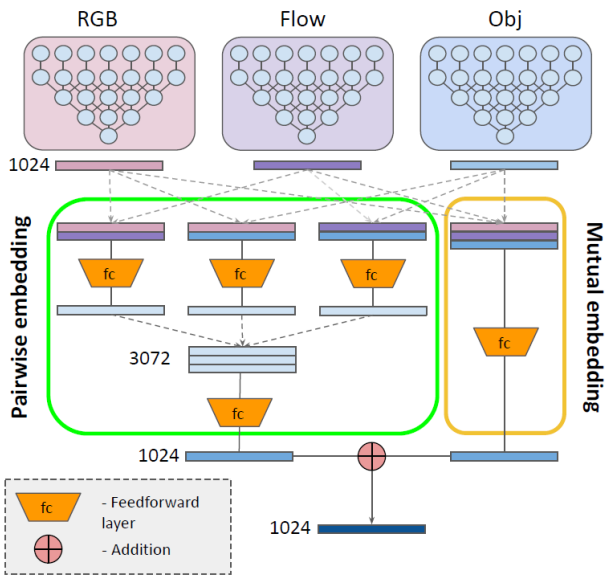

Figure 3. Overview of the multi-modal fusion strategy. Given the output of the individual uni-modal branches, we construct pairwise and mutual embeddings. Finally, these two embeddings are combined and passed to the final classification layers.

and noun, respectively. We demonstrate the effect of using different fusion strategies in the experimental section.

4. Experiments

4.1. Implementation Details

4.1.1 Features Extraction

To extract the RGB and optical flow features, we employ the TBN action recognition network [24]. We use the TBN pre-trained for action recognition, where for pre-training we follow the procedure recommended in [24]. After pre-training, similar to [14], we take RGB frames and stacks of 5 horizontal and 5 vertical optical flow frames of size 456×256 and pass them to the network to extract features. As features, we consider the output produced by the global average pooling layer of the BN-Inception network for the corresponding modality stream, with the resulting representation for each modality containing 1024 channels.

For object features, we use the representation proposed by [14]. They are extracted using a Faster R-CNN [37] object detector with ResNet-101 backbone [17]. For all modalities, the corresponding features extraction models are fixed and not fine-tuned for the anticipation task.

4.1.2 Training Details

We implemented our model using the Pytorch framework [35]. For optimization we use Stochastic Gradient Descent (SGD) with momentum equal to 0.9 and weight decay of $5 \cdot 10^{-4}$. We trained both the uni-modal branches and the fusion layers for 80 epochs. At first, we separately pre-train uni-modal branches for action anticipation. Then, during fusion, we fix the weights of the individual branches, and optimize only the paramters of the fusion layers.

For EPIC-Kitchens-55, we use a batch size of 64 examples. The starting learning rate is set to 0.005 for the object branch and 0.0005 for the RGB and flow branches as well as for the fusion layers. At each epoch we adjust the learning rate l according to the following schedule: $(1 - \frac{e}{E})^{0.99}$, where e denotes the number of the current epoch and E is the total number of epochs. For the residual blocks within the individual branches we use a dropout ratio of 0.5. We also apply dropout of 0.3 to the input sequence, before applying the first layer. Fully-connected classification layers for uni-modal and multi-modal predictions have dropout rates of 0.7 and 0.8, respectively.

For EPIC-Kitchens-100, we use a batch size of 128. We use the same starting learning rates for the uni-modal branches, while for the fusion layers we increase the learning rate to 0.00075. The schedule for the update of the learning rate remains the same. The dropout ratio for the blocks within the individual branches is decreased to 0.3 since the dataset is larger and thus the model is less prone to overfitting. As previously, we also apply dropout of 0.3 to the input sequence. Fully-connected classification layers for both uni-modal and multi-modal predictions have the dropout rate of 0.7.

For reporting results on the test sets, we use models that are pre-trained on both validation and train splits, after optimizing for hyper-parameters on the validation split.

4.2. Datasets

We perform our experiments on two large-scale datasets of egocentric videos: EPIC-Kitchens-55 [6] and EPIC-Kitchens-100 [8].

EPIC-Kitchens-55 contains videos collected by 32 participants, capturing their daily kitchen activities, including cooking, cleaning, doing laundry, etc. The RGB frames and pre-computed optical flow frames are publicly available with the dataset. In total, there are 55 hours of recordings and 39596 action annotations. In annotations, there are 125 verb and 352 noun classes. For action classes, similar to [14], we considered all unique (verb, noun) pairs that are present in the public training set. This amounts to the total of 2513 action classes.

For evaluation purposes, the authors of the dataset defined two test splits: *seen* and *unseen* kitchens. The seen split (S1) contains videos from the kitchens present both in training and test sets, while the unseen test split (S2) contains videos only from such kitchens that have not been observed during training. As a validation set, we use the

same subset of training videos as proposed by [14], who created training and validation sets from the publicly available training set by randomly choosing 232 and 40 videos for each set, respectively.

EPIC-Kitchens-100 [8] extends upon EPIC-Kitchens-55 dataset, with the total of 100 hours of footage. Videos in EPIC-Kitchens-100 were collected by 37 participants in 45 environments. It contains 89977 fine-grained action annotations, with 97 verb and 300 noun classes. By using the same principle as for EPIC-Kitchens-55, there are 3806 action classes. All videos in the dataset are split into train, validation and test sets with a ratio of approximately 75/10/15. The validation and test splits contain two subsets, on which the results are reported separately: unseen participants and tail classes. The unseen subset contains videos of participants that are not present in the train set. The subset of tail classes for verbs and nouns contains the set of classes that have the fewest instances and that account for 20% of the instances in the whole dataset. An action class is considered to be a tail class, if it contains either a tail noun or verb. Overall, there are 86/228/3729 verb/noun/action classes.

4.3. Evaluation Metrics

For both EPIC-Kitchens-55 and EPIC-Kitchens-100, we evaluate our approach using the official dataset metrics. For EPIC-Kitchens-55, we use top-1 and top-5 verb, noun and action accuracy (the prediction is deemed correct if the ground-truth action falls into the top-1 or top-5 predictions, respectively). For EPIC-Kitchens-100, we report class-mean top-5 recall.

4.4. Anticipation Results on EPIC-Kitchens-55

We compare our proposed model to the state-of-the-art methods on the test splits of the EPIC-Kitchens-55 dataset in Table 1. In our model, the RGB and optical flow features are extracted using TBN. For a fair comparison with the methods that use TSN features proposed by [14], namely [14, 38, 48], we also trained a separate model for which we used appearance and motion features provided by the authors. As one can see, both models perform similarly on the unseen test split, with the top-1 action accuracy of 8.9%, while on the seen test split, the model trained with the TBN features outperforms the model trained with the TSN features by a margin of 0.5%. Comparing to other methods, on the seen test split, both models trained with TSN and TBN features outperform LSTM-based methods in top-1 action accuracy: RU-LSTM by 0.5% and 1.0% accordingly and ImagineRNN [48] by 0.2% and 0.7%. On the unseen test split, we outperform RU-LSTM by 0.7%, while ImagineRNN performs better by 0.4%.

Apart from achieving similar or better accuracy, our proposed model is also more efficient during training and inference stages. In Table 2 we compare average training time per epoch and time required for inference on the validation set for our RGB branch and that of RU-LSTM. For measuring training and inference times of the RU-LSTM network, we used the official code made publicly available by the authors. Since RU-LSTM makes predictions at several time steps, to ensure a fair comparison, we performed an experiment where we modified the RU-LSTM training procedure to make predictions at the anticipation time of one second only. This however, resulted in a decrease of both Top-1 and Top-5 accuracy of the model. Therefore, we compare the training time of our method to the original setup of RU-LSTM. Then, for measuring the time required for inference, we considered predictions made by RU-LSTM only for the anticipation time of one second, without performing computations for different anticipation times. To further minimize the effect of factors not related to the direct effectiveness of the models, for both methods we used TSN features, identical training and validation sets, as well as the same GPU and data loading procedure. We conducted the experiments using an Nvidia Titan Xp GPU.

As shown in Table 2, our method is more than two times faster than RU-LSTM during training and almost two times faster during inference stage. Apart from shorter per-epoch training time, our method also does not use an additional teacher-forcing pre-training stage, as well as the total number of epochs required for convergence is 80 compared to 100 for the RU-LSTM. So, all things considered, our proposed branch can be trained approximately five times faster. ImagineRNN builds upon the RU-LSTM baseline by extending it with contrastive learning, while the underlying architecture, along with training and inference procedures are identical to those of RU-LSTM, except for the absence of the additional pre-training stage. Therefore, based on the measurements made for the RU-LSTM, we can expect that both methods require a similar amount of time per-epoch for training and inference. Thus, we can also expect our method to be faster than ImagineRNN. Finally, our model is also more effective than LSTM-based models in terms of memory-usage. Our branch needs less than 60% of the memory requirement for the one of the RU-LSTM. Higher memory efficiency of convolution-based networks over RNNs has also been discussed in [3].

Concerning the other methods, our model performs on par with Liu *et al*. [28] on the seen test split, while Ego-OMG and Sener *et al*. [38] outperform our approach in top-1 action accuracy by 0.6% and 1.2% respectively. On the unseen test split, Liu *et al*. [28], Ego-OMG and Sener *et al*. [38] perform better than our model by 1.0%, 2.9% and 1.2% respectively. Notice, however, that both Ego-OMG [10] and Liu *et al*. [28] use additional annotations to train their proposed approaches. Ego-OMG uses additional supervision in the form of progression time of directed hand movements, as well as ground truth segmentation masks of

		Top-1 Accuracy (%)			Top-5 Accuracy (%)		
	Method	Verb	Noun	Action	Verb	Noun	Action
S1	TSN [6]	31.8	16.2	6.0	76.6	42.2	28.2
	Miech *et al*. [33]	30.7	16.5	9.7	76.2	42.7	25.4
	RU-LSTM [14]	33.0	22.8	14.4	79.6	50.9	33.7
	ImagineRNN [48]	35.4	22.8	14.7	79.7	52.1	34.9
	Liu *et al*. [28]	36.3	23.8	15.4	79.2	51.9	34.3
	Ego-OMG [9]	32.2	24.9	16.0	77.4	50.2	34.5
	Sener *et al*. [38]	37.9	24.1	**16.6**	79.7	54.0	**36.1**
	Ours (TSN)	36.7	22.9	14.9	79.6	51.2	33.6
	Ours (TBN)	37.2	23.7	15.4	79.5	51.9	34.4
S2	TSN [6]	25.3	10.4	2.4	68.3	29.5	6.6
	Miech *et al*. [33]	28.4	12.4	7.2	69.9	32.2	19.3
	RU-LSTM [14]	27.0	15.2	8.2	69.6	34.4	21.1
	ImagineRNN [48]	29.3	15.5	9.3	70.7	35.8	22.2
	Liu *et al*. [28]	29.9	16.8	9.9	71.8	38.9	23.7
	Sener *et al*. [38]	29.5	16.5	10.1	70.1	37.8	23.4
	Ego-OMG [9]	27.4	17.7	**11.8**	68.6	37.9	**23.8**
	Ours (TSN)	29.3	15.2	8.9	71.2	36.8	21.0
	Ours (TBN)	30.7	14.9	8.9	72.0	36.7	21.7

Table 1. Results for action anticipation on the EPIC-Kitchens-55 seen (S1) and unseen (S2) test splits at anticipation time $T_a = 1$ second.

	Time (sec)	
Method	Training	Inference
RU-LSTM [14]	27.7	$3.8 \cdot 10^{-4}$
Ours	12.4	$2.0 \cdot 10^{-4}$

Table 2. Average training time per epoch and inference time on EPIC-Kitchens-55. 'Training' represents training time on the training split. 'Inference' represents average inference time per sample on the validation split.

interaction objects, while Liu *et al*. [28] use additional annotations for interaction hotspots and hand trajectories.

4.5. Anticipation Results on EPIC-Kitchens-100

We compare our proposed model to the baseline methods on the test set of the EPIC-Kitchens-100 dataset in Table 3. Since EPIC-Kitchens-100 has been introduced only recently, for many of the previously mentioned methods no evaluation results are available. Therefore, we compare our method to the officially reported baselines.

As the table shows, our approach performs on par with RU-LSTM on both overall and tail-class splits. The performance is also consistent using different types of features and our model works well with both TBN and TSN features. Furthermore, our approach shows better generalization behavior as indicated by the results on the unseen environments. Our approach outperforms RU-LSTM in the mean top-5 action recall on the unseen split by 2.5% and 1.4% using TBN and TSN features, respectively.

4.6. Ablation Study

In this section, we provide a set of ablation experiments to analyze the different components of our approach. As our main motivation is to develop a model that has a good trade-off between accuracy and efficiency for the task of action anticipation, we verify how much past is really necessary for the network to achieve a good accuracy. Additionally, we also study different fusion strategies for the uni-modal branches. For the ablation experiments, we report results on the validation set of the EPIC-Kitchens-55 dataset using TBN features. As previously, we use the validation set constructed by [14].

4.6.1 Observation length

We report the top-1 action accuracy of our RGB branch trained on observation intervals of different lengths in Table 4. We vary the observation length starting from 0.75 seconds up to 7.75 seconds. As shown in the table, the accuracy of the predictions increases with the length of the observation interval until it saturates at 5.25 seconds. Similar findings have been reported in [38]. Based on this observation, we fix the length of the observed interval to 5.25 seconds which corresponds to an input sequence of 21 snippets.

4.6.2 Fusion strategy

We report the top-1 action accuracy of the individual branches and different multi-modal fusion strategies in Ta-

Mean Top-5 Recall									
	Overall (%)			Unseen (%)			Tail (%)		
Method	Verb	Noun	Act.	Verb	Noun	Act.	Verb	Noun	Act.
Random	6.2	2.3	0.1	8.1	3.3	0.3	1.9	0.7	0.03
RU-LSTM [14]	25.3	26.7	**11.2**	19.4	26.9	9.7	17.6	15.9	**7.9**
Ours (TSN)	20.4	26.6	10.9	17.9	26.9	11.1	11.7	15.2	7.0
Ours (TBN)	21.5	26.8	11.0	20.8	28.3	**12.2**	13.2	15.4	7.2

Table 3. Results for action anticipation at anticipation time $T_a = 1$ on the EPIC-Kitchens-100 test set.

No. of Snippets	Obs. Time (sec)	Top-1 Act. Acc (%)
3	0.75	11.1
7	1.75	11.5
13	3.25	11.8
21	5.25	**12.4**
31	7.75	**12.4**

Table 4. Effect of the observation length on the prediction accuracy. We report top-1 action accuracy for the RGB branch using TBN features on the validation set of EPIC-Kitchens-55.

ble 5. Among the uni-modal branches, the appearance branch has the highest top-1 action accuracy, whereas the flow branch has the lowest. All fusion schemes improve over the performance of uni-modal branches. In total, we experimented with five different fusion methods:

- **Late fusion**: Averaging predictions made by individual branches.
- **Attention fusion**: Learning weights for predictions of the individual branches based on the final features from the three branches: $\{F^{RGB}, F^{Flow}, F^{Object}\}$.
- **Mutual fusion**: Applying only the mutual fusion path from the proposed fusion scheme (see Figure 3).
- **Pairwise fusion**: Applying only the pairwise fusion path from the proposed fusion scheme (see Figure 3).
- **Mutual + Pairwise**: Applying both pairwise and mutual fusion paths and merging them with element-wise addition (see Figure 3).

We observed that merging predictions of individual branches either via late fusion (top-1 action accuracy 14.1%) or attention fusion (top-1 action accuracy 14.3%) achieves lower results than merging individual features of the uni-modal branches and making a cross-branch prediction based on the constructed representations. Furthermore, learning both pairwise and mutual embeddings of the uni-modal features and combining them via the element-wise addition is better than making the final prediction based only either on the pairwise or mutual embeddings.

Fusion	Top-1 Act. Acc (%)
RGB	12.4
Flow	8.9
Obj	10.9
Late fusion	14.1
Attention	14.3
Mutual feature fusion	14.7
Pairwise feature fusion	14.6
Mutual + Pairwise feature fusion	**14.9**

Table 5. Comparison of different multi-modal fusion methods on the EPIC-Kitchens-55 validation set.

5. Conclusion

In this work, we proposed a multi-modal architecture based on temporal convolutional layers for the short-term action anticipation task. Instead of relying on recurrent layers for temporal modelling, we use a stack of temporal convolutional layers, which allows our approach to perform anticipation faster. We further proposed a multi-modal fusion strategy that combines both mutual and pairwise interactions between the different branches. Results on two large-scale datasets of egocentric videos, EPIC-Kitchens-55 and EPIC-Kitchens-100, show that our approach achieves performance comparable to the state-of-the-art approaches while being at least two times faster and more efficient compared to RNN-based approaches.

6. Acknowledgements

This work has been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – GA 1927/4-2 (FOR 2535 AnticipatingHuman Behavior) and the ERC Starting Grant ARCA (677650).

References

[1] Yazan Abu Farha, Alexander Richard, and Juergen Gall. When will you do what? - anticipating temporal occurrences of activities. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[2] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In

IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[3] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, 2018.

[4] Apratim Bhattacharyya, Mario Fritz, and Bernt Schiele. Bayesian prediction of future street scenes using synthetic likelihoods. In *International Conference on Learning Representations (ICLR)*, 2019.

[5] Guglielmo Camporese, Pasquale Coscia, Antonino Furnari, G. Farinella, and Lamberto Ballan. Knowledge distillation for action anticipation via label smoothing. *ArXiv*, abs/2004.07711, 2020.

[6] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The epic-kitchens dataset. In *European Conference on Computer Vision (ECCV)*, 2018.

[7] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. The epic-kitchens dataset: Collection, challenges and baselines. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020.

[8] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Antonino Furnari, Jian Ma, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision. *CoRR*, abs/2006.13256, 2020.

[9] Eadom Dessalene, Chinmaya Devaraj, Michael Maynord, Cornelia Fermuller, and Yiannis Aloimonos. Forecasting action through contact representations from first person video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–1, 2021.

[10] Eadom Dessalene, Michael Maynord, Chinmaya Devaraj, C. Fermüller, and Y. Aloimonos. Egocentric object manipulation graphs. *ArXiv*, abs/2006.03201, 2020.

[11] Yazan Abu Farha, Qiuhong Ke, Bernt Schiele, and Juergen Gall. Long-term anticipation of activities with cycle consistency. In *DAGM German Conference on Pattern Recognition (GCPR)*, 2020.

[12] David F. Fouhey and C. Lawrence Zitnick. Predicting object dynamics in scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[13] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[14] Antonino Furnari and Giovanni Maria Farinella. What would you expect? anticipating egocentric actions with rolling-unrolling lstms and modality attention. In *International Conference on Computer Vision (ICCV)*, 2019.

[15] Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. Forecasting future action sequences with neural memory networks. In *British Machine Vision Conference (BMVC)*, 2019.

[16] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. In *International Conference on Machine Learning (ICML)*, 2017.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[18] Alejandro Hernandez, Juergen Gall, and Francesc Moreno-Noguer. Human motion prediction via spatio-temporal inpainting. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 1997.

[20] Ashesh Jain, Avi Singh, Hema S Koppula, Shane Soh, and Ashutosh Saxena. Recurrent neural networks for driver activity anticipation via sensory-fusion architecture. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[21] Ashesh Jain, Amir R. Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[22] Xiaojie Jin, Huaxin Xiao, Xiaohui Shen, Jimei Yang, Zhe Lin, Yunpeng Chen, Zequn Jie, Jiashi Feng, and Shuicheng Yan. Predicting scene parsing and motion dynamics in the future. In *Advances in Neural Information Processing Systems 30 (NIPS)*. 2017.

[23] Zhenheng Yang Jiyang Gao and Ram Nevatia. Red: Reinforced encoder-decoder networks for action anticipation. In *British Machine Vision Conference (BMVC)*, 2017.

[24] Evangelos Kazakos, Arsha Nagrani, Andrew Zisserman, and Dima Damen. Epic-fusion: Audio-visual temporal binding for egocentric action recognition. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[25] Qiuhong Ke, Mario Fritz, and Bernt Schiele. Time-conditioned action anticipation in one shot. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[26] Kris M. Kitani, Brian D. Ziebart, James Andrew Bagnell, and Martial Hebert. Activity forecasting. In *European Conference on Computer Vision (ECCV)*, 2012.

[27] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[28] Miao Liu, Siyu Tang, Yin Li, and James M. Rehg. Forecasting human-object interaction: Joint prediction of motor attention and actions in first person video. In *European Conference on Computer Vision (ECCV)*, 2020.

[29] Pauline Luc, Camille Couprie, Yann LeCun, and Jakob Verbeek. Predicting future instance segmentation by forecasting convolutional features. In *European Conference on Computer Vision (ECCV)*, 2018.

[30] Pauline Luc, Natalia Neverova, Camille Couprie, Jacob Verbeek, and Yann LeCun. Predicting deeper into the future of semantic segmentation. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[31] Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[32] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. In *International Conference on Learning Representations (ICLR)*, 2016.

[33] Antoine Miech, Ivan Laptev, Josef Sivic, Heng Wang, Lorenzo Torresani, and Du Tran. Leveraging the present to anticipate the future in videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[34] Seyed Shahabeddin Nabavi, Mrigank Rochan, and Yang Wang. Future semantic segmentation with convolutional lstm. In *Biritsh Machine Vision Conference (BMVC)*, 2018.

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NIPS)*. 2019.

[36] Marc'Aurelio Ranzato, Arthur Szlam, Joan Bruna, Michaël Mathieu, Ronan Collobert, and S. Chopra. Video (language) modeling: a baseline for generative models of natural videos. *ArXiv*, abs/1412.6604, 2014.

[37] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*. 2015.

[38] Fadime Sener, Dipika Singhania, and Angela Yao. Temporal aggregate representations for long-range video understanding. In *European Conference on Computer Vision (ECCV)*, 2020.

[39] Fadime Sener and Angela Yao. Zero-shot anticipation for instructional activities. *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[40] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 2014.

[41] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using LSTMs. In *International conference on machine learning (ICML)*, 2015.

[42] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[43] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[44] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alexander Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *Arxiv*, 2016.

[45] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Anticipating visual representations from unlabeled video. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[46] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Val Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision (ECCV)*, 2016.

[47] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[48] Yu Wu, Linchao Zhu, Xiaohan Wang, Yi Yang, and Fei Wu. Learning to anticipate egocentric actions by imagination. *IEEE Transactions on Image Processing*, 30:1143–1152, 2021.

[49] Brian D. Ziebart, Nathan Ratliff, Garratt Gallagher, Christoph Mertz, Kevin Peterson, J. Andrew Bagnell, Martial Hebert, Anind K. Dey, and Siddhartha Srinivasa. Planning-based prediction for pedestrians. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.