KNOWLEDGE DISTILLATION FOR HUMAN ACTION ANTICIPATION

Vinh Tran, Yang Wang, Zekun Zhang, Minh Hoai

Stony Brook University, Stony Brook, NY 11790

ABSTRACT

We consider the task of training a neural network to anticipate human actions in video. This task is challenging given the complexity of video data, the stochastic nature of the future, and the limited amount of annotated training data. In this paper, we propose a novel knowledge distillation framework that uses an action recognition network to supervise the training of an action anticipation network, guiding the latter to attend to the relevant information needed for correctly anticipating the future actions. This framework is possible thanks to a novel loss function to account for positional shifts of semantic concepts in a dynamic video. The knowledge distillation framework is a form of self-supervised learning, and it takes advantage of unlabeled data. Experimental results on JHMDB and EPIC-KITCHENS dataset show the effectiveness of our approach.

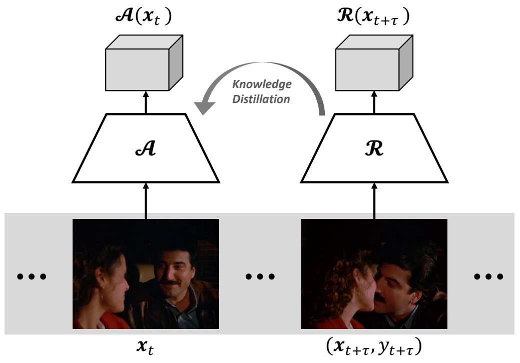

Fig. 1: **Knowledge Distillation for Action Anticipation.** We propose to first learn an action recognition model $\mathcal{R} : \mathbf{x}_{t+\tau} \rightarrow y_{t+\tau}$, then use $\mathcal{R}$ to supervise the training of $\mathcal{A}$ through a novel knowledge distillation framework.

1. INTRODUCTION

Human action anticipation is notoriously difficult due to the stochastic nature of the future. Given what is occurring or what can be observed in a video at the current moment, there are multiple possibilities that can happen. Thus, there is a fundamental limit to what we can anticipate, even when we have an infinite amount of training data. In practice, the amount of annotated training data is limited, so anticipation is a much harder problem.

One common approach to address anticipation is to use supervised learning (e.g. [1–5]), but learning a direct mapping between distant time steps can be challenging due to the weak correlation between the time steps. Suppose we are interested in anticipation with the lead time τ, we can used supervised learning and train a neural network to map from the video observation *up until* time t (denoted $\mathbf{x}_t$) to the human action label $y_{t+\tau}$ at time $t+\tau$. That is to use a set of annotated training data pairs $\{\mathbf{x}_t, y_{t+\tau}\}$ to train a network $\mathcal{A} : \mathbf{x}_t \rightarrow y_{t+\tau}$. To some extent, the training of the anticipation network $\mathcal{A}$ can be done similarly to the training of a recognition network $\mathcal{R}$ that maps from $\mathbf{x}_{t+\tau}$ to $y_{t+\tau}$, with the only difference being that the input to $\mathcal{A}$ is $\mathbf{x}_t$ while the input to $\mathcal{R}$ is $\mathbf{x}_{t+\tau}$. In general, the correlation between $\mathbf{x}_t$ and $y_{t+\tau}$ is weaker than the correlation between $\mathbf{x}_{t+\tau}$ and $y_{t+\tau}$, so the asymptotic performance of $\mathcal{A}$ is expected to be lower than the asymptotic performance of $\mathcal{R}$. Furthermore, $\mathcal{A}$ will converge to its asymptotic performance slower than $\mathcal{R}$. This is due to the complexity of video data, and it will take much training data to separate the relevant features from the irrelevant ones. This separation task is harder for training the anticipation network than for training the recognition network due to the higher ratio of irrelevant features. In general, it will require more training data to get the anticipation network $\mathcal{A}$ to "attend" to the relevant features.

We propose a framework to train the anticipation network to attend to the same type of information that is being attended by the recognition network when making classification decisions. Our framework leverages the abundance of (unlabeled) data, improving the generalization ability of an anticipation network without requiring additional human annotation. However, due to dynamic environment in a video, we cannot use L_2 loss to force the one-to-one mapping [2] between elements of two activated feature maps at t and $t + \tau$. Inspired by [6, 7], we propose a novel attention mechanism that does not require pixel-to-pixel correspondence between two input videos or between two feature maps.

Experiments on three datasets show that the proposed knowledge distillation framework improves the performance of the anticipation network. The level of improvement is consistent with the level of improvement obtained as if the annotated training data is doubled.

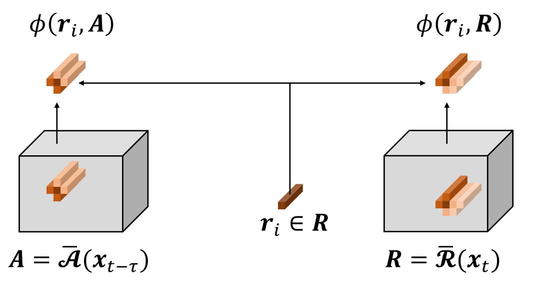
$\phi(r_i, A)$
$\phi(r_i, R)$
$r_i \in R$
$A = \bar{\mathcal{A}}(x_{t-\tau})$
$R = \bar{\mathcal{R}}(x_t)$

Fig. 2: **Knowledge distillation for weakly aligned feature maps.** To perform knowledge distillation for weakly aligned feature maps, we propose an attentional pooling operator ϕ to compute the amount of $\mathbf{r}_i$ within $\mathbf{R}$ and $\mathbf{A}$ respectively, then minimize $\sum_i \|\phi(\mathbf{r}_i, \mathbf{R}) - \phi(\mathbf{r}_i, \mathbf{A})\|_2^2$.

2. FUTURE KNOWLEDGE DISTILLATION

In this section, we describe a knowledge distillation framework that uses a recognition network to guide the training of an anticipation network, leveraging the abundance of unlabeled data.

2.1. Framework overview

Suppose the desired anticipation lead time is τ, our goal is to train an anticipation network $\mathcal{A}$ to map from the input video segment at time $t - \tau$ (denoted $\mathbf{x}_{t-\tau}$) to the human action label y_t at time t. That is to train $\mathcal{A}$ so that $\mathcal{A}(\mathbf{x}_{t-\tau}) = y_t$. We assume there is a recognition network $\mathcal{R}$ to recognize the action class of an observed video clip, predicting y_t from $\mathbf{x}_t$.

Let $\bar{\mathcal{A}}(\mathbf{x})$ denote the feature vector/map at a particular layer of the anticipation network for the input video $\mathbf{x}$. Similarly, let $\bar{\mathcal{R}}(\mathbf{x})$ be the feature vector/map of the recognition network for the input video $\mathbf{x}$. Let $\mathcal{S}$ be the set of time indexes where the frames are annotated with human action labels; t is in $\mathcal{S}$ if y_t is available. One approach for training the anticipation network is to minimize the following classification loss defined on annotated training data as $\sum_{t\in\mathcal{S}} \mathcal{L}_c(\mathcal{A}(\mathbf{x}_{t-\tau}), y_t)$. Here, $\mathcal{L}_c$ is a loss function that penalizes the difference between the prediction output $\mathcal{A}(\mathbf{x}_{t-\tau})$ and the actual class label y_t, e.g., using the negative log likelihood loss.

Let $\mathcal{U}$ be the set of time indexes t's where y_t is not available (i.e., unlabeled data). Our knowledge distillation framework optimizes the below loss function:

$$\sum_{t\in\mathcal{S}} \mathcal{L}_c(\mathcal{A}(\mathbf{x}_{t-\tau}), y_t) + \sum_{t\in\mathcal{U}} \mathcal{L}_c(\mathcal{A}(\mathbf{x}_{t-\tau}), \mathcal{R}(\mathbf{x}_t)) \quad (1)$$

$$+ \lambda \sum_{t\in\mathcal{U}\cup\mathcal{S}} \mathcal{L}_d(\bar{\mathcal{A}}(\mathbf{x}_{t-\tau}), \bar{\mathcal{R}}(\mathbf{x}_t)). \quad (2)$$

The above objective function trains the anticipation network $\mathcal{A}$ to output the same output as the recognition network on the unlabeled data $\mathcal{U}$. Furthermore, $\mathcal{L}_d$ is a loss function that measures the discrepancy between two feature maps $\bar{\mathcal{A}}(\mathbf{x}_{t-\tau})$ and $\bar{\mathcal{R}}(\mathbf{x}_t)$. This loss trains the anticipation network to produce the same feature map as the feature map of the recognition network. This formulation uses unlabeled data and the distilled knowledge from the recognition network to guide the anticipation network to attend to the relevant information that is useful for categorizing the future action.

2.2. Distillation loss

We now describe the loss function $\mathcal{L}_d$ for measuring the differences between two activation feature maps. At first glance, a reasonable option for this loss function is to use the sum of squared differences between the elements of the two feature maps. However, this loss assumes perfect correspondence between the elements of the feature maps. This is too restrictive, as will be explained below.

We use convolutional architectures for the anticipation and recognition networks, and the feature maps $\bar{\mathcal{A}}(\mathbf{x})$ and $\bar{\mathcal{R}}(\mathbf{x})$ are typically 4D tensors: $\bar{\mathcal{A}}(\mathbf{x}), \bar{\mathcal{R}}(\mathbf{x}) \in \Re^{l\times h\times w\times d}$. Usually, l, h, and w can be obtained by dividing the length, the height, and the width of the video $\mathbf{x}$ by their effective convolutional strides respectively. d is the number of channels of the feature map.

Consider a particular video segment $\mathbf{x}_t$, the feature map $\bar{\mathcal{R}}(\mathbf{x}_t)$ encodes the activated features important for recognizing the human action. For example, in order for the recognition network to recognize a "wash a dish" action, some part of the feature map might indicate the presence of the dish in the video. Arguably, for the anticipation network to successfully anticipate the "wash a dish" action, there must be some activated "dish" features in its feature map. The knowledge distillation framework encourages that by training the anticipation network to output the 'same' feature map as the recognition network. However, it would be unreasonable to assume the "dish" feature to stay at the same spatiotemporal location of the feature map. More generally, video is a dynamic environment, where important objects and other semantic entities might not remain at the same locations, as illustrated in Fig. 1. Thus, it is unreasonable to use the sum of squared differences to measure the discrepancy between two feature maps of two different time steps.

For brevity, let us reshape the 4D tensors $\bar{\mathcal{A}}(\mathbf{x}_{t-\tau})$ and $\bar{\mathcal{R}}(\mathbf{x}_t)$ to 2D matrices $\mathbf{A}$ and $\mathbf{R}$, $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_{lhw}] \in \Re^{d\times lhw}$, $\mathbf{R} = [\mathbf{r}_1, \ldots, \mathbf{r}_{lhw}] \in \Re^{d\times lhw}$. One naive approach is to directly minimize $\mathcal{L}_{direct}(\mathbf{A}, \mathbf{R}) = \|\mathbf{A} - \mathbf{R}\|_2^2$. However, it would be unreasonable because the features in $\mathbf{A}$ and $\mathbf{R}$ might be at different spatiotemporal locations after τ seconds. Instead, for each vector $\mathbf{a}_i$, we measure the similarity between $\mathbf{a}_i$ with all vectors in $\mathbf{R}$ and we compute a vector quantity to represent the amount of $\mathbf{a}_i$ in $\mathbf{R}$: $\phi(\mathbf{a}_i, \mathbf{R}) = \sum_{k=1}^{lhw} \omega_k \mathbf{r}_k$, with $\omega_k = \frac{1}{Z}\exp(\alpha \mathbf{r}_k^T \mathbf{a}_i)$, where Z is the normalizing constant so that the sum of ω_k's is 1. Here, α is the hyper-parameter that controls the pooling weights. The default value of α is set to $\frac{1}{\sqrt{d}}$, where d is the number of channels of the feature maps. If the value of α is small, the weights associated to each vector are almost equal, and $\phi(\mathbf{a}_i, \mathbf{R})$ is the average pooling of vectors in $\mathbf{R}$. On the other hand, this operator is similar to max pooling if we use a large value for α, and $\phi(\mathbf{a}_i, \mathbf{R})$ is the vector in $\mathbf{R}$ that is most similar to $\mathbf{a}_i$. Equivalently, $\phi(\mathbf{a}_i, \mathbf{R})$ and $\phi(\mathbf{a}_i, \mathbf{A})$ can be expressed in the form: $\phi(\mathbf{a}_i, \mathbf{R}) = \mathbf{R}\,\mathrm{softmax}(\alpha \mathbf{R}^T \mathbf{a}_i)$, and $\phi(\mathbf{a}_i, \mathbf{A}) = \mathbf{A}\,\mathrm{softmax}(\alpha \mathbf{A}^T \mathbf{a}_i)$. We define the loss for the differences between two feature maps $\mathbf{A}$ and $\mathbf{R}$ as follows:

$$\mathcal{L}_d(\mathbf{A}, \mathbf{R}) = \tilde{\mathcal{L}}_d(\mathbf{A}, \mathbf{R}) + \tilde{\mathcal{L}}_d(\mathbf{R}, \mathbf{A}), \tag{3}$$

$$\text{where} \quad \tilde{\mathcal{L}}_d(\mathbf{A}, \mathbf{R}) = \sum_{i=1}^{lhw} ||\phi(\mathbf{a}_i, \mathbf{R}) - \phi(\mathbf{a}_i, \mathbf{A})||_2^2, \tag{4}$$

$$\tilde{\mathcal{L}}_d(\mathbf{R}, \mathbf{A}) = \sum_{i=1}^{lhw} ||\phi(\mathbf{r}_i, \mathbf{A}) - \phi(\mathbf{r}_i, \mathbf{R})||_2^2. \tag{5}$$

3. EXPERIMENTS

3.1. Datasets

We conducted the main experiments on two challenging datasets: JHMDB [8] and EPIC-KITCHENS [9]. We also performed some controlled experiments on the THUMOS dataset [10] to understand the expected benefits of having extra supervision.

3.2. Experiments on the JHMDB dataset

We performed several experiments on the JHMDB dataset. We used the I3D network as the backbone for this task. We followed the standard protocol [11] for evaluation on this dataset, i.e., using only the first 20% of the frames to predict action class labels. During training, we combined both the classification loss $\mathcal{L}_c$ (using the class labels or the predicted class probability) and the attention loss $\mathcal{L}_d$ (using feature maps). We used KL divergence for the classification loss. For the attention loss, we used Huber loss (with $\delta = 1$). Both RGB frames and optical flow maps were used to train an anticipation network.

We report the action recognition and anticipation performance of different methods in Tab. 1. First, we trained the action recognition network using all the available frames in the training set. Directly applying the recognition network on the first 20% frames of the test videos (i.e., using the recognition network for the anticipation task), the accuracy dropped drastically to 74.9%. Second, we applied the direct loss (denoted as $\mathcal{L}_{direct}$ as in Tab. 1) between two feature maps produced by anticipation network and recognition network. With this additional loss, the accuracy was increased to 75.5%. This was possibly thanks to the small displacement between two feature maps since the dataset contains only action and the 15 frames anticipation is short. Hence, the $\mathcal{L}_{direct}$ loss also helped improving the recognition performance on this dataset.

Method	RGB	Flow	Both
Recognition Network			
I3D	75.3	77.8	83.9
Anticipation Network			
I3D	69.0	64.4	74.9
I3D + $\mathcal{L}_{direct}$	69.5	67.1	75.5
I3D + $\tilde{\mathcal{L}}_d(\mathbf{R}, \mathbf{A})$	70.0	67.4	75.0
I3D + $\tilde{\mathcal{L}}_d(\mathbf{A}, \mathbf{R})$	69.5	67.5	75.8
I3D + $\mathcal{L}_d(\mathbf{A}, \mathbf{R})$	**70.2**	**67.7**	**76.6**

Table 1: **Action anticipation results on the JHMDB dataset.** The recognition network uses the entire video for classification while the anticipation network only observes the first 20% of the video.

Method	Acc(%)
Where/What [12]	10.0
Context-fusion [13]	28.0
Within-class Loss [14]	33.0
ELSTM [15]	55.0
FDI [16]	61.0
FM-RNN [11]	73.4
I3D + Knowledge Distillation (Ours)	**76.6**

Table 2: **Comparison of action anticipation methods on the JHMDB dataset.** All methods use the first 20% of the video for prediction.

Replacing the direct loss function with our distillation loss, the performance increased to 75.8%. Finally, we achieved the best performance of 76.6% when using the symmetric bidirectional attention loss $\mathcal{L}_d(A, R)$. As shown in Tab. 2, we obtained the new state of the art result on the JHMDB dataset.

3.3. Experiments on the Epic-Kitchens dataset

We used the I3D network architecture for the experiments described in this subsection, as in the previous subsection. Since Epic-Kitchen is a large dataset, we used the feature maps extracted from the `MaxPool3d_4a_3x3` layer as the input to the network instead of training directly from video frames.

Collecting unlabeled training data. We collected video clips from unlabeled segments as follows. First, we randomly took two video segments of 32 frames with the anticipation time τ=1s from unlabeled video segments. Second, we used the pre-trained I3D to extract feature maps at the `MaxPool3d_4a_3x3` layer for the two video segments. Third, we fed the feature map of the latter segment to the recognition network and computed its visual representation (i.e., both class probabilities and feature maps). Finally, the feature map of the first segment and the visual representation of the second segment formed a data-pair sample for training the anticipation network. We collected a total of $26,391$ un-

Method	Acc.(%)
R(2+1)D + Vis. Attr. [17]	28.4
TSN-RGB [18]	28.5
TSM-RGB [19]	30.3
I3D [20]	30.1
I3D + Data augmentation	29.8
I3D + Additional data only (Ours)	31.4
I3D + Knowledge Distillation (Ours)	**31.8**

Table 3: **Accuracy of anticipation methods on the EPIC-KITCHENS dataset** (for anticipating the verb actions). All methods reported here are implemented by us, trained with the same amount of labeled data.

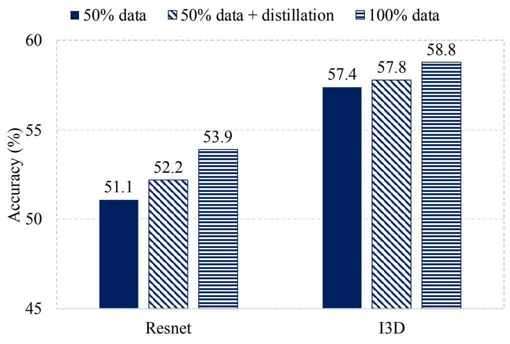

Fig. 3: **Expected performance gain when doubling the amount of training data.**

labeled training samples. Together with the 23,191 labeled examples, we had a total of 49,582 training samples. This is roughly double the amount of the original training data.

Experimental Results. We report all performance values in Tab. 3. When we trained a I3D network using the annotated training data, we achieved an accuracy of 30.1%. This was better than the performance of the TSN-RGB [18], which achieved 28.5% accuracy on the same dataset. We also trained another I3D network with augmented training data, where we used both the video segments prior to the actions and the video segments right at the time of the actions to train the network. However, this approach slightly decreased the performance of the anticipation network. This was perhaps due to the use of 'noisy data', since the video segments at the time of the actions were meant for the recognition network not the anticipation network. Using additional unlabeled data with pseudo label, we improved the accuracy to 31.4. Using knowledge distillation and additional unlabeled data, the obtained anticipation network obtained 1.7% improvement in accuracy (30.1 $\rightarrow$ 31.8). This was also better than the recent video network TSM-RGB [19] method (30.3%).

3.4. Experiments on THUMOS14 dataset

We performed controlled experiments to rectify the expected level of improvement of distills knowledge from a recognition network. We used videos from the THUMOS14 action detection challenge [10] to create a dataset for action anticipation. We first identified the temporal location of an action segment. Interested in the anticipation lead time of one second, we moved back one second and extracted a 1s clip ending at that location and used as the input to the anticipation network. Using this strategy, we can compile a training dataset of multiple 1s clips. We then trained two anticipation networks, one using the full training set and the other using the smaller training set with 50% of the data. We experimented with both 2D and 3D ConvNet architectures to see how the size of the training set affects the anticipation performance.

The anticipation performance of these networks is plotted in Fig. 3. As can be seen, more data improved the performance of an anticipation network. When doubling the amount of annotated training data, the gain in accuracy of the two networks (for two types of features) were 1.4% and 2.8%. This experiment showed that doubling the amount of annotated training data would only yield moderate improvement in anticipation accuracy, perhaps somewhere from 1% to 3%.

Other important factors are the accuracy of the recognition network and also the quality of the unlabeled data. Fig. 3 shows the performance of the anticipation networks trained with knowledge distillation. In this experiment, we only used half of the labeled training data, and the other half as unlabeled data for knowledge distillation. As can be seen, the level of improvement was not as good as having actual ground truth annotations, and this can probably be attributed to the imperfection of the recognition network.

4. SUMMARY

We have presented a framework for knowledge distillation. This framework uses the action recognition network to supervise the training of an action anticipation network. With a novel knowledge distillation technique to account for the positional drift of semantic concepts in video, the action recognition network acts as a teacher guiding the anticipation network to attend to the relevant information needed for predicting the future action. Using this framework, we are able to leverage unlabeled data to train the anticipation network in a self-supervised manner. The experimental results on the JH-MDB and EPIC-KITCHENS datasets show the benefits of our proposed method.

Acknowledgement. This material is based on research sponsored by the Air Force Research Laboratory (AFRL), DARPA, under agreement number FA8750-19-2-1003. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the AFRL, DARPA, or the U.S. Government.

References

[1] Minh Hoai and Fernando De la Torre, "Max-margin early event detectors," *IJCV*, vol. 107, no. 2, pp. 191–202, 2014. 1

[2] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba, "Anticipating visual representations from unlabeled video," in *Proc. CVPR*, 2016. 1

[3] Boyu Wang and Minh Hoai, "Predicting body movement and recognizing actions: an integrated framework for mutual benefits," in *Proc. Int. Conf. Autom. Face and Gesture Recog.*, 2018.

[4] Boyu Wang, Lihan Huang, and Minh Hoai, "Active vision for early recognition of human actions," in *Proc. CVPR*, 2020.

[5] Boyu Wang and Minh Hoai, "Back to the beginning: Starting point detection for early recognition of ongoing human actions," in *CVIU*, 2018, vol. 175, pp. 24–31. 1

[6] Yang Wang and Minh Hoai, "Pulling actions out of context: Explicit separation for effective combination," in *Proc. CVPR*, 2018. 1

[7] Yang Wang, Vinh Tran, Gedas Bertasius, Lorenzo Torresani, and Minh Hoai, "Attentive action and context factorization," in *Proceedings of British Machine Vision Conference*, 2020. 1

[8] Hueihan Jhuang, Juergen Gall, Silvia Zuffi, Cordelia Schmid, and Michael J Black, "Towards understanding action recognition," in *Proc. ICCV*, 2013. 3

[9] 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 *Proc. ECCV*, 2018. 3

[10] A. Gorban, H. Idrees, Y.-G. Jiang, A. Roshan Zamir, I. Laptev, M. Shah, and R. Sukthankar, "THUMOS challenge: Action recognition with a large number of classes," http://www.thumos.info, 2015. 3, 4

[11] Yuge Shi, Basura Fernando, and Richard Hartley, "Action anticipation with rbf kernelized feature mapping rnn," in *Proc. ECCV*, 2018. 3

[12] Khurram Soomro, Haroon Idrees, and Mubarak Shah, "Predicting the where and what of actors and actions through online action localization," in *Proc. CVPR*, 2016. 3

[13] Ashesh Jain, Avi Singh, Hema S Koppula, Shane Soh, and Ashutosh Saxena, "Recurrent neural networks for driver activity anticipation via sensory-fusion architecture," in *Proc. Intl. Conf. on Robotics and Automation*. IEEE, 2016. 3

[14] Shugao Ma, Leonid Sigal, and Stan Sclaroff, "Learning activity progression in lstms for activity detection and early detection," in *Proc. CVPR*, 2016. 3

[15] Mohammad Sadegh Aliakbarian, Fatemeh Sadat Saleh, Mathieu Salzmann, Basura Fernando, Lars Petersson, and Lars Andersson, "Encouraging lstms to anticipate actions very early," in *Proc. ICCV*, 2017. 3

[16] Cristian Rodriguez, Basura Fernando, and Hongdong Li, "Action anticipation by predicting future dynamic images," in *ECCV Workshop on Anticipating Human Behavior*, 2018. 3

[17] Antoine Miech, Ivan Laptev, Josef Sivic, Heng Wang, Lorenzo Torresani, and Du Tran, "Leveraging the present to anticipate the future in videos," in *CVPR Workshops*, 2019. 4

[18] 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 *Proc. ECCV*, 2016. 4

[19] Ji Lin, Chuang Gan, and Song Han, "Tsm: Temporal shift module for efficient video understanding," in *Proc. ICCV*, 2019. 4

[20] Joao Carreira and Andrew Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *Proc. CVPR*, 2017. 4