

Rolling-Unrolling LSTMs for Action Anticipation from First-Person Video

Antonino Furnari and Giovanni Maria Farinella

Abstract—In this paper, we tackle the problem of egocentric action anticipation, i.e., predicting what actions the camera wearer will perform in the near future and which objects they will interact with. Specifically, we contribute Rolling-Unrolling LSTM, a learning architecture to anticipate actions from egocentric videos. The method is based on three components: 1) an architecture comprised of two LSTMs to model the sub-tasks of summarizing the past and inferring the future, 2) a Sequence Completion Pre-Training technique which encourages the LSTMs to focus on the different sub-tasks, and 3) a Modality ATTention (MATT) mechanism to efficiently fuse multi-modal predictions performed by processing RGB frames, optical flow fields and object-based features. The proposed approach is validated on EPIC-Kitchens, EGTEA Gaze+ and ActivityNet. The experiments show that the proposed architecture is state-of-the-art in the domain of egocentric videos, achieving top performances in the 2019 EPIC-Kitchens egocentric action anticipation challenge. The approach also achieves competitive performance on ActivityNet with respect to methods not based on unsupervised pre-training and generalizes to the tasks of early action recognition and action recognition. To encourage research on this challenging topic, we made our code, trained models, and pre-extracted features available at our web page: <http://iplab.dmi.unict.it/rulstm>.

Index Terms—Action Anticipation, Egocentric Vision, Recurrent Neural Networks, LSTM

1 INTRODUCTION

THE ability to anticipate what is going to happen in the near future is fundamental for human beings in order to interact with the environment and make decisions. Anticipation abilities are also fundamental to deploy intelligent systems which need to interact with a complex environment or other humans to automate challenging tasks and provide assistance. Examples of such applications include autonomous vehicles [1], human-robotic symbiotic systems [2], and wearable assistants [3], [4]. However, designing computational approaches to address tasks such as early action recognition [5], [6], [7] and action anticipation [2], [8], [9] is challenging as it often requires to model the relationship between past and future events, in the presence of incomplete observations. First-Person (Egocentric) Vision [4] offers an interesting scenario to study tasks related to anticipation. On one hand, wearable cameras provide a means to collect naturally long videos containing multiple subsequent interactions with objects, which makes anticipation tasks unconstrained. On the other hand, the ability to predict in advance what actions the camera wearer is going to perform and what objects they are going to interact with is useful to build intelligent wearable systems capable of anticipating the user’s goals to provide assistance [4].

In this paper, we tackle the problem of egocentric action anticipation. The task consists in recognizing a future action from an observation of the past. Fig. 1 illustrates the problem as defined in [10]: given an action starting at time τ_s , the system should predict the related action class by observing a video segment of temporal bounds $[\tau_s - (\tau_o + \tau_a), \tau_s - \tau_a]$, where τ_o denotes the “observation time”, i.e. the length of

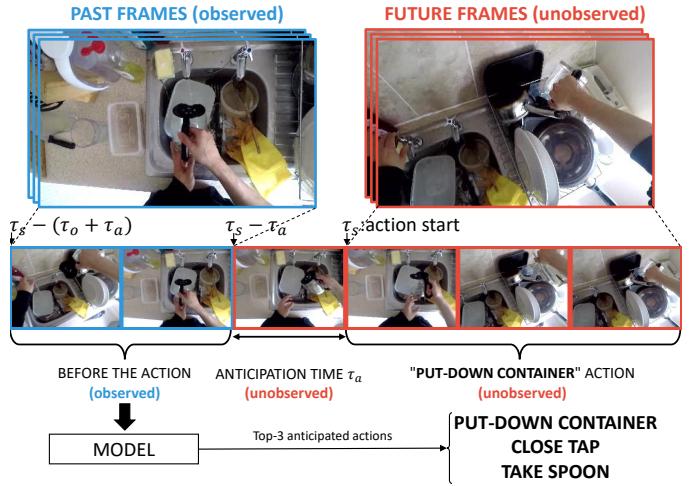


Fig. 1. Egocentric Action Anticipation. See text for notation.

the observed video, and τ_a denotes the “anticipation time”, i.e., how much in advance the action has to be anticipated. Since the future is naturally uncertain, action anticipation models usually predict more than one possible action and the evaluation is performed using Top-k measures [2], [11]. While the task of action anticipation has been investigated in the domain of third person vision [2], [8], [9], [12], [13], [14], less attention has been devoted to the challenging scenario of egocentric videos [10], [11], [15].

Our work stems from the observation that egocentric action anticipation methods need to address two sub-tasks, which we refer to as “encoding” and “inference”. In the encoding stage, the model has to summarize what has been observed in the past (e.g., “a container has been washed” in the observed segment in Fig. 1). In the infer-

Antonino Furnari and Giovanni Maria Farinella are co-first authors.

- A. Furnari and G. M. Farinella are with the Department of Mathematics and Computer Science, University of Catania, Italy
E-mail: {furnari,gfarinella}@dmi.unict.it

ence stage, the algorithm makes hypotheses about what may happen in the future, given the summary of the past and the current observation (e.g., “put-down container”, “close tap”, “take spoon” in Fig. 1). Previous approaches have generally addressed the two sub-tasks jointly [5], [7], [9], [10]. Our method is designed to disentangle them by using two separate LSTMs. “A Rolling” LSTM (R-LSTM) continuously encodes streaming observations and keeps an updated summary of what has been observed so far. When an anticipation is required, the “Unrolling” LSTM (U-LSTM) is initialized with the current hidden and cell states of the R-LSTM (which encode the summary of the past) and makes predictions about the future. While previous approaches considered fixed anticipation times [9], [10], [11], our architecture is designed to anticipate an action at multiple anticipation times. For instance, our model can anticipate actions from 2s, to 0.25s before they occur, with the prediction refined as we get closer to the beginning of the action. To encourage the disentanglement of encoding and inference, we propose to pre-train our model with a novel “Sequence Completion Pre-training” (SCP) technique. Our method processes video in a multi-modal fashion, analyzing spatial observations (RGB frames), motion (optical flow) and object-based features obtained through an object detector. We find that classic multimodal fusion techniques such as late and early fusion are limited in the context of action anticipation. Therefore, we propose a novel “Modality ATTention” (MATT) mechanism to adaptively estimate optimal fusion weights for each modality by considering the outputs of the modality-specific R-LSTM components. We perform experiments on two large-scale datasets of egocentric videos, EPIC-KITCHENS [10] and EGTEA Gaze+ [16], and a standard benchmark of third person videos, Activitynet [17]. The experiments show that the proposed method outperforms several state-of-the-art approaches and baselines in the task of egocentric action anticipation and generalizes to the scenario of third person vision, as well as to the tasks of early action recognition and action recognition. The proposed approach also achieved top performances in the 2019 EPIC-Kitchens egocentric action anticipation challenge¹.

The contributions of our work are the following: 1) we systematically investigate the problem of egocentric action anticipation within the framework provided by the EPIC-Kitchens dataset and its related challenges; 2) we benchmark popular ideas and approaches to action anticipation and define the proposed “Rolling-Unrolling LSTM” (RU) architecture, which is able to anticipate egocentric actions at multiple temporal scales; 3) we introduce two novel techniques specific to the investigated problem, i.e., i) “Sequence Completion Pre-training” and ii) adaptive fusion of multi-modal predictions with Modality ATTention (MATT); 4) we performed extensive evaluations to highlight the limits of previous methods and report improvements of the proposed approach over the state-of-the-art. To support future research in this field, we publicly release the code of our approach.

The reminder of this paper is organized as follows.

¹. See <https://epic-kitchens.github.io/Reports/EPIC-Kitchens-Challenges-2019-Report.pdf> for more details.

Section 2 revises the related works. Section 3 details the proposed approach. Section 4 reports the experimental settings, whereas Section 5 discusses the results. Finally, Section 7 concludes the paper.

2 RELATED WORK

Our work is related to past research on action recognition, early action recognition, and anticipation tasks in both third and first-person vision.

2.1 Action Recognition

Classic approaches to action recognition from video have generally relied on the extraction and processing of hand-designed features. Among the most notable approaches, Laptev [18] proposed space-time interest points to classify events. Laptev et al. [19] further investigated the use of space-time features, space-time pyramids and SVMs for human action classification. Later, Wang et al. [20], [21] introduced dense trajectories to encode local motion and object appearance. More recent approaches investigated the use of deep learning to learn representations suitable to recognize actions directly from video. Karpathy et al. [22] considered the use of Convolutional Neural Networks (CNNs) and investigated different strategies to fuse per-frame predictions. Simonyan et al. [23] proposed Two-Stream CNN (2SCNN), a multi-branch architecture which recognizes actions by processing both appearance (RGB) and motion (optical flow) data. Feichtenhofer et al. [24], [25], [26] studied approaches to fuse predictions performed by the motion and appearance streams of a 2SCNN to improve recognition. Wang et al. [27] designed Temporal Segment Network (TSN), a general framework to train two-stream CNNs for action recognition. Zhou et al. [28] introduced Temporal Relation Network (TRN), a module capable of encoding temporal dependencies between video frames at multiple time scales. Lin et al. [29] proposed the Temporal Shift Module (TSM), a component which facilitates information exchange among neighboring frames without the introduction of extra parameters in the network. Other authors investigated the use of 3D CNNs as a natural extension of 2D convolutional networks for video processing. Tran et al. [30] demonstrated the use of 3D CNNs to learn spatio-temporal features for video classification. Carreira and Zisserman [31] proposed Inflated 3D (I3D) CNNs and showed how the weights of this architecture can be bootstrapped from a 2D CNN pre-trained on Imagenet. Hara et al. [32] studied whether 3D CNNs based on standard 2D ResNet [33] architectures could be exploited for action recognition from video. Tran et al. [34] proposed R(2+1)D CNNs which factorize 3D convolutions as sequences of spatial and temporal convolutions.

Egocentric action recognition has also been studied in past works. Spriggs et al. [35] investigated the problem of supervised and unsupervised action segmentation using Inertial Measurement Units (IMU) and egocentric video. Fathi et al. [36] proposed to recognize actions by modeling activities, hands and objects. Fathi et al. [37] employed eye gaze measurements to recognize egocentric actions. Pirsavash and Ramanan [38] proposed to recognize egocentric activities using object-centric representations. Li et al. [39] studied

how different egocentric cues, (including gaze, the presence of hands and objects, as well as head motion), can be used to perform the task. Ryoo et al. [40] proposed an approach to temporally pool features for egocentric action recognition. Ma et al. [41] designed a deep learning architecture which allows to integrate different egocentric-based features to recognize actions. Singh et al. [42] adapted improved dense trajectories to the problem of egocentric action recognition. Singh et al. [43] proposed a multi-stream CNN to recognize egocentric actions using spatial features, temporal features and egocentric cues. Li et al. [16] introduced a deep learning model for joint gaze estimation and action recognition in egocentric video. Sudhakaran et al. [44], [45] proposed to use a convolutional LSTM to recognize actions from egocentric video with an attention mechanism which learns to focus on image regions containing objects.

Our work builds on previous ideas investigated in the context of action recognition such as the use of multiple modalities for video analysis [23], the use of Temporal Segment Networks [27] as a principled way to train CNNs for action recognition, as well as the explicit encoding of object-based features [36], [38], [41], [43], [45] to analyze egocentric video. However, in contrast with the aforementioned works, we address the problem of egocentric action *anticipation* and show that approaches designed for action recognition, such as TSN [27] and late fusion to merge spatial and temporal predictions [23] are not directly applicable to the problem of egocentric action anticipation.

2.2 Early Action Recognition in Third Person Vision

Early action recognition refers to the task of recognizing an ongoing action as early as possible from partial observations [6]. The problem of early action recognition has been widely investigated in the domain of third person vision. Ryoo [46] introduced the problem of recognizing ongoing actions from streaming video and addressed it proposing an integral histogram of spatio-temporal features. Cao et al. [47] used sparse coding to recognize actions from partially observed videos. Haoi and De la Torre [48] proposed to use Structured Output SVM to detect partially observed events. Huang et al. [49] introduced Sequential Max-Margin Event Detectors, a method which performs early action detection by sequentially discarding classes until one class is identified as the detected one. De Geest et al. [6] released a new dataset for online action detection and benchmarked several baseline methods to address the task. Ma et al. [7] used LSTMs to address the problem of early action detection from video. Aliakbarian et al. [5] proposed a two stage LSTM architecture which models context and action to perform early action recognition. Beccattini et al. [50] designed ProgressNet, an approach capable of estimating the progress of actions and localizing them in space and time. De Geest and Tuytelaars [51] addressed early action recognition proposing a “feedback network” which uses two LSTM streams to interpret feature representations and model the temporal structure of subsequent observations.

Differently from these works, we address the task of anticipating actions from egocentric video, i.e., predicting an action before it starts, and hence before it can be even partially observed. However, given the similarity between

early action recognition and action anticipation, we consider and evaluate some ideas investigated in the context of early action recognition, such as the use of LSTMs to process streaming observations [5], [7], [51] and the use of dedicated loss functions [7]. Moreover, we show that the proposed architecture also generalizes to the problem of early egocentric action recognition, achieving state-of-the-art performance.

2.3 Action Anticipation in Third Person Vision

Action anticipation refers to the task of predicting an action *before* it actually begins [8]. Previous works investigated different forms of action and activity anticipation from third person video. Kitani et al. [52] considered the task of inferring future paths followed by people observed from a static camera. Huang and Kitani [13] explored the task of action anticipation in the context of dual-agent interactions, where the actions of an agent are used to predict the response of the other agent. Lan et al. [53] proposed a hierarchical representation to anticipate future human actions from a still image or a short video clip. Jain et al. [54] designed an Autoregressive Input-Output HMM to anticipate driving maneuvers a few seconds before they occur using video, vehicle dynamics, GPS, and street maps. Jain et al. [14] proposed a learning architecture based on LSTMs for driver activity anticipation. Koppula and Saxena [2] used object affordances to anticipate the possible future actions performed by a user from a robotic point of view. Vondrick et al. [9] addressed action anticipation by training a CNN to regress the representations of future frames from past ones in an unsupervised way. Gao et al. [8] proposed an Encoder-Decoder LSTM architecture which predicts future actions by encoding the representations of past frames and regressing the representations of future frames. Similarly to [9], the model can be pre-trained from unlabeled videos in an unsupervised way. Felsen et al. [55] developed a framework to forecast future events in team sports video from visual input. Mahmud et al. [56] designed a system able to infer the labels and starting frames of future actions. Zeng et al. [57] introduced a general framework which uses inverse reinforcement learning to perform visual forecasting at different levels of abstraction, including story-line forecasting, action anticipation and future frames generation. Abu et al. [12] explored the use of CNNs and RNNs to predict the occurrence of future actions based on past observations.

In this work, we consider the problem of action anticipation from egocentric visual data. Nevertheless, our work builds on some of the ideas explored in past works such as the use of LSTMs [8], [12], [14] to anticipate actions, the use of the encoder-decoder framework to encode past observations and produce hypotheses of future actions [8], and the use of object specific features [56] to determine which objects are present in the scene, we show that other approaches, such as the direct regression of future representations [8], [9], do not achieve satisfactory performance in the egocentric scenario.

2.4 Anticipation in First-Person Vision

Past works have investigated different problems related to anticipation from first-person vision. Zhou and Berg [58] studied methods to infer the ordering of egocentric video

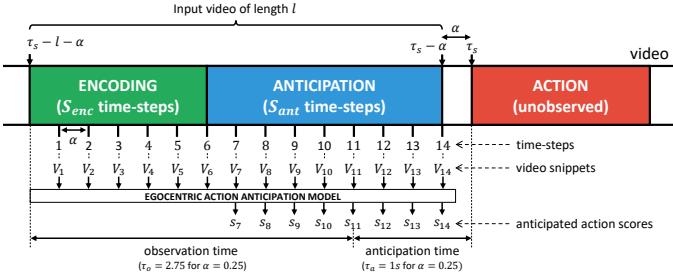


Fig. 2. Video processing scheme adopted by the proposed method. In the example above, we set $S_{enc} = 6$ and $S_{ant} = 8$.

segments. Ryoo et al. [15] proposed to analyze onset actions to anticipate potentially dangerous actions performed by humans against a robot. Soran et al. [3] developed a system capable of inferring the next action performed in a known workflow to notify the user if a missing action is detected. Park et al. [59] proposed a method to predict the future trajectories of the camera wearer from egocentric video. Zhang et al. [60] developed a method to predict eye gaze fixations in future video frames. Furnari et al. [61] proposed to anticipate human-object interactions by analyzing the motion of objects in egocentric video. Chenyou et al. [62] designed a method capable of forecasting the position of hands and objects in future frames. Rhinehart and Kitani [63] used inverse reinforcement learning to anticipate future locations, objects and activities from egocentric video.

Previous works on anticipation from egocentric video have investigated different tasks and evaluated methods on different datasets and under different evaluation frameworks. Differently from these works, we consider the egocentric action anticipation challenge recently proposed by Damen et al. [10]. It should be noted that few works [11] have tackled the problem so far. While directly comparing our approach with respect to most of the aforementioned approach is unfeasible due to the lack of a common framework, our method incorporates some ideas from past works, such as the analysis of past actions [15] and the detection of the objects present in the scene to infer future actions [61].

3 PROPOSED APPROACH

In this Section, we discuss the proposed approach. Specifically, Section 3.1 introduces the strategy used to process video, Section 3.2 presents the proposed Rolling-Unrolling LSTMs module, Section 3.3 discusses the Sequence Completion Pre-Training (SCP) approach used to encourage the rolling and unrolling LSTMs to focus on different sub-tasks, Section 3.4 introduces the modality attention mechanism used to fuse multi-modal predictions, Section 3.5 details the definition of the representation functions used in the different branches of our architecture.

3.1 Video Processing Scheme

Past approaches [9], [10], [11] performed action anticipation considering a fixed anticipation time τ_a , usually set to 1 second. This has been usually achieved by training a classifier to predict the action happening τ_a seconds after the end of an observed video segment. Similarly to [8], we

propose to anticipate actions at different temporal scales by using recurrent models. The authors of [8] obtain this multi-scale anticipation by training the model with variable anticipation times and performing inference using a fixed anticipation time chosen from the ones used during training. Also, the approach proposed in [8] requires the model to consume all the observed video before anticipating actions, which results in the separation between the observation and anticipation stages. We argue that it would be natural to allow the model to make predictions *while* observing the video and possibly refine them as more frames are processed. We hence propose the video processing strategy illustrated in Fig. 2. According to the proposed scheme, the video is processed sequentially, with a video snippet V_t consumed every α seconds, where t indexes the current time-step. At each time-step t , the model processes an input video snippet V_t and optionally outputs a set of scores s_t for the anticipated actions. Since the video is processed sequentially, the prediction made at time-step t depends only on observations processed at previous time-steps. Specifically, the video is processed in two stages: an “encoding” stage, carried out for S_{enc} time-steps and an “anticipation” stage, carried out for S_{ant} time-steps. During the “encoding” stage, the model only observes incoming video snippets V_i and does not anticipate actions. During the “anticipation” stage, the model both observes the input video snippets V_i and outputs action scores s_i for the anticipated actions. This scheme effectively allows to anticipate actions at different anticipation times. In particular, in our experiments we set $\alpha = 0.25s$, $S_{enc} = 6$ and $S_{ant} = 8$. In these settings, the model will process videos of length $l = \alpha(S_{enc} + S_{ant}) = 3.5s$ and output 8 predictions at the following anticipation times: $\tau_a \in \{2s, 1.75s, 1.5s, 1.25s, 1s, 0.75s, 0.5s, 0.25s\}$. At time step t , the effective observation time will be given by $\alpha \cdot t$. Therefore, the 8 predictions will be performed at the following observation times: $\tau_o \in \{1.75s, 2s, 2.25s, 2.5s, 2.75s, 3s, 3.25s, 3.5s\}$. It should be noted that our formulation generalizes the one proposed in [10]. For instance, at time-step $t = 11$, our model will anticipate actions with an effective observation time equal to $\tau_o = \alpha \cdot t = 2.75s$ and an anticipation time equal to $\tau_a = \alpha(S_{ant} + S_{enc} + 1 - t) = 1s$.

3.2 Proposed Rolling-Unrolling LSTMs

Our model is inspired by encoder-decoder sequence to sequence models for text processing [64]. Such models include an encoder which processes the words of the input sentence and a decoder which generates the words of the output sentence. Both the encoder and decoder are often implemented using LSTMs. Rather than analyzing words, our model processes high level representations of frames obtained through a representation function φ . The decoder is initialized with the internal representation of the encoder and iterates over the last representation to anticipate future actions. To allow for continuous anticipation of actions, the decoder is attached to each encoding time-step. This allows to anticipate actions and refine predictions in a continuous fashion. We term the encoder “Rolling LSTM” (R-LSTM) and the decoder “Unrolling LSTM” (U-LSTM). Figure 3 shows a diagram of the overall architecture of the proposed

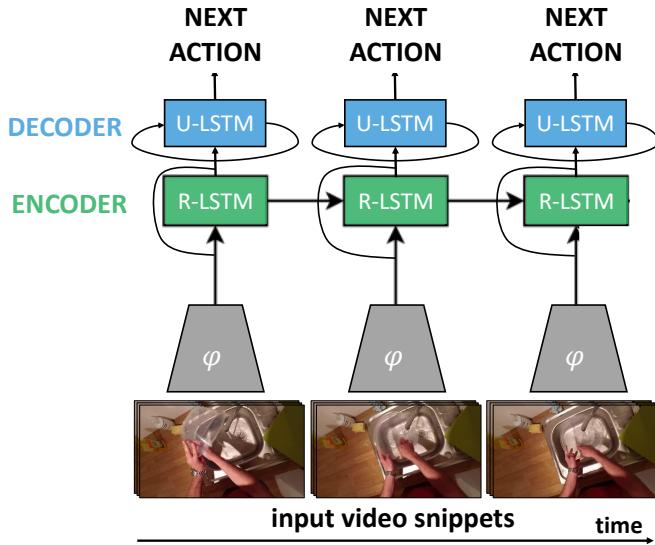


Fig. 3. Overall architecture of the proposed Rolling-Unrolling LSTM architecture based on encoder-decoder models.

Rolling-Unrolling (RU) LSTMs, which is described in details below.

Following previous literature [23], we include multiple branches which analyze the video pre-processed according to different modalities. Specifically, at each time-step t , the input video snippet V_t is represented using different modality-specific representation functions $\varphi_1, \dots, \varphi_M$, where M is the number of considered modalities. The representation functions can be learned and depend on the parameters $\theta^{\varphi_1}, \dots, \theta^{\varphi_M}$. This process allows to obtain modality-specific representations for the input video snippets $f_{1,t} = \varphi_1(V_t), \dots, f_{M,t} = \varphi_M(V_t)$, where $f_{m,t}$ is the feature vector computed at time-step t for modality m . The feature vector $f_{m,t}$ is hence fed to the m^{th} branch of the architecture. In this work, we consider $M = 3$ modalities, i.e., RGB frames (spatial branch), optical flow (motion branch) and object-based features (object branch).

Fig. 4 illustrates in details the processing happening in a single branch m . For illustration purposes, the figure shows an example for $S_{enc} = 1$ and $S_{ant} = 3$. At a given time step t , the feature vector $f_{m,t}$ is fed to the R-LSTM, which is responsible for recursively encoding the semantic content of the incoming representations. This is performed according to the following equation:

$$(h_{m,t}^R, c_{m,t}^R) = LSTM_{\theta_m^R}(f_{m,t}, h_{m,t-1}^R, c_{m,t-1}^R). \quad (1)$$

In the equation above, $LSTM_{\theta_m^R}$ denotes the R-LSTM related to branch m , which depends on the learnable parameters θ_m^R , whereas $h_{m,t}^R$ and $c_{m,t}^R$ are the hidden and cell states computed at time step t in the branch related to modality m . The initial hidden and cell states of the R-LSTM are initialized with zeros:

$$h_{m,0}^R = \mathbf{0}, c_{m,0}^R = \mathbf{0}. \quad (2)$$

Note that the $LSTM$ function follows the standard implementation of LSTMs [65], [66].

During the anticipation stage, at each time step t , the U-LSTM is used to predict future actions. The U-LSTM is

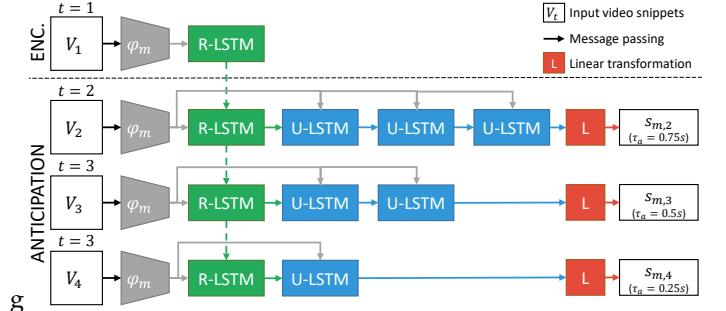


Fig. 4. Example of modality-specific branch with $S_{enc} = 1$ and $S_{ant} = 3$.

initialized with the hidden and cell states of the R-LSTM at the current time-step:

$$h_{m,0}^U = h_{m,t}^R, c_{m,0}^U = c_{m,t}^R \quad (3)$$

and iterates over the representation of the current video snippet $f_{m,t}$ for a number of times n_t equal to the number of time-steps needed to reach the beginning of the action: $n_t = S_{ant} + S_{enc} + 1 - t$. Note that this number is proportional to the current anticipation time, which can be computed as $\alpha \cdot n_t$. Similarly to the R-LSTM, the hidden and cell states of the U-LSTM are computed as follows at the generic iteration j :

$$(h_{m,j}^U, c_{m,j}^U) = LSTM_{\theta_m^U}(f_{m,t}, h_{m,j-1}^U, c_{m,j-1}^U). \quad (4)$$

In Equation (4), $LSTM_{\theta_m^U}$ represents the U-LSTM network related to branch m , which depends on the learnable parameters θ_m^U . The vectors $h_{m,t}^U$ and $c_{m,t}^U$ are the hidden and cell states computed at iteration j for modality m . It is worth noting that the input $f_{m,t}$ of the U-LSTM does not depend on j (see Eq. (4)), because it is fixed during the “unrolling” procedure. The main rationale of “unrolling” the U-LSTM for a different number of times at each time-step is to encourage the architecture to produce different predictions at different anticipation times.

Modality-specific anticipation scores $s_{m,t}$ for the anticipated actions are finally computed at time-step t by feeding the last hidden vector h_{m,n_t}^U of the U-LSTM to a linear transformation with learnable parameters θ_m^W and θ_m^b :

$$s_{m,t} = \theta_m^W h_{m,n_t}^U + \theta_m^b. \quad (5)$$

Anticipated action probabilities for modality m at time-step t are computed normalizing the scores $s_{m,t}$ with the Softmax function:

$$p_{m,t,i} = \frac{\exp(s_{m,t,i})}{\sum_k \exp(s_{m,t,k})} \quad (6)$$

where $s_{m,t,i}$ denotes the i^{th} component of the score vector $s_{m,t}$. A modality-specific RU branch is hence trained with the cross-entropy loss:

$$\mathcal{L}(p_m, y) = -\frac{1}{S_{ant}} \sum_t \log p_{m,t,y} \quad (7)$$

where p_m is the set of probability distributions over actions computed by branch m in all time-steps, y is the ground truth class of the current sample and \mathcal{L} is minimized with respect to the parameters $\theta_m^R, \theta_m^U, \theta_m^W$ and θ_m^b .

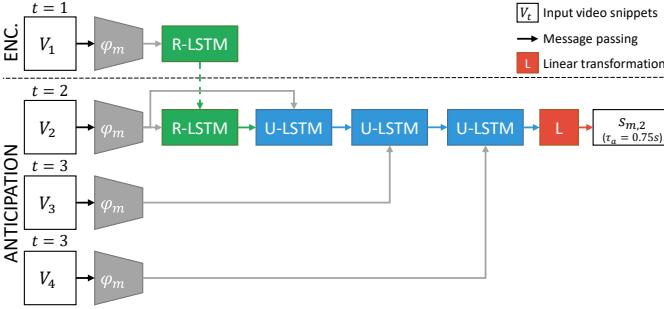


Fig. 5. Example of connection scheme used during SCP for time-step $t = 2$.

3.3 Sequence Completion Pre-Training (SCP)

The two LSTMs included in the proposed Rolling-Unrolling architecture are introduced to address two specific sub-tasks: the R-LSTM should encode past observations and summarize what has happened up to a given time-step, whereas the U-LSTM should focus on anticipating future actions conditioned on the hidden and cell vectors of the R-LSTM. However, in practice, this might not happen. For instance, the R-LSTM could try to both summarize the past and anticipate future actions, which would make the task of the R-LSTM harder. To encourage the two LSTMs to focus on the two different sub-tasks, we introduce a novel Sequence Completion Pre-training (SCP) procedure. During SCP, the connections of the network are modified to allow the U-LSTM to process future representations rather than iterating on the most recent one. In practice, the U-LSTM hidden and cell states are computed as follows during SCP:

$$(h_{m,j}^U, c_{m,j}^U) = \text{LSTM}_{\theta_m^U}(f_{m,t+j-1}, h_{m,j-1}^U, c_{m,j-1}^U) \quad (8)$$

where the input representations $f_{m,t+j-1}$ are sampled from future time-steps $t + j - 1$. Fig. 5 illustrates an example of the connection scheme used during SCP for time-step $t = 2$. Note that this is different from Eq. (4), in which only the most recent representation is processed. After SCP, the network is fine-tuned to the action anticipation task following Eq. (4). The main goal of pre-training the model with SCP is to allow the R-LSTM to focus on summarizing past representations without trying to anticipate the future. Indeed, since the U-LSTM can “cheat” by looking into the future, the R-LSTM does not need to try to anticipate future actions to minimize the loss and is hence encouraged to focus on encoding.

3.4 Modality ATTention (MATT)

Equation (5) allows to obtain modality-specific action scores $s_{m,t}$ from the hidden representations of the U-LSTMs contained in each branch. One way to fuse these scores is to compute a linear combination with a set of fixed weights w_1, \dots, w_M , which is generally referred to as late fusion:

$$s_t = w_1 \cdot s_{1,t} + \dots + w_M \cdot s_{M,t}. \quad (9)$$

The fusion weights w_m are fixed and generally found using cross validation. We observe that, in the case of action anticipation, the relative importance of each modality might depend on the observed video. For instance, in some cases

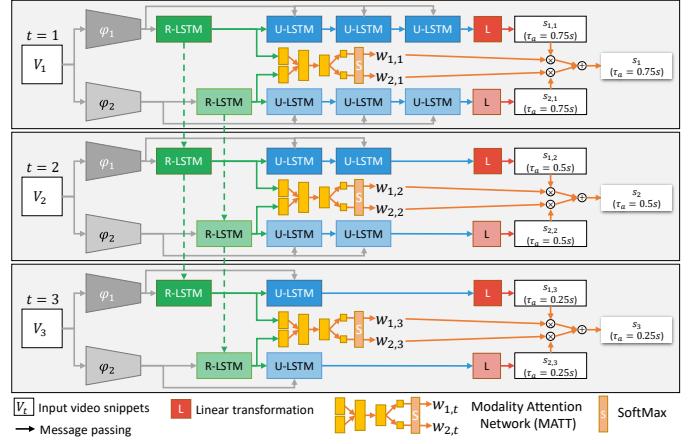


Fig. 6. Example of the complete architecture with two branches and the Modality ATTention mechanism (MATT).

the object detector computing object-based features might fail and hence become unreliable, or in other cases there could be little motion in the scenes, which would make the optical flow modality less useful. Inspired by previous work on attention [67], [68] and multi-modal fusion [69], we introduce a Modality ATTention (MATT) module which computes a set of attention scores indicating the relative importance of each modality for the final prediction. At a given time-step t , the attention scores are computed by feeding the concatenation of the hidden and cell vectors of the modality specific R-LSTM networks to a feed-forward neural network D which depends on the learnable parameters θ^{MATT} . This computation is defined as follows:

$$\lambda_t = D_{\theta^{MATT}}(\oplus_{m=1}^M (h_{m,t}^R \oplus c_{m,t}^R)) \quad (10)$$

where \oplus denotes the concatenation operator and $\oplus_{m=1}^M (h_{m,t}^R \oplus c_{m,t}^R)$ is the concatenation of the hidden and cell vectors produced by the R-LSTM at time-step t across all modalities. Late fusion weights can be obtained by normalizing the score vector λ_t using the softmax function, which makes sure that the computed fusion weights sum to one:

$$w_{m,t} = \frac{e^{\lambda_{t,m}}}{\sum_k e^{\lambda_{t,k}}} \quad (11)$$

where $\lambda_{t,m}$ is the m^{th} component of the score vector λ_t . The final anticipated action scores are obtained at time-step t by fusing the modality-specific predictions produced by the different branches with a linear combination as follows:

$$s_t = \sum_m w_{m,t} \cdot s_{m,t}. \quad (12)$$

Fig. 6 illustrates an example of a complete RU with two modalities and the MATT fusion mechanism. For illustration purposes, the figure shows only three anticipation steps. Note that, since the result of the linear combination defined in Eq. (12) is differentiable with respect to both the weights and the scores, the whole architecture is trainable end-to-end using the loss reported in Equation (7).

3.5 Branches and Representation Functions

We instantiate the proposed architecture with 3 branches: a spatial branch which processes RGB frames, a motion

branch which processes optical flow fields, and an object branch which processes object-based features. The input to the representation functions are video snippets of 6 frames $V_t = \{I_{t,1}, I_{t,2}, \dots, I_{t,6}\}$, where $I_{t,i}$ is the i^{th} frame of the video snippet V_t . The representation function φ_1 of the spatial branch is implemented as a Batch Normalized Inception CNN [70] CNN_{RGB} which is trained for action anticipation on the target dataset within the TSN framework [27]. The CNN takes as input the last frame of each video snippet and extracts features from the last layer of 1024 units preceding the final classifier. Hence:

$$\varphi_1(V_t) = CNN_{RGB}(I_{t,6}) \quad (13)$$

Similarly, the representation function φ_2 of the motion branch is implemented as a Batch Normalized Inception CNN [70] CNN_{Flow} pre-trained for action recognition on the target dataset following TSN [27]. The network analyzes a stack of optical flow fields computed from the 6 frames (5 frame pairs) of the current video snippet as proposed in [27]. Similarly to CNN_{RGB} , CNN_{Flow} extracts 1024-dimensional features from the last layer preceding the final classifier. This representation function is hence defined as:

$$\varphi_2(V_t) = CNN_{Flow}(\text{flow}(I_{t,1}, \dots, I_{t,6})) \quad (14)$$

where flow computes the optical flow fields of the input frames and returns a tensor of 10 channels obtained by stacking x and y optical flow images computed from frame pairs. It is worth noting that, since the CNNs have been trained for the action recognition task, φ_1 and φ_2 allow to obtain “action-centric” representations of the input frames, which can be used by the R-LSTM to summarize what has happened in the past.

The representation function φ_3 related to the object branch includes an object detector OD which detects objects in the last frame $I_{t,6}$ of the input video snippet V_t . When object-level annotations are available, the detector OD is trained on the target dataset. A fixed-length “bag of objects” representation is hence obtained by accumulating the confidence scores of all bounding boxes predicted for each object category. Let $\{(b_{t,i}, b_{t,i}^c, b_{t,i}^s)\} = OD(I_{t,6})$ be the set of bounding boxes $b_{t,i}$ predicted by the object detector along with the corresponding classes $b_{t,i}^c$ and confidence scores $b_{t,i}^s$. The j^{th} component of the fixed-length representation vector is obtained by summing the confidence scores of all objects detected for class j , i.e.,

$$\text{boo}(OD(I_{t,6}))_j = \sum_i [b_{t,i}^c = j] b_{t,i}^s \quad (15)$$

where boo is the “bag of objects” function and $[.]$ denotes the Iverson bracket. The representation function can hence be defined as follows:

$$\varphi_3(V_t) = \text{boo}(OD(I_{t,6})) \quad (16)$$

This representation encodes only the presence of an object in the scene, discarding its position in the frame, similarly to the representation proposed in [38] for egocentric activity recognition. We found this representation to be sufficient in the case of egocentric action anticipation. Differently

from φ_1 and φ_2 , φ_3 produces object-centric features which indicate what objects are likely to be present in the scene.²

3.6 Early Action Recognition and Action Recognition

We note that the proposed model can also be used for early action anticipation and action recognition. Specifically, this can be done by sampling a given number of frames N from the video segment containing the action to be recognized and feeding the frames to the RU-LSTM model. To perform early action recognition, i.e., classifying the video before the action is completed, we set $S_{enc} = 0$ and $S_{ant} = N$. The output of the model at the last time-step $t = N$ can be used to perform action recognition.

4 EXPERIMENTAL SETTINGS

This section discusses the experimental settings, including the datasets considered for the evaluation, the evaluation measures and the compared methods.

4.1 Datasets

We performed experiments on two large-scale datasets of egocentric videos and a large-scale dataset of third person videos: EPIC-Kitchens [10], EGTEA Gaze+ [16] and ActivityNet [17]. EPIC-Kitchens contains 39,596 action annotations, 125 verbs, and 352 nouns. We split the public training set of EPIC-Kitchens (28,472 action segments) into training (23,493 segments) and validation (4,979 segments) sets by randomly choosing 232 videos for training and 40 videos for validation. We considered all unique $(verb, noun)$ pairs in the public training set, thus obtaining 2,513 unique actions. EGTEA Gaze+ contains 10,325 action annotations, 19 verbs, 51 nouns and 106 unique actions. Methods are evaluated on EGTEA Gaze+ reporting the average performance across the three splits provided by the authors of the dataset [16]. We considered the 1.3 release of ActivityNet, which contains 10024 training videos, 4926 validation videos, and 5044 test videos. Each video is labeled with one or more action segments belonging to one of 200 action classes. Videos are provided as YouTube links. Hence, depending on the country and time of download, some videos are not available. We have been able to download 7911 training videos and 3832 validation videos. Test videos are not used in our experiments as their labels are not publicly available. The total number of training annotations amounts to 11890, while the total amount of validation annotations is equal to 5786. While the main focus of this work is on egocentric vision, we report experiments on this challenging dataset of third person videos to investigate the differences between the two scenarios and to what extent our approach generalizes to this domain.

We have extracted frames from the EPIC-Kitchens and EGTEA Gaze+ datasets using a constant frame-rate of 30fps , whereas ActivityNet videos have been sub-sampled to 12fps . All frames have been resized to 456×256 pixels. Optical flow fields have been computed on all datasets using the TVL1 algorithm [71].

² The reader is referred to Appendix A for the implementation and training details of the proposed approach.

TABLE 1
Egocentric action anticipation results on the EPIC-KITCHENS dataset.

	Top-5 ACTION Accuracy% @ different τ_a (s)								Top-5 Acc.% @1s			M. Top-5 Rec.% @1s			Mean $TtA(5)$		
	2	1.75	1.5	1.25	1.0	0.75	0.5	0.25	VERB	NOUN	ACT.	VERB	NOUN	ACT.	VERB	NOUN	ACT.
DMR [9]	/	/	/	/	/	16.86	/	/	73.66	29.99	16.86	24.50	20.89	03.23	/	/	/
ATSN [10]	/	/	/	/	/	16.29	/	/	77.30	39.93	16.29	33.08	32.77	07.60	/	/	/
MCE [11]	/	/	/	/	/	26.11	/	/	73.35	38.86	26.11	34.62	32.59	06.50	/	/	/
TCN [72]	19.33	19.95	20.43	20.82	21.82	23.03	23.35	24.40	73.93	36.75	21.82	28.95	30.28	05.28	01.54	00.88	00.56
ED* [8]	21.45	22.37	23.26	24.51	25.20	26.34	27.45	28.67	76.24	42.18	25.20	42.25	42.00	09.98	01.59	00.99	00.61
ED [8]	21.53	22.22	23.20	24.78	25.75	26.69	27.66	29.74	75.46	42.96	25.75	41.77	42.59	10.97	01.60	01.02	00.63
FN [51]	23.47	24.07	24.68	25.66	26.27	26.87	27.88	28.96	74.84	40.87	26.27	35.30	37.77	06.64	01.52	00.86	00.56
RL [7]	25.95	26.49	27.15	28.48	29.61	30.81	31.86	32.84	76.79	44.53	29.61	40.80	40.87	10.64	01.57	00.94	00.62
EL [14]	24.68	25.68	26.41	27.35	28.56	30.27	31.50	33.55	75.66	43.72	28.56	38.70	40.32	08.62	01.55	00.94	00.62
LSTM [66]	26.45	27.11	28.22	29.24	29.89	31.03	31.88	33.19	76.33	44.21	29.89	39.31	40.30	10.42	01.56	00.93	00.63
RU-LSTM	29.44	30.73	32.24	33.41	35.32	36.34	37.37	38.98	79.55	51.79	35.32	43.72	49.90	15.10	01.62	01.11	00.76
Improvement	+2.99	+3.62	+4.02	+4.17	+5.43	+5.31	+5.49	+5.43	+2.25	+7.26	+5.43	+1.47	+7.31	+4.13	+0.02	+0.09	+0.13

4.2 Evaluation Measures

We evaluate all methods using Top-k evaluation measures, i.e., we consider a prediction to be correct if the ground truth action label is included in the top-k predictions. As observed in previous works [2], [11], this evaluation scheme is appropriate given the uncertainty of future predictions (i.e., many plausible actions can be performed after an observation). Specifically, we use the Top-5 accuracy as a class-agnostic measure and the Mean Top-5 Recall as a class aware metric. The Top-5 recall for a given class c is defined as the fraction of samples of ground truth class c for which the class c is in the list of the top-5 anticipated actions [11]. The mean Top-5 Recall is obtained by averaging the Top-5 recall values over classes. When evaluating on EPIC-Kitchens, Top-5 Recalls are averaged over the provided list of many-shot verbs, nouns and actions. Results on the EPIC-Kitchens official test set are reported using the suggested evaluation measures, i.e., Top-1 accuracy, Top-5 accuracy, Precision and Recall. Early action recognition and action recognition models are evaluated using Top-1 accuracy.

To assess the timeliness of anticipations, we propose a novel evaluation measure inspired by the AMOC curve [48]. Let s_t be the action scores predicted at time-step t for an action of ground truth class c . Let τ_t be the anticipation time at time-step t , and $tk(s_t)$ be the set of top- k actions as ranked by the action scores s_t . We define as “time to action” at rank k the largest anticipation time (i.e., the time of earliest anticipation) in which a correct prediction has been made according to the Top- k criterion:

$$TtA(k) = \max\{\tau_t | c \in tk(s_t), \forall t\} \quad (17)$$

If an action is not correctly anticipated in any of the time-steps, we set $TtA(k) = 0$. The mean time to action over the whole test set $mTtA(k)$ indicates how early, in average, a method can anticipate actions.

The time to action measure can be extended also to the case of early action recognition. If the current video is composed by N frames, we define the observation ratio at time-step t as $OR(t) = \frac{t}{N}$. This number can also be interpreted as a percentage, which defines how much of the action has been observed so far. We hence define as the “Minimum Observation Ratio” (MOR) the smallest observation ratio in which a correct prediction has been made according to the Top-1 criterion:

$$MOR = \min\{OR(t) | c = argmax_j\{s_{t,j}\}, \forall t\}. \quad (18)$$

We evaluated performances for verb, noun and action predictions on the EPIC-Kitchens and EGTEA Gaze+ datasets. We obtained verb and noun scores by marginalization over the action scores for all methods except the one proposed in [10], which predicts verb and noun scores directly.

4.3 Compared Methods

We compare the proposed method with respect to several state-of-the approaches and baselines. Specifically, we consider the Deep Multimodal Regressor (DMR) proposed in [9], the Anticipation Temporal Segment Network (ATSN) of [10], the anticipation Temporal Segment Network trained with verb-noun Marginal Cross Entropy Loss (MCE) described in [11], and the Encoder-Decoder LSTM (ED) introduced in [8]. We also consider two standard sequence-to-sequence models: a single LSTM architecture [65], [66] (LSTM), and Temporal Convolutional Networks [72] (TCN). We further compare our approach with respect to the following methods originally proposed for the task of early action recognition: a single LSTM architecture (we use the same parameters as our R-LSTM) trained using the Ranking Loss on Detection Score proposed in [7] (RL), an LSTM trained using the Exponential Anticipation Loss proposed in [14] (EL), and the Feedback Network LSTM (FN) proposed in [51]. Note that, being essentially sequence-to-sequence models, these approaches can be easily adapted to the considered action anticipation scenario. All these baselines adopt the video processing scheme illustrated in Fig. 2. Among them, LSTM, RL, FN and EL are implemented as two stream networks with a spatial and a temporal branch whose predictions are fused by late fusion. In our experiments, TCN obtained very low performance when processing optical flows on the EPIC-Kitchens and EGTEA Gaze+ datasets. In these cases, fusing the RGB and Flow branches actually resulted in lower performances than the RGB branch alone. On the contrary, on the ActivityNet dataset, fusing the RGB and Flow branches led to better performance. Hence, we implemented TCN as a single RGB branch on EPIC-Kitchens and EGTEA Gaze+ and as a two-branch network with late fusion on ActivityNet.³ Additionally, we compare our approach with Two-Stream CNNs (2SCNN) [23] and the method proposed by Miech et al [73] on the official test sets of EPIC-Kitchens.

³ The reader is referred to Appendix B for the implementation details of the considered methods.

5 RESULTS

This section compares the performance of the proposed Rolling-Unrolling LSTMs with other state-of-the-art approaches. Specifically, Sections 5.1-5.3 discuss the action anticipation results on the three considered datasets, Section 5.4 reports the ablation study on the EPIC-Kitchens dataset, whereas Section 5.5 reports some qualitative examples of the proposed method.

5.1 Egocentric Action Anticipation on EPIC-Kitchens

TABLE 1 compares RU with respect to the other state-of-the-art approaches on our validation set of the EPIC-Kitchens dataset. The left part of the table reports Top-5 action anticipation accuracy for the 8 considered anticipation times. Note that some methods [9], [10], [11] have been designed to anticipate actions only at a fixed anticipation time. The right part of the table reports the Top-5 accuracy and Mean Top-5 Recall for verbs, nouns and actions, for the fixed anticipation time of $\tau_a = 1s$, as well as the mean $TtA(5)$ scores obtained across the validation set. Best results are highlighted in bold, whereas second-best results are underlined. The last row reports the improvements obtained by RU with respect to second-best results. ED* denotes the Encoder-Decoder approach proposed in [8] without the unsupervised pre-training procedure proposed by the authors. These results are reported for reference.

The proposed approach outperforms all competitors by consistent margins according to all evaluation measures, obtaining an average improvement over prior art of about 5% with respect to Top-5 action anticipation accuracy on all anticipation times. The methods based on TSN (ATSN and MCE) tend to achieve low performance, which suggests the limits of simply adapting action recognition methods to the problem of anticipation. Interestingly, DMR and ED, which are explicitly trained to anticipate future representations, achieve sub-optimal Top-5 action anticipation accuracy as compared to methods trained to predict future actions directly from input images (e.g., compare DMR with MCE, and ED with FN/RL/EL/LSTM/TCN/RU). Comparing ED* to ED reveals that the unsupervised pre-training based on the regression of future representations is not beneficial in the considered problem of egocentric action anticipation. Indeed, in most cases the results achieved by the two methods are comparable. This might be due to the fact that anticipating future representations is very challenging in the case of egocentric video, in which the visual content tends to change continuously because of the mobility of the camera. The LSTM baseline consistently achieves second best results with respect to all anticipation times, except for $\tau_a = 0.25$, where it is outperformed by EL. This suggests that the loss functions employed in the RL and EL baselines, originally proposed for early action recognition in third person videos, are not effective in the case of egocentric action anticipation. TCN achieves very low performance as compared to most of the considered approach. This suggests that the non-recurrent nature of this approach is not very well suited to the considered anticipation problem, in which it is in general beneficial to refine predictions as more observations are processed. The proposed RU model is particularly strong on nouns, achieving a Top-5 noun accuracy of 51.79% and

a mean Top-5 noun recall of 49.90%, which improves over prior art by +7.26% and +7.31% respectively. The small drop in performance between class-agnostic and class-aware measures (i.e., 51.79% vs 49.90%) suggests that our method does not over-fit to the distribution of nouns seen during training set. It is worth noting that mean Top-5 Recall values are averaged over fairly large sets of 26 many-shot verbs, 71 many-shot nouns, and 819 many-shot actions, as specified in [10]. Differently, all compared methods obtain large drops in verb and action performance when comparing class-agnostic measures to class-aware measures. Our insight into this different pattern is that anticipating the next active object (i.e., anticipating nouns) is much less ambiguous than anticipating the way in which the object will be used (i.e., anticipating verbs and actions). It is worth noting that second best Top-5 verb and noun accuracy scores are obtained by different methods (i.e., ATSN in the case of verbs and RL in the case of nouns), while both are outperformed by the proposed RU. Despite its low performance when evaluated with class-agnostic measures, ED systematically achieves second best results with respect to mean Top-5 recall and mean $TtA(5)$. This highlights that there is no clear second-best performing method. Finally, the mean $TtA(k)$ highlights that the proposed method can anticipate verbs, nouns and actions 1.62, 1.11 and 0.76 seconds in advance respectively.

TABLE 2 compares the performance of the proposed method with baselines and other approaches on the official test sets of the EPIC-Kitchens dataset. RU-LSTM outperforms all competitors by consistent margins on both the “seen” test, which includes scenes appearing in the training set (**S1**) and on the “unseen” test set, with scenes not appearing in the training set (**S2**). Also in this case, RU is strong on nouns, obtaining +6.31% and +8.23% in **S1**, as well as +2.76% and +2.18 in **S2**. Improvements in terms of actions are also significant: +3.63% and +5.52% in **S1**, as well as +0.92% and +1.81% on **S2**.

5.2 Egocentric Action Anticipation on EGTEA Gaze+

TABLE 3 reports egocentric action anticipation results on EGTEA Gaze+. The proposed RU approach outperforms the competitors for all anticipation times except for $\tau_a = 2s$, in which case its performance is on par with the LSTM baseline. Note that the margins of improvement obtained by the proposed method are smaller on EGTEA Gaze+, probably due to its smaller scale (106 actions in vs 2, 513 actions in EPIC-KITCHENS). Second-best results are achieved by EL for most of the anticipation times, except for $\tau_a \in \{1.25s, 1.0s\}$, where LSTM achieves comparable results. The table shows similar trends to the ones observed in the case of EPIC-Kitchens. The methods based on the direct regression of future representations such as DMR and ED still achieve sub-optimal results, especially as compared to other sequence-to-sequence models. Interestingly, ED* achieves better results than ED, which seem to confirm the limited ability of approaches based on direct regression of future representations in the egocentric domain. Also in this case, the performance of TCN tends to be somewhat limited as compared to recurrent approaches. Since no object annotations are available for EGTEA Gaze+, our RU

TABLE 2
Egocentric action anticipation results on the EPIC-Kitchens test set.

	Top-1 Accuracy%			Top-5 Accuracy%			Avg Class Precision%			Avg Class Recall%			
	VERB	NOUN	ACTION	VERB	NOUN	ACTION	VERB	NOUN	ACTION	VERB	NOUN	ACTION	
S1	DMR [9]	26.53	10.43	01.27	73.30	28.86	07.17	06.13	04.67	00.33	05.22	05.59	00.47
	2SCNN [10]	29.76	15.15	04.32	76.03	38.56	15.21	13.76	17.19	02.48	07.32	10.72	01.81
	ATSN [10]	31.81	16.22	06.00	76.56	42.15	28.21	23.91	19.13	03.13	09.33	11.93	02.39
	MCE [11]	27.92	16.09	10.76	73.59	39.32	25.28	23.43	17.53	06.05	14.79	11.65	05.11
	ED [8]	29.35	16.07	08.08	74.49	38.83	18.19	18.08	16.37	05.69	13.58	14.62	04.33
	Miech et al. [73]	30.74	16.47	09.74	76.21	42.72	25.44	12.42	16.67	03.67	08.80	12.66	03.85
	RU-LSTM	33.04	22.78	14.39	79.55	50.95	33.73	25.50	24.12	07.37	15.73	19.81	07.66
S2	Imp. wrt best	+1.23	+6.31	+3.63	+2.99	+8.23	+5.52	+1.59	+4.99	+1.32	+0.94	+5.19	+2.55
	DMR [9]	24.79	08.12	00.55	64.76	20.19	04.39	09.18	01.15	00.55	05.39	04.03	00.20
	2SCNN [10]	25.23	09.97	02.29	68.66	27.38	09.35	16.37	06.98	00.85	05.80	06.37	01.14
	ATSN [10]	25.30	10.41	02.39	68.32	29.50	06.63	07.63	08.79	00.80	06.06	06.74	01.07
	MCE [11]	21.27	09.90	05.57	63.33	25.50	15.71	10.02	06.88	01.99	07.68	06.61	02.39
	ED [8]	22.52	07.81	02.65	62.65	21.42	07.57	07.91	05.77	01.35	06.67	05.63	01.38
	Miech et al. [73]	28.37	12.43	07.24	69.96	32.20	19.29	11.62	08.36	02.20	07.80	09.94	03.36
RU-LSTM	27.01	15.19	08.16	69.55	34.38	21.10	13.69	09.87	03.64	09.21	11.97	04.83	
	Imp. wrt best	-1.36	+2.76	+0.92	-0.41	+2.18	+1.81	-2.68	+1.08	+1.44	+1.41	+2.03	+1.47

TABLE 3
Egocentric action anticipation results on EGTEA Gaze+.

	Top-5 ACTION Accuracy% @ different τ_a (s)							Top-5 Acc.% @1s			M. Top-5 Rec.% @1s			Mean TtA(5)			
	2	1.75	1.5	1.25	1.0	0.75	0.5	0.25	VERB	NOUN	ACT.	VERB	NOUN	ACT.	VERB	NOUN	ACT.
DMR [9]	/	/	/	/	55.70	/	/	/	92.78	71.36	55.70	70.22	53.92	38.11	/	/	/
ATSN [10]	/	/	/	/	40.53	/	/	/	90.60	69.94	40.53	69.24	57.02	31.61	/	/	/
MCE [11]	/	/	/	/	56.29	/	/	/	90.73	70.02	56.29	72.38	58.67	43.75	/	/	/
TCN [72]	49.86	51.05	54.08	55.17	58.50	59.34	62.87	65.53	91.10	71.94	58.50	73.36	63.11	47.14	01.86	01.58	01.33
ED* [8]	52.91	54.16	56.22	58.31	60.18	62.57	64.77	67.05	91.12	73.50	60.18	78.19	68.33	54.61	01.87	01.57	01.34
ED [8]	45.03	46.22	46.86	48.36	50.22	51.86	49.99	49.17	86.79	64.35	50.22	69.66	56.62	42.74	01.84	01.40	01.24
FN [51]	54.06	54.94	56.75	58.34	60.12	62.03	63.96	66.45	91.05	71.64	60.12	76.73	63.59	49.82	01.83	01.39	01.26
RL [7]	55.70	56.45	58.65	60.69	62.74	64.37	67.02	69.33	91.54	74.51	62.74	78.55	67.10	52.17	01.84	01.43	01.29
EL [14]	55.05	56.75	58.81	61.00	63.76	66.37	69.12	72.33	91.77	75.68	63.76	79.63	69.93	55.11	01.85	01.47	01.32
LSTM [66]	56.88	58.23	59.87	61.83	63.87	65.84	67.70	70.65	91.56	75.30	63.87	78.27	68.43	53.35	01.85	01.54	01.33
RU-LSTM	56.82	59.13	61.42	63.53	66.40	68.41	71.84	74.28	93.11	77.48	66.40	82.07	73.30	58.64	01.88	01.61	01.41
Improv.	-0.06	+0.90	+1.55	+1.70	+2.53	+2.04	+2.72	+1.95	+0.33	+1.80	+2.53	+2.44	+3.37	+3.53	+0.01	+0.03	+0.07

TABLE 4
Anticipation results on ActivityNet.

	Top-5 Accuracy% @ different τ_a (s)							Top-1%		M.T-5 Rec.%		M.TtA(5)	
	2	1.75	1.5	1.25	1.0	0.75	0.5	0.25	1	1	1	/	
DMR [9]	/	/	/	/	52.39	/	/	/	24.13	39.55	39.55	/	
ATSN [10]	/	/	/	/	48.08	/	/	/	29.09	45.94	45.94	/	
TCN [72]	52.71	53.55	55.21	56.69	58.39	59.42	60.85	62.06	34.15	57.21	57.21	01.27	
ED* [8]	62.89	63.71	64.23	65.27	65.84	67.09	68.16	69.79	42.83	64.98	64.98	01.40	
ED [8]	70.20	70.45	71.04	71.75	72.93	72.95	72.52	71.43	48.58	72.32	72.32	01.54	
FN [51]	59.44	60.01	60.79	61.38	62.29	63.32	64.16	65.34	37.57	60.65	60.65	01.27	
RL [7]	65.11	65.66	66.56	67.51	68.71	69.78	71.15	72.48	45.27	67.68	67.68	01.40	
EL [14]	63.21	63.99	65.15	66.03	67.05	68.25	69.49	71.15	43.06	66.17	66.17	01.37	
LSTM [66]	61.61	62.92	63.72	64.31	65.70	66.71	67.87	69.24	40.37	64.44	64.44	01.34	
RU-LSTM	65.53	66.67	67.59	69.13	70.23	71.66	72.73	73.97	46.49	69.41	69.41	01.45	
Imp	-4.67	-3.78	-3.45	-2.62	-2.70	-1.29	+0.21	+1.49	-2.09	-2.91	-2.91	-0.09	

model uses the object detector trained on EPIC-Kitchens for the object branch. Despite the use of object classes not perfectly aligned with the ones contained in the dataset, our approach is strong on nouns even in this case, obtaining an improvement of +1.80% and +3.37% with respect to Top-5 accuracy and mean Top-5 recall.

5.3 Action Anticipation on ActivityNet

Table 4 reports the results on the third person ActivityNet dataset. Interestingly, in this context, ED significantly outperforms ED* and achieves top performances for most of the anticipation times. This is in line with the findings of the authors of the approach [8], and highlights the different nature of the egocentric scenario as compared to the

rather static third person scenario. While the proposed RU model is outperformed by ED on most anticipation times, it systematically achieves second-best performance and has a significant advantage over ED*, which, similarly to the proposed method, does not make use of the unsupervised pre-training. Differently from previous results, in this case, both RL and EL outperform the LSTM baseline, which suggests that the anticipation losses used in these baselines are more beneficial in the case of third person videos than in the case of first-person videos. This highlights again the differences between the two scenarios. Also in these experiments, TCN achieve worse performance as compared to recurrent models.

TABLE 5
Ablation study on EPIC-KITCHENS.

	Top-5 ACTION Accuracy% @ different τ_a (s)								TtA
	2	1.75	1.5	1.25	1.0	0.75	0.5	0.25	
BL (Late)	27.96	28.76	29.99	31.09	32.02	33.09	34.13	34.92	0.66
RU (Late)	29.10	29.77	31.72	33.09	34.23	35.28	36.10	37.61	0.73
Imp.	+1.14	+1.01	+1.73	+2.00	+2.21	+2.19	+1.97	+2.69	+0.07

(a) Rolling-Unrolling Mechanism.

	RU (RGB)	25.44	26.89	28.32	29.42	30.83	32.00	33.31	34.47	0.69
RU (Flow)	17.38	18.04	18.91	19.97	21.42	22.37	23.49	24.18	24.51	
RU (OBJ)	24.56	25.60	26.61	28.32	29.89	30.85	31.82	33.39	0.67	
Early Fusion	25.58	27.25	28.58	29.59	31.88	32.78	33.99	35.62	0.72	
Late Fusion	29.10	29.77	31.72	33.09	34.23	35.28	36.10	37.61	0.73	
MATT	29.44	30.73	32.24	33.41	35.32	36.34	37.37	38.98	0.76	
Imp.	+0.34	+0.96	+0.52	+0.32	+1.09	+1.06	+1.27	+1.37	+0.03	

(b) Modality Attention Fusion Mechanism.

	Flow+OBJ	21.10	21.24	21.84	23.05	23.93	25.00	26.11	26.45	0.57
RGB+Flow	26.75	27.43	29.20	30.15	32.16	33.49	34.37	35.46	0.70	
RGB+OBJ	28.04	29.51	31.48	32.22	34.27	35.36	36.89	37.79	0.74	
RGB+Flow+OBJ	29.44	30.73	32.24	33.41	35.32	36.34	37.37	38.98	0.76	

(c) MATT fusion with different modalities.

	w/o SCP	29.22	30.43	32.34	33.37	34.75	35.84	36.79	37.93	0.75
with SCP		29.44	30.73	32.24	33.41	35.32	36.34	37.37	38.98	0.76
Imp. of SCP		+0.22	+0.30	-0.10	+0.04	+0.57	+0.50	+0.58	+1.05	+0.01

(d) Sequence-Completion Pre-training.

	BL (Fusion)	27.96	28.76	29.99	31.09	32.02	33.09	34.13	34.92	0.66
RU (Fusion)	29.44	30.73	32.24	33.41	35.32	36.34	37.37	38.98	0.76	
Imp. (Fusion)	+1.48	+1.97	+2.25	+2.32	+3.30	+3.25	+3.24	+4.06	+0.1	

(e) Overall comparison wrt strong baseline.

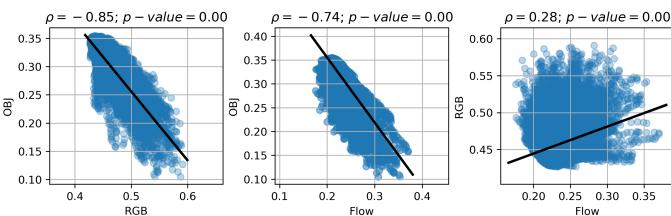


Fig. 7. Correlations between modality attention weights

5.4 Ablation Study on EPIC-Kitchens

We performed an ablation study on the EPIC-Kitchens dataset to assess the role of the different components involved in our architecture. Specifically, to assess the role of the proposed rolling-unrolling mechanism, we considered a strong baseline composed of a single LSTM (same configuration as R-LSTM) and three branches (RGB, Flow, OBJ) with late fusion (BL). Note that, differently from the LSTM baseline compared in the previous sections, this baseline also includes an object branch. To study the role of rolling-unrolling in isolation, we compare this baseline with respect to a variant of the proposed RU architecture in which MATT has been replaced with late fusion in TABLE 5(a). As can be observed, the rolling-unrolling mechanism brings systematic improvements over the strong baseline for all anticipation times.

In TABLE 5(b), we study the influence of MATT by comparing it with respect to two standard fusion approaches: early fusion (i.e., feeding the model with the concatenation of the modality-specific representations) and late fusion (i.e., averaging predictions). MATT always outperforms late fusion, which consistently achieves second best results, while early fusion always leads to sub-optimal results. All fusion

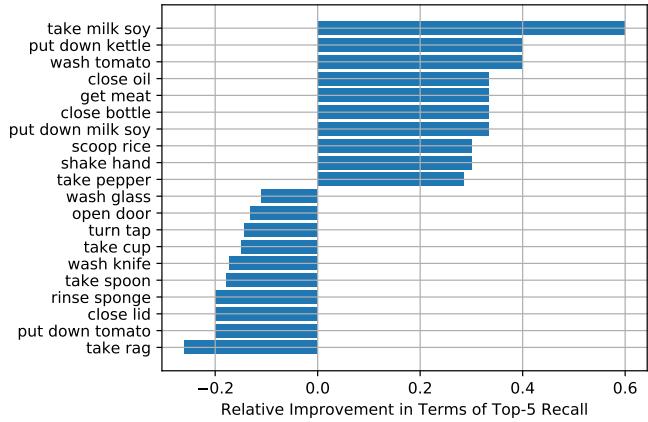
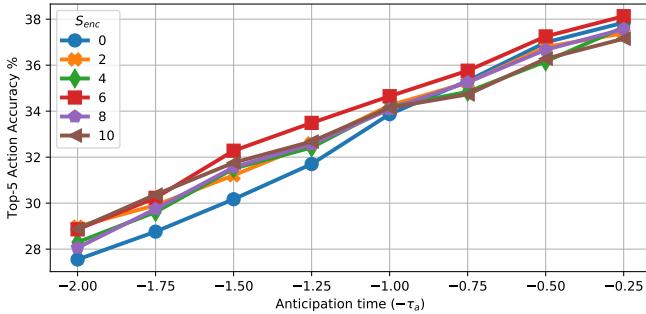


Fig. 8. Top-10 and bottom-10 actions which benefited from modality attention in terms of Top-5 Recall.

schemes always improve over the single branches. Fig. 7 shows regression plots of the modality attention weights computed by the proposed method on all the samples of the validation set. The RGB and OBJ weights are characterized by a strong and steep correlation. A similar pattern is observed between Flow and OBJ weights, whereas Flow and RGB weights are characterized by a small positive correlation. This suggests that MATT gives more credit to OBJ when RGB and Flow are less informative, whereas it relies on RGB and Flow when the detected objects are not informative. Figure 8 shows the top-10 and bottom-10 action categories which benefited from MATT as compared to late fusion, in terms of mean Top-5 Recall. For the analysis, we have considered only classes containing at least 5 instances in the validation set. We can observe significant improvements for some actions such as “take milk soy” and “put down kettle”, while there are relatively small negative performance differences with respect to late fusion for some actions such as “take rag” and “put down tomato”.

TABLE 5(c) compares the performances of different versions of the proposed architecture in which MATT is used to fuse different subsets of the considered modalities. Fusing RGB with optical flow (RGB+Flow) or objects (RGB+OBJ) allows to improve over the respective single-branches. Fusing optical flow and objects (Flow+OBJ) improves over the Flow branch, but does not improve over the OBJ branch, while adding RGB (RGB+Flow+OBJ) does allow to improve over the single branches. This suggests that the model is not able to take advantage of representations based on optical flow when the RGB signal is not available. Interestingly, fusing RGB and objects (RGB+OBJ) allows to obtain better results than fusing RGB and optical flow (RGB+Flow), as it is generally considered in many standard pipelines. This further highlights the importance of objects for egocentric action anticipation. Fusing all modalities leads to the best performance.

In TABLE 5(d), we assess the role of Sequence Completion Pre-Training (SCP). The proposed pre-training procedure brings small but consistent improvements for most anticipation times. TABLE 5(e) compares RU with the strong baseline of TABLE 5(a). The comparison shows the cumula-

Fig. 9. Impact of the choice of S_{enc} on performance.

tive effect of all the proposed procedures/component with respect to a strong baseline which uses three modalities. It is worth noting that the proposed architecture brings improvements for all anticipation times, ranging from +1.48% to +4.06%.

Figure 9 finally investigates the effect of choosing different values of S_{enc} , while the number of anticipation steps is fixed to $S_{ant} = 8$. As can be noted, our approach achieves best results across most of the anticipation times for $S_{enc} = 6$, while smaller and larger number of encoding steps lead to lower performance. This suggests that, while a sufficiently long temporal context is required to correctly anticipate actions, leveraging very long temporal contexts can be challenging.

5.5 Qualitative Results

Fig. 10 reports two qualitative examples of predictions made by the proposed approach at four anticipation times. Under each example, are reported the top-4 predictions, whereas modality weights computed by MATT are reported in percentage on the right. We show green bounding boxes around the detected objects and orange arrows to illustrate optical flow. In the first example (top), the model can predict “close door” based on the context and the history of past actions (e.g., taking objects out of the cupboard), hence it assigns large weights to the RGB and Flow modalities and low weights to the OBJ modality. In the second example (bottom), the model initially predicts “squeeze lime” at $\tau_a = 2s$. Later, as the lemon is predicted, the prediction is corrected to “squeeze lemon”. Note that in this case the network assigns larger weights to OBJ as compared to the previous example.⁴

6 ADDITIONAL RESULTS ON EARLY ACTION RECOGNITION AND ACTION RECOGNITION

We note that, being a sequence-to-sequence method, our approach can also be employed to perform early action recognition. This is done by processing the video of an action sequentially and outputting a prediction at every time-step. The prediction performed at the last time-step can be also used action recognition.

4. See Appendix C and <https://iplab.dmi.unict.it/rulstm/> for additional examples and videos.

6.1 Early Action Recognition

We adapt all sequence-to-sequence models to perform early action recognition by sampling 8 video snippets from each action segment uniformly and set $S_{enc} = 0$, $S_{ant} = 8$. The models produce predictions at each time-step, corresponding to the observation rates: 12.5%, 25%, 37.5%, 50%, 62.5%, 75%, 87.5%, 100%. Modality-specific branches are fused by late fusion. We compared our approach with respect to the following baselines: FN, RL, EL, LSTM, TCN.

TABLE 6 reports the Top-1 accuracy results obtained by the compared methods with respect to different observation rates on our validation set of EPIC-Kitchens. *MOR* scores are reported for verbs, nouns and actions. Best results are highlighted in bold numbers. Note that, differently from Top-1 accuracy, lower *MOR* scores are better than higher *MOR* scores, meaning that the method can, in average, recognize an action by observing less frames. The proposed method consistently outperforms the competitors at all observation rates by about +2% in average. Interestingly, RU achieves an early action recognition accuracy of 33.09% when observing only 75% of the action, which is already comparable to the accuracy of 34.07% achieved when the full action is observed. Also, the proposed method achieves *MOR* values lower than the competitors, meaning that, in average, it can predict action correctly by observing less frames. This suggests that RU can timely recognize actions before they are completed. Second-best results are obtained by the LSTM baseline in most cases, indicating that the losses employed by RL and EL are not effective on this dataset for early action recognition. Similarly to what observed previously, TCN achieves sub-optimal results as compared to the recurrent approaches.

TABLE 7 reports the Top-1 accuracy results on EGTEA Gaze+. The proposed RU is outperformed by the LSTM baseline for observation rates up to 50%, while it performs comparably to the competitors for the other observation rates. Also, the *MOR* values suggest that the proposed approach predicts actions by observing marginally less video content. Second-best results are obtained by different methods, and there is not a clear second-best performer in this case. Interestingly, TCN achieves performances comparable with the recurrent methods on this dataset.

TABLE 8 reports the early action recognition results obtained by the different methods on ActivityNet. Interestingly, the RL baseline achieves best results or second-best results for most of the observation rates. This suggests that the loss employed by this baseline is effective for early action recognition in the domain of third person videos, which is the scenario for which RL has been originally designed. As we found the RL loss beneficial, we trained the compared RU method with this loss on this dataset. The proposed approach achieves performances comparable in average with the other methods, but obtains a *MOR* smaller by 5.8%, which highlights that it can recognize actions by observing 5.8% less video content, in average. Also in this case, the performance of TCN are lower as compared to the recurrent approaches.

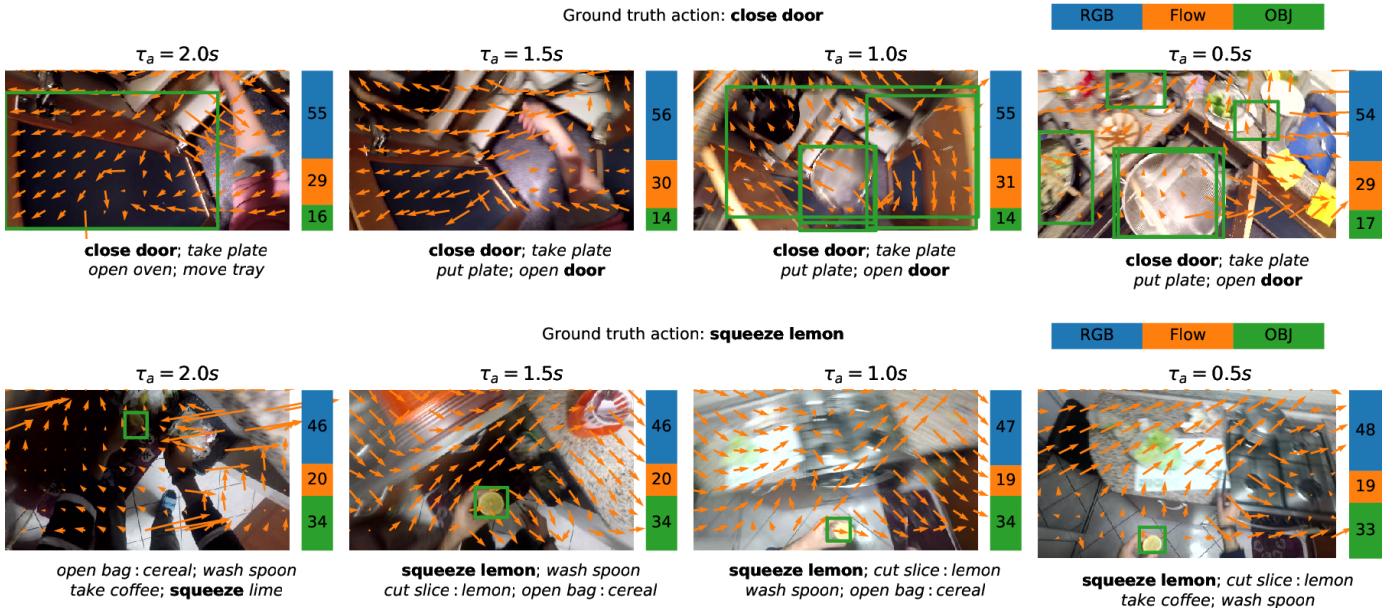


Fig. 10. Qualitative examples (best seen on screen). Legend for attention weights: blue - RGB, orange - Flow, green - objects.

TABLE 6
Early recognition results on EPIC-KITCHENS.

	Top-1 ACTION Accuracy% @ different observation rates								MOR%		
	12.5%	25.0%	37.5%	50.0%	62.5%	75.0%	87.5%	100%	VERB	NOUN	ACT.
TCN [72]	16.93	18.42	19.27	19.05	20.62	20.53	19.45	17.12	53.89	63.49	76.29
FN [51]	19.61	23.85	25.66	26.85	27.47	28.34	28.26	28.38	51.12	65.12	74.49
RL [7]	22.53	25.08	27.19	28.64	29.57	30.13	30.45	30.47	51.29	63.59	73.13
EL [14]	19.69	23.27	26.03	27.49	29.06	29.97	30.91	31.46	51.89	64.00	74.10
LSTM [66]	22.16	25.78	27.80	28.98	29.87	31.13	31.28	30.93	50.64	63.27	72.60
RU-LSTM	24.48	27.63	29.44	30.93	32.16	33.09	33.63	34.07	47.91	59.15	69.34
Imp.	+1.95	+1.85	+1.64	+1.95	+2.29	+1.96	+2.35	+2.61	-2.73	-4.12	-3.26

TABLE 7
Early recognition results on EGTEA Gaze+.

	Top-1 ACTION Accuracy% @ different observation rates								MOR%		
	12.5%	25.0%	37.5%	50.0%	62.5%	75.0%	87.5%	100%	VERB	NOUN	ACT.
TCN [72]	49.61	52.88	55.47	56.24	56.90	57.58	57.62	56.11	34.58	33.54	45.13
FN [51]	44.02	50.32	53.34	55.10	56.58	57.31	57.95	57.72	37.07	37.80	48.94
RL [7]	45.42	51.00	54.20	56.54	58.09	58.93	59.29	59.50	37.80	36.86	48.43
EL [14]	40.31	48.08	51.84	54.71	56.93	58.45	59.55	60.18	38.91	37.48	49.93
LSTM [66]	50.22	53.82	55.73	57.20	58.01	58.79	59.09	59.32	36.20	35.46	46.89
RU-LSTM	45.94	51.84	54.39	57.05	58.15	59.31	60.10	60.20	34.91	34.64	45.57
Imp.	-4.28	-1.98	-1.34	-0.15	0.06	0.38	0.55	0.02	0.33	1.10	0.44

6.2 Action Recognition

We finally compare the performance of our method with respect to other state-of-the-art approaches on the task of action recognition on EPIC-Kitchens and EGTEA Gaze+. We do not assess the performance of our approach on ActivityNet as this dataset is generally used by the community for action localization rather than recognition. Although our method does not generally outperform the competitors, it achieves competitive results in some cases.

TABLE 9 compares the performance of the proposed method with the state-of-the-art approaches to egocentric action recognition on the two test sets of EPIC-Kitchens. Being designed for early egocentric action anticipation, the proposed RU approach does not outperform the competitors, but achieves competitive results with the state-of-the-

art, obtaining -4.44% and -3% on Top-1 and Top-5 action accuracy. Also, it outperforms several action recognition baselines such as TSN, 2SCNN, TRN and TSM on Top-1 and Top-5 action accuracy.

Table 10 reports the action recognition results on EGTEA Gaze+. Despite being designed for action anticipation, RU outperforms recent approaches, such as Li et al. [16] ($+6.9\%$ wrt 53.3%) and Zhang et al. [78] ($+3.19\%$ wrt 57.01% - reported from [44]), and obtains performances comparable to state-of-the-art approaches such as Sudhakaran and Lanz [45] (-0.56% wrt 60.76%) and Sudhakaran et al. [44] (-1.66% wrt 61.86%).

TABLE 8
Early recognition results on ActivityNet.

	Top-1 ACTION Accuracy% @ different observation rates								MOR%
	12.5%	25.0%	37.5%	50.0%	62.5%	75.0%	87.5%	100%	
TCN [72]	50.03	54.85	56.73	58.92	60.60	61.55	61.51	60.71	41.65
FN [51]	53.74	58.94	61.82	63.82	65.51	66.22	67.00	67.68	42.69
RL [7]	55.44	62.41	65.23	67.40	68.65	69.83	70.35	70.56	39.23
EL [14]	55.51	61.74	64.92	66.69	68.22	68.98	69.66	70.50	39.22
LSTM [66]	55.99	61.91	65.05	67.00	68.31	69.07	69.45	70.20	39.16
RU+RL	55.74	62.01	64.81	66.67	68.58	70.27	70.69	71.17	36.89
Imp.	-0.25	-0.40	-0.42	-0.73	-0.07	+0.44	+0.34	+0.61	-5.80

TABLE 9
Egocentric action recognition results on the EPIC-Kitchens test set.

	Top-1 Accuracy%			Top-5 Accuracy%			Avg Class Precision%			Avg Class Recall%		
	VERB	NOUN	ACTION	VERB	NOUN	ACTION	VERB	NOUN	ACTION	VERB	NOUN	ACTION
2SCNN [10]	42.16	29.14	13.23	80.58	53.70	30.36	29.39	30.73	05.53	14.83	21.10	04.46
TSN [10]	48.23	36.71	20.54	84.09	62.32	39.79	47.26	35.42	10.46	22.33	30.53	08.83
LSTA [44]	59.55	38.35	30.33	85.77	61.49	49.97	42.72	36.19	14.46	38.12	36.19	17.76
VNMCE [11]	54.22	38.85	29.00	85.22	61.80	49.62	53.87	38.18	18.22	35.88	32.27	16.56
TRN [28]	61.12	39.28	27.86	87.71	64.36	47.56	52.32	35.68	16.38	32.93	34.18	14.36
TSM [74]	62.37	41.88	29.90	88.55	66.43	49.81	59.51	39.50	18.38	34.44	36.04	15.80
TBN [75]	64.75	46.03	34.80	90.70	71.34	56.65	55.67	43.65	22.07	45.55	42.30	21.31
Miech et al. [73]	43.51	32.94	20.19	84.38	61.66	43.57	28.42	27.99	07.62	24.18	26.83	08.85
FAIR [76]	64.14	47.65	35.75	87.64	70.66	54.65	43.64	40.53	18.95	38.31	45.29	21.13
FB [77]	60.00	45.00	32.70	88.40	71.80	55.32	/	/	/	/	/	/
RU-LSTM	56.93	43.05	33.06	85.68	67.12	55.32	50.42	39.84	18.91	37.82	38.11	19.12
Imp.	-07.82	-04.60	-02.69	-05.02	-04.68	-01.33	-09.09	-03.81	-03.16	-07.73	-07.18	-02.19
2SCNN [10]	36.16	18.03	07.31	71.97	38.41	19.49	18.11	15.31	02.86	10.52	12.55	02.69
TSN [10]	39.40	22.70	10.89	74.29	45.72	25.26	22.54	15.33	05.60	13.06	17.52	05.81
LSTA [44]	47.32	22.16	16.63	77.02	43.15	30.93	31.57	17.91	08.97	26.17	17.80	11.92
VNMCE [11]	40.90	23.46	16.39	72.11	43.05	31.34	26.62	16.83	07.10	15.56	17.70	10.17
TRN [28]	51.62	26.02	17.34	78.42	48.99	32.57	32.47	19.99	09.45	21.63	21.53	10.11
TSM [74]	51.96	25.61	17.38	79.21	49.47	32.67	27.43	17.63	09.17	20.19	22.93	11.18
TBN [75]	52.69	27.86	19.06	79.93	53.78	36.54	31.44	21.48	12.00	28.21	23.53	12.69
Miech et al. [73]	39.30	22.43	14.10	76.41	47.35	32.43	20.42	15.96	04.83	16.95	17.72	08.46
FAIR [76]	55.24	33.87	23.93	80.23	58.25	40.15	25.71	28.19	15.72	25.69	29.51	17.06
FB [77]	50.90	31.50	21.20	77.60	57.80	39.40	/	/	/	/	/	/
RU-LSTM	43.67	26.77	19.49	73.30	48.28	37.15	23.40	20.82	09.72	18.41	21.59	13.33
Imp.	-11.57	-07.10	-04.44	-06.93	-09.97	-03.00	-09.07	-07.37	-06.00	-09.80	-07.92	-03.73

TABLE 10
Recognition results on EGTEA Gaze+.

Method	Acc.%	Imp.
Lit et al. [39]	46.50	+13.7
Li et al. [16]	53.30	+6.90
Two stream [23]	41.84	+18.7
I3D [31]	51.68	+8.52
TSN [27]	55.93	+4.27
eleGAtt [78]	57.01	+3.19
ego-rnn [45]	60.76	-0.56
LSTA [44]	61.86	-1.66
RU-LSTM	60.20	/

7 CONCLUSION

We presented Rolling-Unrolling LSTMs, an architecture for egocentric action anticipation. The proposed architecture includes two separate LSTMs, designed to explicitly disentangle two sub-tasks: summarizing the past (encoding) and predicting the future (inference). To encourage such disentanglement, the architecture is trained with a novel

sequence-completion pre-training. A modality attention network is introduced to fuse multi-modal predictions obtained by three branches processing RGB frames, optical flow fields and object-based features. Experiments on three benchmark datasets highlight that the proposed approach achieves state-of-the-art results on the task of action anticipation on both first-person and third-person scenarios, and generalizes to the tasks of early action recognition and action recognition. To encourage research on the topic, we publicly released the source code of the proposed approach, together with pre-trained models and extracted features at our project web page: <http://iplab.dmi.unict.it/rulstm>.

ACKNOWLEDGMENTS

This research is supported by Piano della Ricerca 2016-2018, linea di Intervento 2 of DMI, University of Catania and MIUR AIM - Attrazione e Mobilità Internazionale Linea 1 - AIM1893589 - CUP E64118002540007

APPENDIX

A IMPLEMENTATION AND TRAINING DETAILS OF THE PROPOSED METHOD

This section reports implementation and training details of the different components involved in the proposed method. The reader is also referred to the code available online for the implementation of the proposed approach: <https://iplab.dmi.unict.it/rulstm/>.

A.1 Architectural Details

We use a Batch Normalized Inception architecture [70] in the representation functions ϕ_1 and ϕ_2 of the spatial and motion branches. For the object branch, we use a Faster R-CNN object detector [79] with a ResNet-101 backbone [33], as implemented in [80]. Both the Rolling LSTM (R-LSTM) and the Unrolling LSTM (U-LSTM) contain a single hidden layer with 1024 units. Dropout with $p = 0.8$ is applied to the input of each LSTM and to the input of the final fully connected layer used to obtain class scores. The Modality ATTention network (MATT) is a feed-forward network with three fully connected layers containing respectively $h/4$, $h/8$ and 3 hidden units, where h is the dimension of the input to the attention network (i.e., the concatenation of the hidden and cell states of 1024 units of all modality-specific R-LSTMs). When three modalities are considered, we obtain an input of dimension $h = 3 \cdot 2 \cdot 1024 = 6144$. Dropout with $p = 0.8$ is applied to the input of the second and third layers of the attention network to avoid over-fitting. ReLU activations are used within the attention network.

A.2 Training Procedure

We train the spatial and motion CNNs for the task of ego-centric action recognition with TSN [27]. We set the number of segments of TSN to 3 and train the model with Stochastic Gradient Descent (SGD) and standard cross entropy for 160 epochs with an initial learning rate equal to 0.001, which is decreased by a factor of 10 after 80 epochs. We use a mini-batch size of 64 samples and train the models on a single Titan X GPU. For all other parameters, we use the values recommended in [27]. We train the object detector to recognize the 352 object classes of the EPIC-Kitchens dataset. We use the same object detector trained on EPIC-Kitchens when performing experiments on EGTEA Gaze+, as the latter dataset does not contain object bounding box annotations. We do not use an object branch in the case of ActivityNet. This training procedure allows to learn the parameters θ^1 , θ^2 and θ^3 of the representation functions related to the three modalities (i.e., RGB, Flow, OBJ). After this procedure, these parameters are fixed and they are no further optimized. For efficiency, we pre-compute representations over the whole dataset.

Each branch of the RU-LSTM is trained with SGD and the cross entropy loss with a fixed learning rate equal to 0.01 and a momentum equal to 0.9. The loss is averaged both across the samples of the mini-batches and across the predictions obtained at different time-stamps. Each branch is first pre-trained with Sequence Completion Pre-training (SCP). Specifically, appearance and motion branches are trained for 100 epochs, whereas the object branch is trained

for 200 epochs. The branches are then fine-tuned for the action anticipation task. Once each branch has been trained, the complete architecture with three branches is assembled using MATT to form a three-branch network and the model is further fine-tuned for 100 epochs using cross entropy and the same learning parameters. In the case of early action recognition, each branch is trained for 200 epochs (both SCP and main task) with a fixed learning rate equal to 0.01 and momentum equal to 0.9.

We apply early stopping at each training stage. This is done by choosing the iterations of the intermediate and final models which obtain the best Top-5 action anticipation accuracy for the anticipation time $\tau_a = 1s$ on the validation set. In the case of early action recognition, we choose the epoch in which we obtain the best average Top-1 action accuracy across observation rates. The same early stopping strategy is applied to all the methods for fair comparison. The proposed architecture has been implemented using the PyTorch library [81].

B IMPLEMENTATION AND TRAINING DETAILS OF THE COMPARED METHODS

Since no official public implementations are available for the compared methods, we performed experiments using our own implementations. In this section, we report the implementation details of each of the compared method.

B.1 Deep Multimodal Regressor (DMR)

We implemented the Deep Multimodal Regressor proposed in [9] setting the number of multi-modal branches with interleaved units to $k = 3$. For fair comparisons, we substituted the AlexNet backbone originally considered in [9] with a BNInception CNN pre-trained on ImageNet. The CNN has been trained to anticipate future representations extracted using BNInception pre-trained on ImageNet using the procedure proposed by the authors. Specifically, we performed mode update every epoch. Since training an SVM with large number of classes is challenging (we have 2,513 different action classes in the case of EPIC-Kitchens), we substituted the SVM with a Multi Layer Perceptron (MLP) with 1024 hidden units and dropout with $p = 0.8$ applied to the input of the first and second layer. To comply with the pipeline proposed in [9], we pre-trained the model in an unsupervised fashion and then trained the MLP separately on representations pre-extracted from the training set using the optimal modes found at training time. As a result, during the training of the MLP, the weights of the CNN are not optimized. The DMR architecture has been trained with Stochastic Gradient Descent using a fixed learning rate equal to 0.1 and a momentum equal to 0.9. The network has been trained for several epochs until the validation loss saturated. Note that training the CNN on the EPIC-Kitchens dataset takes several days on a single Titan X GPU using our implementation. After training the DMR, we applied early stopping by selecting the iteration with the lowest validation loss. The MLP has then been trained with Stochastic Gradient Descent with fixed learning rate equal to 0.01 and momentum equal to 0.9. Early stopping has been applied also in this case considering the iteration

of the MLP achieving the highest Top-5 action accuracy on the validation set.

B.2 Anticipation TSN (ATSN)

We implemented this model following [10]. Specifically, the model has been trained using TSN with 3 segments. We modified the network to output verb and noun scores and trained it summing the cross entropy losses applied independently to verbs and nouns. At test time, we obtained action probabilities by assuming conditional independence of verbs and nouns given the sample as follows: $p(a = (v, n)|x) = p(v|x) \cdot p(n|x)$, where $a = (v, n)$ is an action involving verb v and noun n , x is the input sample, whereas $p(v|x)$ and $p(n|x)$ are the verb and noun probabilities computed directly by the network.

B.3 ATSN + VNMCE Loss (MCE)

This method has been implemented training the TSN architecture used in the case of ATSN with the Verb-Noun Marginal Cross Entropy Loss proposed in [11]. We used the official code available at <https://github.com/fpv-iplab/action-anticipation-losses/>.

B.4 Encoder-Decoder LSTM (ED)

We implemented this model following the details specified in [8]. For fair comparison with respect to the proposed method, the model takes RGB and Flow features obtained using the representation functions considered for the RGB and Flow branches of our RU architecture. Differently from [8], we do not include a reinforcement learning term in the loss as our aim is not to distinguish the action from the background as early as possible as proposed in [8]. The hidden state of the LSTMs is set to 2048 units. The model encodes representations for 16 steps, while decoding is carried out for 8 steps at a step-size of 0.25s. The architecture is trained in two stages. In the first stage, the model is trained to predict future representations. This stage is carried out for 100 epochs. The training data for a given epoch is obtained by sampling 100 random sequences for each video in the case of EPIC-Kitchens and EGTEA Gaze+ and 10 random sequences for each video in the case of ActivityNet. The difference in the number of sampled sequences reflects the different natures of the datasets: EPIC-Kitchens and EGTEA Gaze+ tend to contain few long videos, while ActivityNet tends to contain many shorter videos. The sequences are re-sampled at each epoch, which reduces the risk of overfitting. In all cases, the models converged within 100 epochs. Similarly to the other approaches, we apply early stopping by selecting the model with the lowest validation loss. In the second stage, the architecture is fine-tuned to predict future representations and anticipate future actions for 100 epochs. In both stages we use the Adam optimizer and a learning rate of 0.001 as suggested by the authors of [8]. We also compare this method with a version of the approach in which the unsupervised pre-training stage is skipped (termed ED*). In this case, only the second stage of training is performed.

B.5 Feedback-Network LSTM (FN)

The method proposed in [51] has been implemented considering the best performing architecture among the ones investigated by the authors. This architecture comprises the “optional” LSTM layer and performs fusion by concatenation. The network uses our proposed video processing strategy. For fair comparison, we implemented the network as a two-stream architecture with two branches processing independently RGB and Flow features. The final predictions have been obtained with late fusion (equal weights for the two modalities). For fair comparisons, we used the representation functions of our architecture to obtain RGB and Flow features. The model has hidden layers of 1024 units, which in our experiments lead to improved results with respect to the 128 features proposed by the authors [51]. The model has been trained using the same parameters used in the proposed architecture.

B.6 Temporal Convolutional Network (TCN)

This baseline has been implemented using the code provided by the authors [72]. Similarly to FN, this method uses the our proposed video processing strategy. The network has 5 layers with kernels of size 7 in each layer. A dropout with $p = 0.8$ is applied to the input of each layer of the network. The model has been trained using cross entropy.

B.7 LSTM, RL & EL

These three methods have been implemented considering a single LSTM with the same parameters of our Rolling LSTM. Similarly to FN, the models have been trained as two-stream models with late fusion used to obtain final predictions (equal weights). The input RGB and Flow features have been computed using the representation functions considered in our architecture. The models have been trained with the same parameters used in the proposed architecture. LSTM has been trained using cross entropy, RL has been trained using the ranking loss on the detection score proposed in [7], whereas EL has been trained using the exponential anticipation loss proposed in [14].

C ADDITIONAL QUALITATIVE EXAMPLES

Fig. 11 reports qualitative results of three additional success action anticipation examples. For improved clarity, we report frames with and without optical flows for each example. In the top example, MATT assigns a small weight to the object branch as the contextual appearance features (i.e., RGB) are already enough to reliably anticipate the next actions. In the middle example object detection is fundamental to correctly anticipate “put down spoon”, as soon as the object is detected. The bottom example shows a complex scene with many objects. The ability to correctly recognize objects is fundamental to anticipate certain actions (i.e., “wash spoon”). The algorithm can anticipate “wash” well in advance. As soon as the spoon is detected ($\tau_a = 2s$), “wash spoon” is correctly anticipated. Note that, even if the spoon is not correctly detected at time $\tau_a = 0.5s$, “wash spoon” is still correctly anticipated.

Fig. 12 reports three failure examples. In the top example, the model fails to predict “adjust chair”, probably due to



Fig. 11. Success action anticipation example qualitative results (best seen on screen).

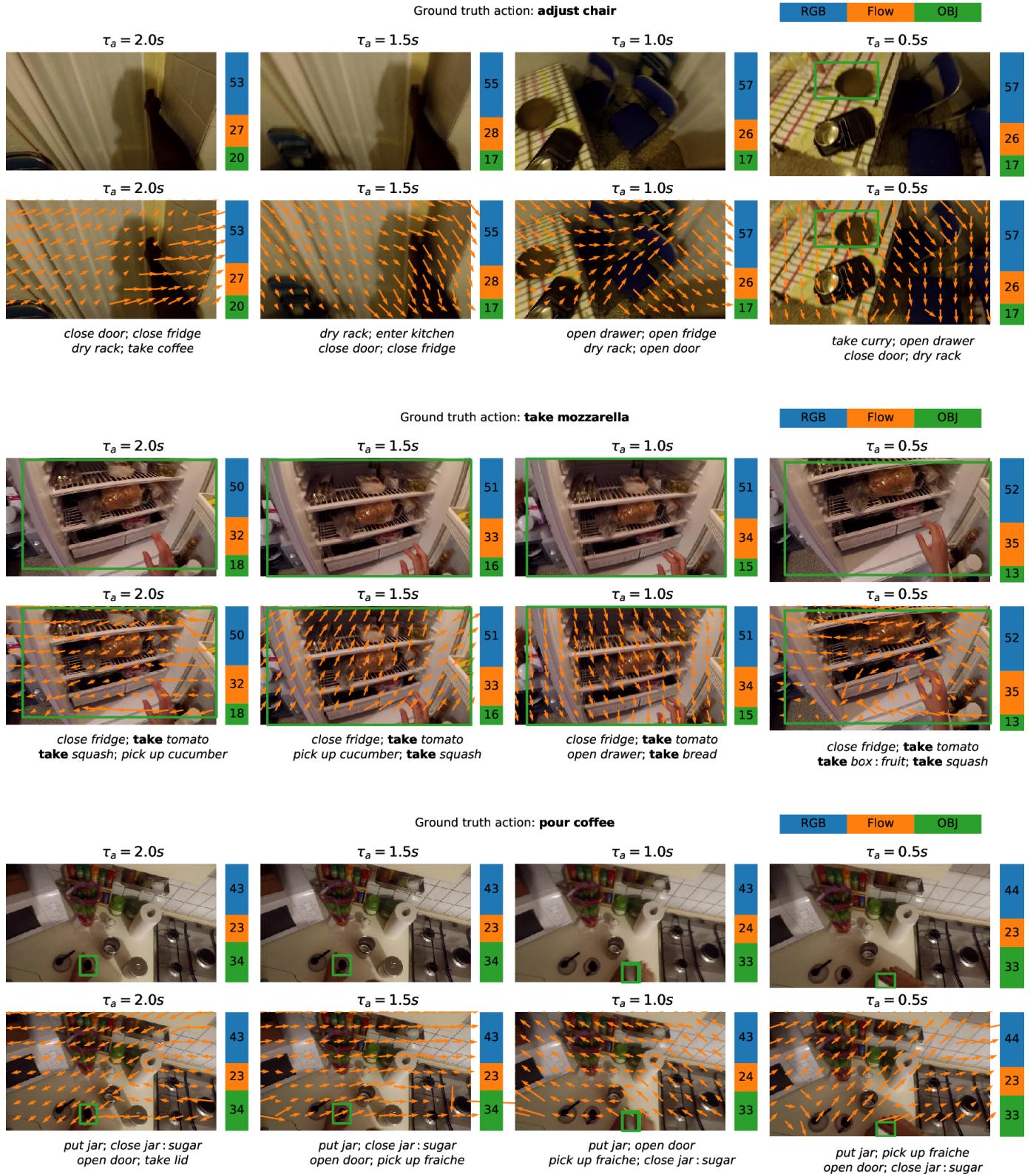


Fig. 12. Failure action anticipation example qualitative results (best seen on screen).

the inability of the object detector to identify the chair. Note that, when the object “pan” on the table is detected, “take curry” is wrongly anticipated. In the middle example, the algorithm successfully detects the fridge and tries to anticipate “close fridge” and some actions involving the “take” action, with wrong objects. This is probably due to the inability of the detector to detect “mozzarella”, which is not yet appearing in the scene. In the bottom example, the method tries to anticipate actions involving “jar”, as soon as “jar” is detected. This misleads the algorithm as the correct action is “pour coffee”.

The reader is referred to the videos available at <https://iplab.dmi.unict.it/rulstm/> for additional success and failure qualitative examples.

REFERENCES

- [1] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha, “Trafficpredict: Trajectory prediction for heterogeneous traffic-agents,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 6120–6127.
- [2] H. S. Koppula and A. Saxena, “Anticipating human activities using object affordances for reactive robotic response,” *Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 14–29, 2016.
- [3] B. Soran, A. Farhadi, and L. Shapiro, “Generating notifications for missing actions: Don’t forget to turn the lights off!” in *International Conference on Computer Vision*, 2015, pp. 4669–4677.
- [4] T. Kanade and M. Hebert, “First-person vision,” *Proceedings of the IEEE*, vol. 100, no. 8, pp. 2442–2453, 2012.
- [5] M. S. Aliakbarian, F. S. Saleh, M. Salzmann, B. Fernando, L. Petersson, and L. Andersson, “Encouraging lstms to anticipate actions very early,” in *International Conference on Computer Vision*, vol. 1, 2017, pp. 280–289.
- [6] R. De Geest, E. Gavves, A. Ghodrati, Z. Li, C. Snoek, and T. Tuytelaars, “Online action detection,” in *European Conference on Computer Vision*, 2016, pp. 269–284.
- [7] S. Ma, L. Sigal, and S. Sclaroff, “Learning activity progression in lstms for activity detection and early detection,” in *Computer Vision and Pattern Recognition*, 2016.
- [8] J. Gao, Z. Yang, and R. Nevatia, “RED: Reinforced encoder-decoder networks for action anticipation,” *British Machine Vision Conference*, 2017.
- [9] C. Vondrick, H. Pirsiavash, and A. Torralba, “Anticipating visual representations from unlabeled video,” in *Computer Vision and Pattern Recognition*, 2016, pp. 98–106.
- [10] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray, “Scaling egocentric vision: The epic-kitchens dataset,” in *European Conference on Computer Vision*, 2018, pp. 720–736.
- [11] A. Furnari, S. Battiato, and G. M. Farinella, “Leveraging uncertainty to rethink loss functions and evaluation measures for egocentric action anticipation,” in *European Conference on Computer Vision Workshops*, 2018.
- [12] Y. Abu Farha, A. Richard, and J. Gall, “When will you do what?: anticipating temporal occurrences of activities,” in *Computer Vision and Pattern Recognition*, 2018, pp. 5343–5352.
- [13] D.-A. Huang and K. M. Kitani, “Action-reaction: Forecasting the dynamics of human interaction,” in *European Conference on Computer Vision*, 2014, pp. 489–504.
- [14] A. Jain, A. Singh, H. S. Koppula, S. Soh, and A. Saxena, “Recurrent neural networks for driver activity anticipation via sensory-fusion architecture,” in *International Conference on Robotics and Automation*. IEEE, 2016, pp. 3118–3125.
- [15] M. S. Ryoo, T. J. Fuchs, L. Xia, J. K. Aggarwal, and L. Matthies, “Robot-centric activity prediction from first-person videos: What will they do to me?” in *International Conference on Human-Robot Interaction*, 2015, pp. 295–302.
- [16] Y. Li, M. Liu, and J. M. Rehg, “In the eye of beholder: Joint learning of gaze and actions in first person video,” in *European Conference on Computer Vision*, 2018.
- [17] F. Caba Heilbron, V. Escorcia, B. Chanthem, and J. Carlos Niebles, “Activitynet: A large-scale video benchmark for human activity understanding,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 961–970.
- [18] I. Laptev, “On space-time interest points,” *International Journal of Computer Vision*, vol. 64, no. 2-3, pp. 107–123, 2005.
- [19] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, “Learning realistic human actions from movies,” in *Computer Vision and Pattern Recognition*, 2008.
- [20] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, “Dense trajectories and motion boundary descriptors for action recognition,” *International Journal of Computer Vision*, vol. 103, no. 1, pp. 60–79, 2013.
- [21] H. Wang and C. Schmid, “Action recognition with improved trajectories,” in *International Conference on Computer Vision*, 2013, pp. 3551–3558.
- [22] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, “Large-scale video classification with convolutional neural networks,” in *Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.
- [23] K. Simonyan and A. Zisserman, “Two-stream convolutional networks for action recognition in videos,” in *Advances in Neural Information Processing Systems*, 2014, pp. 568–576.
- [24] C. Feichtenhofer, A. Pinz, and R. Wildes, “Spatiotemporal residual networks for video action recognition,” in *Advances in Neural Information Processing Systems*, 2016, pp. 3468–3476.
- [25] C. Feichtenhofer, A. Pinz, and R. P. Wildes, “Spatiotemporal multiplier networks for video action recognition,” in *Computer Vision and Pattern Recognition*, 2017, pp. 4768–4777.
- [26] C. Feichtenhofer, A. Pinz, and A. Zisserman, “Convolutional two-stream network fusion for video action recognition,” in *Computer Vision and Pattern Recognition*, 2016, pp. 1933–1941.
- [27] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. V. Gool, “Temporal segment networks: Towards good practices for deep action recognition,” in *European Conference on Computer Vision*, 2016, pp. 20–36.
- [28] B. Zhou, A. Andonian, A. Oliva, and A. Torralba, “Temporal relational reasoning in videos,” in *European Conference on Computer Vision*, 2018, pp. 803–818.
- [29] J. Lin, C. Gan, and S. Han, “Temporal shift module for efficient video understanding,” *arXiv preprint arXiv:1811.08383*, 2018.
- [30] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, “Learning spatiotemporal features with 3d convolutional networks,” in *International Conference on Computer Vision*, 2015, pp. 4489–4497.
- [31] J. Carreira and A. Zisserman, “Quo vadis, action recognition? a new model and the kinetics dataset,” in *Computer Vision and Pattern Recognition*, 2017, pp. 4724–4733.
- [32] K. Hara, H. Kataoka, and Y. Satoh, “Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?” in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6546–6555.
- [33] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
- [34] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, “A closer look at spatiotemporal convolutions for action recognition,” in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6450–6459.
- [35] E. H. Spriggs, F. De La Torre, and M. Hebert, “Temporal segmentation and activity classification from first-person sensing,” in *Computer Vision and Pattern Recognition Workshops*, 2009, pp. 17–24.
- [36] A. Fathi, A. Farhadi, and J. M. Rehg, “Understanding egocentric activities,” in *International Conference on Computer Vision*, 2011, pp. 407–414.
- [37] A. Fathi, Y. Li, and J. Rehg, “Learning to recognize daily actions using gaze,” in *European Conference on Computer Vision*, 2012, pp. 314–327.
- [38] H. Pirsiavash and D. Ramanan, “Detecting activities of daily living in first-person camera views,” in *Computer Vision and Pattern Recognition*, 2012, pp. 2847–2854.
- [39] Y. Li, Z. Ye, and J. M. Rehg, “Delving into egocentric actions,” in *Computer Vision and Pattern Recognition*, 2015, pp. 287–295.
- [40] M. S. Ryoo, B. Rothrock, and L. Matthies, “Pooled motion features for first-person videos,” in *Computer Vision and Pattern Recognition*, 2015, pp. 896–904.
- [41] M. Ma, H. Fan, and K. M. Kitani, “Going deeper into first-person activity recognition,” in *Computer Vision and Pattern Recognition*, 2016, pp. 1894–1903.
- [42] S. Singh, C. Arora, and C. Jawahar, “Trajectory aligned features for first person action recognition,” *Pattern Recognition*, vol. 62, pp. 45–55, 2017.

- [43] ——, "First person action recognition using deep learned descriptors," in *Computer Vision and Pattern Recognition*, 2016, pp. 2620–2628.
- [44] S. Sudhakaran, S. Escalera, and O. Lanz, "Lsta: Long short-term attention for egocentric action recognition," in *Computer Vision and Pattern Recognition*, 2018.
- [45] S. Sudhakaran and O. Lanz, "Attention is all we need: Nailing down object-centric attention for egocentric activity recognition," *British Machine Vision Conference*, 2018.
- [46] M. S. Ryoo, "Human activity prediction: Early recognition of ongoing activities from streaming videos," in *International Conference on Computer Vision*, 2011, pp. 1036–1043.
- [47] Y. Cao, D. Barrett, A. Barbu, S. Narayanaswamy, H. Yu, A. Michaux, Y. Lin, S. Dickinson, J. Mark Siskind, and S. Wang, "Recognize human activities from partially observed videos," in *Computer Vision and Pattern Recognition*, 2013, pp. 2658–2665.
- [48] M. Hoai and F. D. la Torre, "Max-margin early event detectors," *International Journal of Computer Vision*, vol. 107, no. 2, pp. 191–202, 2014.
- [49] D. Huang, S. Yao, Y. Wang, and F. De La Torre, "Sequential max-margin event detectors," in *European conference on computer vision*, 2014, pp. 410–424.
- [50] F. Becattini, T. Uricchio, L. Seidenari, A. Del Bimbo, and L. Ballan, "Am I done? predicting action progress in videos," *British Machine Vision Conference*, 2017.
- [51] R. D. Geest and T. Tuytelaars, "Modeling temporal structure with lstm for online action detection," in *Winter Conference on Applications in Computer Vision*, 2018.
- [52] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert, "Activity forecasting," in *European Conference on Computer Vision*. Springer, 2012, pp. 201–214.
- [53] T. Lan, T.-C. Chen, and S. Savarese, "A hierarchical representation for future action prediction," in *European Conference on Computer Vision*, 2014, pp. 689–704.
- [54] A. Jain, H. S. Koppula, B. Raghavan, S. Soh, and A. Saxena, "Car that knows before you do: Anticipating maneuvers via learning temporal driving models," in *International Conference on Computer Vision*, 2015, pp. 3182–3190.
- [55] P. Felsen, P. Agrawal, and J. Malik, "What will happen next? forecasting player moves in sports videos," in *International Conference on Computer Vision*, 2017, pp. 3342–3351.
- [56] T. Mahmud, M. Hasan, and A. K. Roy-Chowdhury, "Joint prediction of activity labels and starting times in untrimmed videos," in *International Conference on Computer Vision*, 2017, pp. 5773–5782.
- [57] K.-H. Zeng, W. B. Shen, D.-A. Huang, M. Sun, and J. Carlos Niebles, "Visual forecasting by imitating dynamics in natural sequences," in *International Conference on Computer Vision*, 2017, pp. 2999–3008.
- [58] Y. Zhou and T. L. Berg, "Temporal perception and prediction in ego-centric video," in *International Conference on Computer Vision*, 2016, pp. 4498–4506.
- [59] H. S. Park, J.-J. Hwang, Y. Niu, and J. Shi, "Egocentric future localization," in *Computer Vision and Pattern Recognition*, 2016, pp. 4697–4705.
- [60] M. Zhang, K. T. Ma, J. H. Lim, Q. Zhao, and J. Feng, "Deep future gaze: Gaze anticipation on egocentric videos using adversarial networks," in *Computer Vision and Pattern Recognition*, 2017, pp. 3539–3548.
- [61] A. Furnari, S. Battiato, K. Grauman, and G. M. Farinella, "Next-active-object prediction from egocentric videos," *Journal of Visual Communication and Image Representation*, vol. 49, pp. 401–411, 2017.
- [62] C. Fan, J. Lee, and M. S. Ryoo, "Forecasting hand and object locations in future frames," *CoRR*, vol. abs/1705.07328, 2017. [Online]. Available: <http://arxiv.org/abs/1705.07328>
- [63] N. Rhinehart and K. M. Kitani, "First-person activity forecasting with online inverse reinforcement learning," in *International Conference on Computer Vision*, 2017.
- [64] I. Sutskever, O. Vinyals, and Q. Le, "Sequence to sequence learning with neural networks," *Advances in NIPS*, 2014.
- [65] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [66] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000.
- [67] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *International Conference on Learning Representations*, 2015.
- [68] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*, 2015, pp. 2048–2057.
- [69] O. Mees, A. Eitel, and W. Burgard, "Choosing smartly: Adaptive multimodal fusion for object detection in changing environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Deajeon, South Korea, Oct. 2016. [Online]. Available: <http://ais.informatik.uni-freiburg.de/publications/papers/mees16iros.pdf>
- [70] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.
- [71] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime tv-l1 optical flow," in *Joint Pattern Recognition Symposium*, 2007, pp. 214–223.
- [72] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv:1803.01271*, 2018.
- [73] A. Miech, I. Laptev, J. Sivic, H. Wang, L. Torresani, and D. Tran, "Leveraging the present to anticipate the future in videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.
- [74] J. Lin, C. Gan, and S. Han, "Tsm: Temporal shift module for efficient video understanding," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7083–7093.
- [75] E. Kazakos, A. Nagrani, A. Zisserman, and D. Damen, "Epic-fusion: Audio-visual temporal binding for egocentric action recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 5492–5501.
- [76] D. Ghadiyaram, D. Tran, and D. Mahajan, "Large-scale weakly-supervised pre-training for video action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 046–12 055.
- [77] C.-Y. Wu, C. Feichtenhofer, H. Fan, K. He, P. Krahenbuhl, and R. Girshick, "Long-term feature banks for detailed video understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 284–293.
- [78] P. Zhang, J. Xue, C. Lan, W. Zeng, Z. Gao, and N. Zheng, "Adding attentiveness to the neurons in recurrent neural networks," in *European Conference on Computer Vision*, 2018, pp. 135–151.
- [79] R. Girshick, "Fast R-CNN," in *International Conference on Computer Vision*, 2015, pp. 1440–1448.
- [80] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He, "Detectron," <https://github.com/facebookresearch/detectron>, 2018.
- [81] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019, pp. 8024–8035.