Modeling Long-Term Interactions to Enhance Action Recognition

Alejandro Cartas
University of Barcelona
alejandro.cartas@ub.edu

Petia Radeva
University of Barcelona
petia.ivanova@ub.edu

Mariella Dimiccoli
Institut de Robòtica i Informàtica Industrial (CSIC-UPC)
mdimiccoli@iri.upc.edu

***Abstract*—In this paper, we propose a new approach to understand actions in egocentric videos that exploits the semantics of object interactions at both frame and temporal levels. At the frame level, we use a region-based approach that takes as input a primary region roughly corresponding to the user hands and a set of secondary regions potentially corresponding to the interacting objects and calculates the action score through a CNN formulation. This information is then fed to a Hierarchical Long Short-Term Memory Network (HLSTM) that captures temporal dependencies between actions within and across shots. Ablation studies thoroughly validate the proposed approach, showing in particular that both levels of the HLSTM architecture contribute to performance improvement. Furthermore, quantitative comparisons show that the proposed approach outperforms the state-of-the-art in terms of action recognition on standard benchmarks, without relying on motion information.**

I. Introduction

Activity recognition has been a very active research area in computer vision during the last decade [1]. In recent years, progress in wearable camera technology has offered the opportunity to capture naturally-occurring activities from a first-person point of view. The egocentric paradigm is particularly beneficial for analyzing activities involving object manipulations [2], commonly referred to as *human-object interactions*. Following the egocentric literature on the latter [3], [4], the term *action* represents a short-term human-object interaction such as *open ketchup* or *close fridge*, consisting of an action verb (i.e. open) and an interacting object (i.e. ketchup). The term *activity* indicates a sequence of actions such as *take bread, take cheese, open ketchup,* etc. performed with a specific goal, i.e. making a sandwich. Most works about human-to-object interaction model actions through egocentric features such as hand location and pose [5], [3], head motion and gaze [6], manipulated objects [7], [8], [9], or as a combination of all them [4], [10], [11]. However, only a few of them [3], [9], [11] specifically aimed at reasoning about different semantic entities in images across space and/or time.

In the egocentric setting, the temporal structure context has been used for temporal segmentation [14], story-telling [15], [16], and recognizing human interaction activities [17]. Only a few works have been focused on exploiting the temporal order of actions. [3] modeled contextual relations between actions and activities and the ordering between adjacent actions. In [9], the spatio-temporal reasoning is performed on an object-level by relying on state-of-the-art object segmentation networks that allow tracking the object appearance over time.

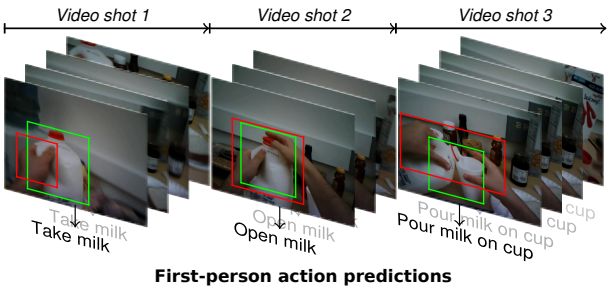

Figure 1. Our model takes as input for each frame a primary region (user hands, red boxes) and a set of secondary regions (object proposals, green boxes). These regions are used to make action predictions at frame level, that are aggregated firstly at shot level and then across shots.

In this work, we present a new egocentric action recognition approach that is able to reason about relevant image regions at the frame level, and also to understand action dynamics at the temporal level. At the frame level, we model first-person actions as spatially structured regions in a deformable star configuration, consisting of primary and secondary regions corresponding to the user hands and to the objects being potentially interacted with, respectively (see Fig. 1). We adapt the CNN framework proposed in [18] to train jointly the egocentric action-specific models and the feature maps. At the temporal level, we model first-person activities as sequences of actions and we propose a Hierarchical LSTM architecture, whose first layer captures the temporal evolution of contextual features over time within shots and the second one captures long-term temporal dependencies between actions across shots.

We evaluated the proposed approach on the GTEA, GAZE, and GAZE+ benchmarks. Quantitative comparisons to five state-of-the-art methods demonstrated both the effectiveness and the potential of our approach, which is able to outperform current methods even without relying on motion information. Unlike most available architectures, where motion is processed through a dedicated stream, our approach explicitly models the dynamics of human-object interaction at the temporal level.

Our experiments demonstrate that effectively exploiting semantics can have a major impact on performance than exploiting modeling motion. Furthermore, with this work, we provide useful insights on (1) the role of spatial reasoning at image level and (2) the role of reasoning over the temporal

Table I
STATE OF THE ART EGOCENTRIC OBJECT-INTERACTION RECOGNITION METHODS AND THEIR CHARACTERISTICS

Methods	Hands	Objects	Motion	Gaze	Spatio-temporal feature modelling	Spatio-temporal reasoning	Spatial reasoning on hand-objects	Temporal reasoning on actions
Fathi et al. (2011) [3]	✓	✓	✓		✓			✓
Fathi et al. (2012)[12]	✓	✓	✓	✓	✓			
Pirsiavash & Ramanan (2012)[8]		✓	✓		✓			
McCandless & Grauman (2013)[7]	✓	✓	✓		✓			
Bambach et al. (2015) [5]	✓							
Li et al. (2015) [6]	✓	✓	✓	✓				
Ma et al. [4]	✓	✓	✓		✓			
Singh et al. [13]	✓	✓	✓		✓			
Sudhakaran & Lanz (2018)[11]		✓	✓			✓		
Baradel et al. (2018)[9]		✓	✓			✓		
Ours	✓	✓					✓	✓

structure of human-object interactions.

II. RELATED WORK

a) Third-person action recognition from videos: Comprehensive surveys that cover the pre-deep learning era, can be found in [1], [19]. In recent years, there has been a proliferation of approaches based on deep learning architectures. Most of these works [20], [21], [22], [23] focus on atomic actions such as *brushing hairs*, *shake hands*, *kiss*, etc., instead of a sequence of actions aiming at performing a given activity. Typical neural network architectures for action recognition include ConvNet+LSTM [21], [22], two-stream convolutional networks [24], [23], [20], 3D ConvNet [25] and Long Temporal Convolutions [26]. In ConvNet-LSTM architecture, LSTM cells are typically added at the end of a CNN. Furthermore, the network input is a sequence of frames sampled from a video. In two-stream convolutional networks, one of the streams captures the spatial information while the other carries the temporal information. 3D ConvNets are a natural extension of 2D CNNs to video that allows us to capture invariance to translation in time. Finally, in LTC 3D, ConvNets are further extended to much longer temporal convolutions that enable action representation at their full-scale [26].

b) First-person action recognition from videos: Early works were focused on modeling spatio-temporal features. [3] modeled activities as a sequence of actions and each action as a spatio-temporal relationship between the hands and the objects involved in it. This work was further extended to include gaze features [12]. [8] introduced a temporal pyramid approach to encode spatio-temporal features along with detected active objects.[7] proposed a randomized spatio-temporal pyramids framework able to model the frequency with which each object category appears in particular space-time regions.

More recently, several end-to-end learning approaches have been proposed. [5] focused on hand location and pose, [6] on object features and gaze. Ma et al. [4] proposed a twin stream RGB-optical flow CNN architecture. Singh et al. [13] proposed four-stream neural network architecture. The first two streams capture temporal features from binary hand-arm masks. While the other two process spatial information from an RGB frame and motion taken from stacked optical flow encoded in a saliency map, respectively. [11] proposed an attention mechanism that enables the network to attend regions containing objects correlated with the activity under consideration. The output of the attention module serves as input for an LSTM network. [9] proposed a network able to reason about detected semantic object instances through space and time, building on the result of an object detector.

Differently to [9], [11] who fed to the recurrent units the features descriptions of relevant objects, we first perform action prediction by reasoning at the frame level, and then we work with action probabilities at the temporal level. Our temporal model is conceptually similar to [3], but contrary to them, we do not rely on domain knowledge about the activity being performed in the prediction process. In Table I, we list current state-of-the-art approaches to first-person action recognition along with ours and indicate the characteristics of each one in terms of features used and type of modeling.

c) Temporal context for action recognition: The role of the temporal context in action recognition has been the focus of works on action anticipation [27] and detection [28]. The former task consists in recognizing an action from a video shot as early as possible. The latter task determines the set of actions occurring in long videos and their time boundaries, but the actions do not necessarily belong to a single activity. The idea of using the temporal context for improving action recognition was explored in [29], [3]. Based on the observation that adding knowledge of the temporal structure of events helps to disambiguate low-level features on sequences of actions, [29] proposed a stochastic grammar parsing over the results of a Hidden Markov Model.

III. PROPOSED APPROACH

A. Frame-level modelling

To capture contextual cues of actions, we employed a star-structured region model with CNN features, adapting the R*CNN framework proposed in [18] for action recognition in still images captured by a third-person camera. This approach takes as input a manually selected primary region containing the person whose action has to be predicted, and a set of secondary regions that are supposed to capture contextual information such as objects the person is interacting with, the pose of the person, or what other people in the image

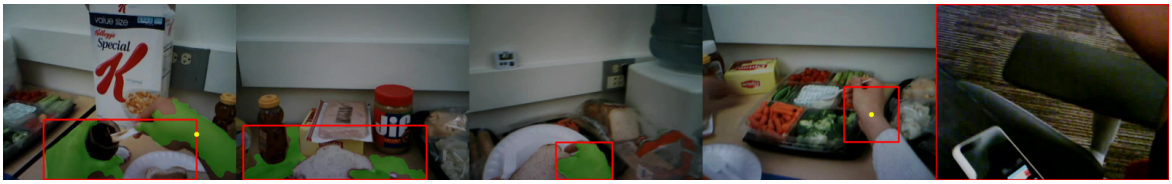

Figure 2. Examples of primary regions extraction taken from random frames of the GAZE dataset. The green-colored regions show detected skin pixels, the yellow dots are the wrist center locations, and the red bounding box shows the located primary region.

are doing. In our egocentric setting, the model is defined by a previously detected primary region corresponding to hands plus a set of parts regions corresponding to manipulated objects with associated CNN features.

a) Primary region detection: Since the human-to-object interactions are mainly done by manipulating the objects with the hands, we focus on the area covering the hand starting from the wrist to the tip of its middle finger. We define a primary region as the single area covered by one or both hands of the person. Fig. 1 depicts three examples where the primary regions detected with the proposed approach are shown in red bounding boxes. Our method determines the primary region based on skin regions and wrists located points. First, we perform skin pixel detection using the algorithm proposed in [30]. The training examples for this method are images containing visible arms and their respective skin masks. This method trains a random forest for each training example using a cluster from a set of local appearance features, such as different colorspaces (RGB, HSV, LAB) or spatial descriptors (SIFT [31], HOG [32], BRIEF [33]). The skin classification is done by globally combining the output of each random forest using a probabilistic model. In this work, the skin pixel detector was trained using only colorspaces.

The location of the wrists is performed using the Parts Affinity Fields model [34]. This CNN model has two branches that separately predict confidence maps for body part detection and part affinity fields (PAFs). The PAFs encode the orientation and location of limbs over the image. Both kinds of confidence maps are associated using a bipartite graph matching algorithm. The results are body poses for each person on the scene. Specifically, we only used the confidence maps generated for body part detection, since other body joints are not visible and their association cannot be made.

Once the skin regions and wrist location points were calculated, the primary region is determined as follows. If no skin regions and wrist location points are found, then the primary region is considered to be the full-frame (see the last column in the second row of Fig. 2). When no wrist points are located, then the complete skin region is determined to be the primary region. This case usually happens when a small part of the hand is seen (see the third column on the first row Fig. 2). In case, where no skin regions are found, then a fixed area near the wrists is considered as primary region (see the fourth column of the second row on Fig. 2). Finally, when both wrists points are located and skin regions are found, then wrists points determine the right, left, and lower sides of the primary region, while the skin determine its upper side (i.e. first two columns of the first row of Fig. 2).

b) Secondary regions prediction: In contrast to [18], where secondary regions are defined as those that overlap the primary region, our set of secondary regions coincides with the set of proposed regions. The reason is that often while performing an action as *take*, *poor water* etc., the hands are not in contact with the object during the full duration of the shot. Considering the performance results obtained in [35], we compute region proposals by using Multiscale Combinatorial Grouping (MCG) [36] instead of Selective Search [37] as in [18].

c) Action prediction: The score function implemented by the R*CNN architecture corresponds to those of a latent SVM formulation. In a latent SVM, each example x is scored by a function of the following form

$$f_w(x) = \max_{z \in Z(x)} w \cdot \phi(x, z),$$

where w is a vector of model parameters, z are latent values, and $\phi(x, z)$ is a feature vector.

In our particular case, given an image x and the primary region p in x containing the hands, the score for the action α is defined as

$$f(\alpha; x, p) = w_p^\alpha \cdot \phi(x, p) + \max_{z \in Z(p,x)} w_z^\alpha \cdot \phi(x, z) \qquad (1)$$

where z is the latent variable representing a secondary region corresponding to an object, $Z(p, x)$ is the set of candidates for the secondary region (object proposals), w_p^α and w_z^α are the model parameters.

The probability that given the hand p on the image x, and the action being performed α, is computed using softmax as in [18]. The feature vector $\phi(\cdot)$ and the vectors of model parameters, w_p^α and w_z^α, are learned jointly for all actions using a CNN in an end-to-end fashion (see Fig. 3).

B. Temporal modelling

Our architecture models sequences of human-object interactions using a hierarchical training structure. Variations between successive frames in a video encode useful information to make more accurate action predictions [22]. A standard way to account for such variations is to feed the output of a CNN to an LSTM in an end-to-end fashion. Consequently, at the bottom level of our hierarchy, we use a many-to-many

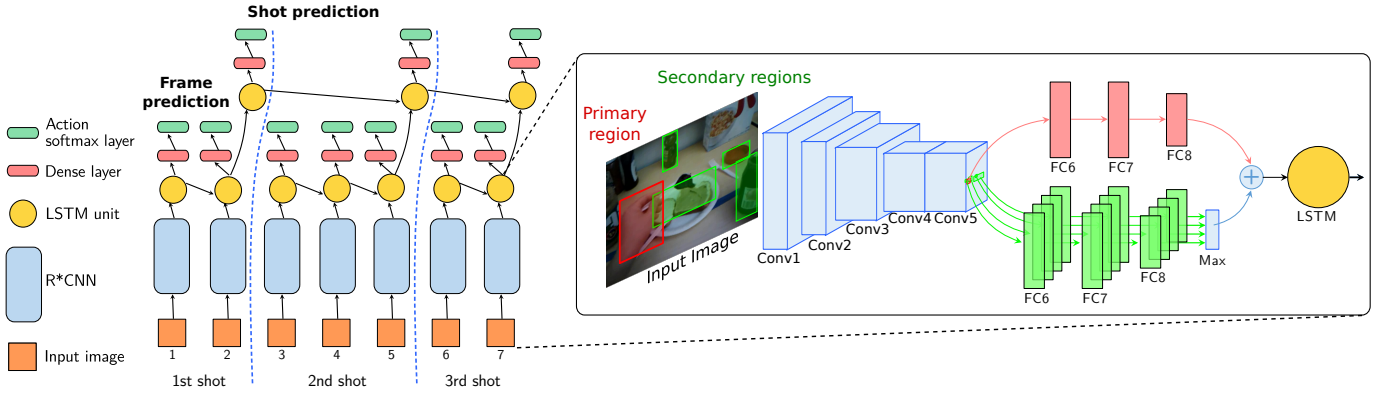

Figure 3. Overview of our proposed architecture. The left part shows the hierarchical action recognition pipeline across different video shots. The output of the last LSTM unit of each shot in the first level serves as input for an LSTM unit of the second level of the hierarchy. The right part shows the frame level processing, the primary (red), and secondary regions (green) boxes are previously estimated and passed as input to the network. The R*CNN architecture is applied to each frame within the video shot. The action prediction is measured at the frame level, as the output of the softmax of the first layer of LSTM.

LSTM to explicitly capture the temporal coherence of action probabilities within a video shot. Every LSTM unit at time step i takes as inputs the action probabilities returned by R*CNN for the i-th frame and the hidden state from the previous LSTM unit. Additionally, each LSTM output is connected to a dense layer and a softmax to output action probabilities that account for these of previous frames (see Fig. 3). The top-level of our hierarchy captures a long time temporal dependencies across *actions*. Specifically, the LSTM output of the last frame of each shot is fed to another LSTM unit, which also takes as input the hidden state of the LSTM units of the previous shot. The loss function of our complete HLSTM architecture is as follows:

$$\mathcal{L} = (1-\beta)\mathcal{L}_N + \beta\mathcal{L}_M \tag{2}$$

where β $(0 \leq \beta \leq 1)$ is a hyper-parameter used as a trade-off between two objectives and $\mathcal{L}_N$ and $\mathcal{L}_M$ are the cross-entropy loss defined at frame-level and shot-level, respectively. They are defined as follows:

$$\mathcal{L}_N = \sum_{i=1}^{N} L(y^i, \hat{y}^i),\ \mathcal{L}_M = \sum_{j=1}^{M} L(y^j, \hat{y}^j) \tag{3}$$

where N is the total number of frames, M is the total number of shots, y^i is the ground truth action at the $i-$th frame and $\hat{y}^i$ is the action prediction, accordingly. Similarly, y^j is the ground truth action at the $j-$th shot, and $\hat{y}^j$ is the action prediction.

C. Data augmentation

a) Visual data augmentation: This data augmentation process consisted in rotating the frames of all videos in a continuous manner while maintaining the temporal consistency between consecutive ones. This tries to mimic the head movements of the camera user. Given a video from the dataset containing N frames and a uniformly sampled random number $r \in \left[0, \frac{1}{2}\right]$, the rotation angle $\theta(n)$ for the $n-th$ frame is calculated using the following pair of equations:

$$\rho(n) = 2\pi\Big(\frac{C}{N}n + r\Big),\ \theta(n) = \Theta_{max}\sum_{i}^{M}\lambda_i \sin\gamma_i\rho(n) \tag{4}$$

where the constants C, γ_i, and $\sum_i^M \lambda_i = 1$ determine the sinusoidal behavior of θ within a period. The purpose of the function $\theta(n)$ is to generate continuous rotation angles for each sequence in the range $[-\Theta_{max}, \Theta_{max}]$.

After rotating a frame n by the angle θ, then its largest inscribed rectangle is cropped. This rectangle represents the augmented version of the $n-th$ frame. The width w_r and height h_r of this rectangle is calculated as follows. Let $s = \min(w,h)$ and $l = \max(w,h)$ denote the value of the shortest and longest side respectively.

If $\dfrac{s}{l} > 2\cdot|\sin\theta||\cos\theta|$

$$w_r = \frac{w|\cos\theta| - h|\sin\theta|}{\cos 2\theta},\ h_r = \frac{h|\cos\theta| - w|\sin(\theta)|}{\cos 2\theta} \tag{5}$$

else if $w > h$

$$w_r = \frac{s}{2|\sin\theta|},\quad h_r = \frac{s}{2|\cos\theta|} \tag{6}$$

otherwise

$$w_r = \frac{s}{2|\cos\theta|},\quad h_r = \frac{s}{2|\sin\theta|} \tag{7}$$

The primary region previously obtained is rotated with respect to its new center location. Afterward, a new bounding box is computed using the rotated corners. The secondary regions are calculated again for the augmented version of the $n-th$ frame by using the Multiscale Combinatorial Grouping (MCG) algorithm. This process duplicated our training data.

b) Temporal data augmentation: The goal of the temporal data augmentation was to extend the sequences of actions in a logical way. For instance, the original order of three actions for a sequence named *making a ham sandwich* could

Method	GTEA 61*	GTEA 71	GAZE 25*	GAZE 40*	GAZE+	Backbone CNN
Fathi and Rehg † [38]	39.7	-	-	-	-	-
Li et al. †[6]	66.8	62.1*	60.9	39.60	60.50	-
Two-Streams Network [20]	57.64	49.65	-	-	58.77	CNN-M-2048 [39]
Temporal Segments Network [40]	67.76	67.23	-	-	55.25	ResNet-34 [41]
Ma et al. [4]	75.8	73.24	62.40	43.42	**66.40**	CNN-M-2048 [39]
Sudhakaran and Lanz [11]	**77.59**	**77**	-	-	60.13	ResNet-34 [41]
CNN Baseline †	54.67	48.95	43.44	40.76	49.83	VGG-16[42]
Ours (frame level)	68.97	64.74	56.94	47.25	52.75	
Ours (1 level LSTM)	69.83	71.04	63.89	49.45	58.41	
Ours (Hierarchical LSTM)	70.69	72.95	**65.28**	**52.75**	59.96	

Table II

ACTION RECOGNITION ACCURACY ON THE GTEA, GAZE, AND GAZE+ DATASETS. (NOTE: * FIXED SPLIT, † ACCURACY MEASURED AT FRAME LEVEL).

be *take ham*, *take bread*, and *put ham on bread*, but it could be augmented by changing the order of the first two actions. The temporal sequences of the datasets are augmented in a two-step process. First, a meta-sequence is manually created for each video sequence in the dataset. This meta-sequence defines all the possible logical combinations that the sequence can have. Second, an alternate sequence is randomly generated from its meta-sequence during training. In order to create a meta sequence, groups of related actions are created along with their possible combinations. The groups can be combined using three different operations: they can be swapped/moved, skipped, or added to one another.

Method	S1	S2	S3	S4	Avg.
Frame level	62.6	68.42	58.46	69.47	64.74
1st level LSTM	73.28	73.68	70.0	67.18	71.04
HLSTM	**74.05**	**73.68**	**73.08**	**70.99**	**72.95**

Table III

CLASSIFICATION ACCURACY ON EACH SPLIT OF THE GTEA 71 DATASET.

Method	S1	S2	S3	S4	S5	S6	Avg.
Frame level	49.8	58.05	46.11	60.66	51.02	50.83	52.75
1st level LSTM	56.55	**65.77**	47.58	63.16	63.27	54.17	58.41
HLSTM	**58.53**	64.43	**51.37**	**64.54**	**64.63**	**56.25**	**59.96**

Table IV

CLASSIFICATION ACCURACY ON EACH SPLIT OF THE GAZE+ DATASET.

IV. EXPERIMENTS

A. Datasets and experimental protocol

a) Datasets: Our experiments were done using the GTEA [43], GTEA Gaze [12], and the GTEA GAZE+ [6] datasets. The GTEA dataset consists of 21 videos acquired by 4 different subjects while performing scripted activities such as making a *cheese sandwich*, or *instant coffee*. Each activity contains sequences of actions that are freely performed by the subject. The dataset includes two subsets of 71 and 61 actions. The GTEA Gaze dataset consists of 17 videos from 14 different subjects following unscripted food recipes. The dataset has two subsets of 25 and 40 actions, such as *open milk* or *take carrot*. Both action subsets have the same fixed training and test data splits as presented in [6]. The GAZE+ dataset has 37 videos of 7 activities recorded by 6 participants. The subjects were asked to perform some activity with the available objects. The total number of evaluated actions is 40. Some examples of the activities include *prepare a Greek salad* and *making cheese burger*, whereas some examples of actions are *put milk container* and *open fridge drawer*. For the former three datasets, the reported results were obtained by using the same splits as in previous works [3], [38], [6], [4], [11]. Specifically, when doing cross-validation, we left one person out of the datasets as the test user.

b) Evaluation: Since our model only uses the temporal structure of the entire video sequence during the training, the evaluation was carried out considering individual video shots and only the first level LSTM. Thus we are using the same given information for testing as previous works on first-person action recognition [20], [40], [4], [11]. Additionally, we considered that using the temporal structure information during training to be fair for two reasons. First, our method employs the same information provided in the dataset and even less, considering the GAZE information. Second, it is not stated in the cited literature what information should be used during training. The performance measurements were done using the frame average accuracy score per sequence. In addition, other methods that used the accuracy score at the frame level are indicated on the reported results (Table II).

B. Implementation details

a) Primary regions: Our method estimates the primary region by detecting skin regions and locating wrists points. We trained the skin pixel detector in [30] for each video of every dataset using skin masks. For datasets not including the skin mask, a different number of frames was uniformly sampled from each video and the skin contour was annotated using LabelMe [44]. In the case of the GTEA dataset, we used only the 814 skin masks provided. For the Gaze dataset, an average of 37 frames per video was annotated, giving a total of 625 skin masks. Lastly, 2,230 skin masks were annotated for the GAZE+ dataset, thus having an average of 60 annotated frames per video. The wrist points were detected by using the PAFs network [34]. This network was originally trained using the COCO 2016 keypoints challenge dataset [45]. Specifically, we only used the confidence maps generated for part detection and one scale for the wrist point detection for each dataset.

b) Secondary regions: We used the MCG method, pre-trained on the PASCAL 2012 and the BSDS500 datasets, for computing our object proposals. This method generated an average of 1,484 proposals per video frame.

	GTEA 61	GTEA 71	GAZE 25	GAZE 40	GAZE+
Avg. Levenshtein distance	2.36	2.57	15.12	19.40	50.40

Table V
SIMILARITY BETWEEN SEQUENCES OF ACTIONS IN VIDEOS FOR EACH DATASET. THE CLOSER THE VALUE TO ZERO THE MORE SIMILAR.

c) Data augmentation: The secondary regions were calculated again for the augmented version of frame n using the MCG algorithm. The effect of the rotation on the MCG was to make new secondary regions that do not overlap with the originals, this augmented the data used by the R*CNN architecture. This process roughly duplicated our training data.

d) Frame-level modelling: For each dataset, we fine-tuned the fully-connected layers of the R*CNN model on top of a VGG-16 network [42] pretrained on ImageNet. We used stochastic gradient descent (SGD) with a step learning policy and a step decay learning rate of $\gamma = 1e^{-1}$ for every 30K iterations. Depending on the dataset, the learning rate α was between $1e^{-4}$ and $2e^{-4}$ with momentum $\mu = 9e^{-1}$ and a batch size of 10. We randomly selected 10 regions from the set of proposal regions, without considering overlapping thresholds with respect to the primary region. The models were trained between 60K and 90K iterations.

e) Temporal modelling: We trained our hierarchical model by using SGD and the resulting weights of the previous frame-level modeling stage as initialization. The output of the softmax for each frame is provided as input to the hierarchical LSTM model. We first trained the model with $\beta = 0$ and then we used the resulting weights as initialization for the training with $\beta > 0$. The model with $\beta = 0$ was trained by using a step decay learning rate $\gamma = 1e^{-1}$ for every 30K iterations. Its learning rate α was between $1e^{-4}$ and $2e^{-4}$, depending on the dataset, a momentum $\mu = 9e^{-1}$, and a batch size of 6. The optimization was performed between 30K and 60K iterations. For $\beta > 0$, we trained the model between 5 and 8 epochs depending on the dataset by using a step decay learning rate $\gamma \approx 9.5e^{-1}$, and starting with a learning rate $\alpha = 5e^{-5}$, a momentum $\mu = 9e^{-1}$, and a batch size of 6. Since the second LSTM layer captures the sequence of actions across the videos, during the test all the video frames are fed sequentially to the network. The prediction at the frame level of the first layer of the LSTM is used to measure the performance. Thus, shot boundary information coming from the second LSTM layer is not needed for testing in our hierarchical model.

C. Ablation studies

a) Component Analysis: On the last rows of Table II we report performances obtained by using different modules of the proposed architecture. The module that performs spatial reasoning alone determines a consistent improvement with respect to the VGG baseline (13.79 %). The first LSTM layer is responsible for an increase of 8.19 % of per frame accuracy on average on all datasets. The second LSTM layer is responsible for a further increase of 2.69 %. It is worth stressing that the hierarchical model has a much smaller training set compared to the single-layer LSTM, which explains the relatively small improvement in performance. Moreover, the dissimilarity between the sequences of actions in the datasets explains the difference in performance improvement among them. We measure the similarity as the minimal cost of transforming one sequence into the other using the Levenshtein distance[46], as shown in Table V. The more similar the action sequences are, the higher the accuracy improvement.

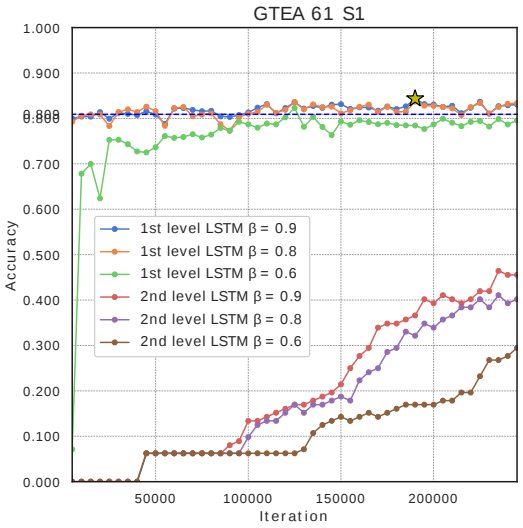

Figure 4. GTEA 61 test curves for different values of β. The horizontal dotted blue line indicates the test accuracy obtained for $\beta = 0$.

To better quantify the impact of our frame-level model, we compared its performance to those of the VGG-16 network [42] pretrained on ImageNet, since it serves as the basis for the R*CNN architecture. We observed considerable improvement in the performance of 24.70% on average. The effect of the hierarchical training over the first level LSTM can be seen using different values of β. First, we trained our model using a $\beta = 0$ (i.e. only the first LSTM level) until the optimization process converged. We then performed a grid search varying the learning rate α and the hyperparameter β on a single split of each dataset. In particular, we first found a suitable learning rate α, and then we looked for the best value of β in the set $\{0.5, 0.6, \ldots, 0.9\}$. For example, the effect of different values for β can be seen on Fig. 4. Moreover, the plot shows the resulting test curves for the first split of the GTEA 61 dataset. The classification accuracy for the first and second levels are measured per frame and shot, respectively.

b) Primary Region Importance: To gain an understanding of the importance of the primary region in our model, we trained the R*CNN architecture with both manually provided regions and with fixed regions covering the low central part of each image. For a single split of the GTEA 71, we achieved 66.11% accuracy when trained with ground truth primary region, 63.80% when trained with our method, 65.05% when trained using the skin as primary region, and 61.93% when trained with a fixed region. Although limited to a single split, these experiments suggest the importance of accurately

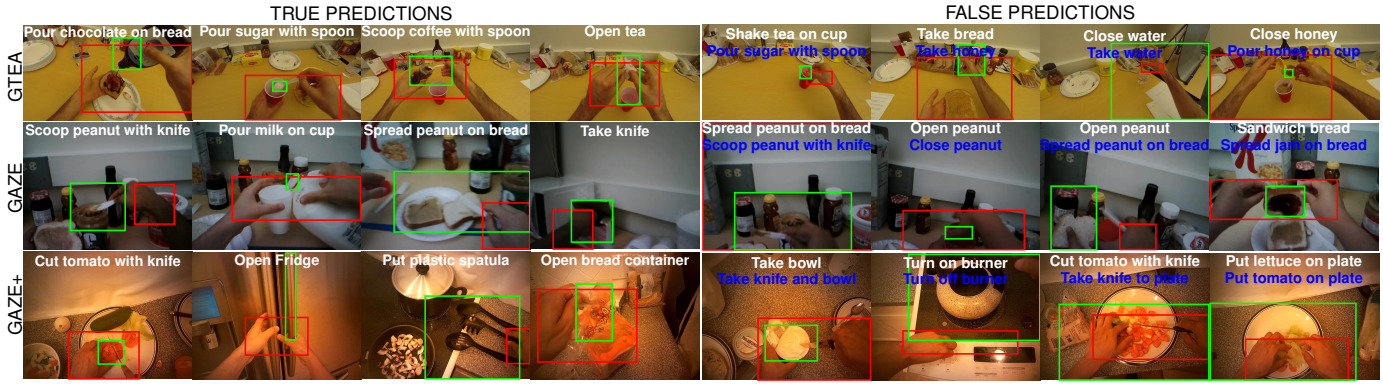

Figure 5. Examples of true and false predictions on all evaluated datasets. The true action category and its false prediction appears on top of the image on white and blue colors, respectively. The primary and secondary regions are the bounding boxes highlighted in red and green colors, accordingly.

Shot Evaluation	Sequence	Splits						Average
		Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6	
Avg. Prediction	Original	55.56	64.09	45.89	62.88	61.9	**56.67**	57.83
	Augmented	**56.55**	**65.77**	**47.58**	**63.16**	**63.27**	54.17	**58.41**
Weighted Avg. Prediction	Original	54.56	**63.76**	47.79	62.88	**62.59**	**57.08**	**58.11**
	Augmented	**56.94**	**63.76**	**49.47**	**63.16**	60.54	54.58	58.08

Table VI

CLASSIFICATION PERFORMANCE COMPARISON USING VISUAL DATA AUGMENTATION ON THE GAZE+ DATASET FOR 44 CATEGORIES.

Shot Evaluation	Sequence	Splits						Average
		Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6	
Avg. Prediction	Original	**58.53**	64.43	**51.37**	**64.54**	**64.63**	**56.25**	**59.96**
	Augmented	58.33	**65.44**	**51.37**	63.99	**64.63**	55	59.79
Weighted Avg. Prediction	Original	**58.13**	63.42	**52.21**	63.44	62.59	**57.08**	59.48
	Augmented	57.74	**64.09**	52	**65.10**	**63.26**	56.67	**59.81**

Table VII

CLASSIFICATION PERFORMANCE COMPARISON USING TEMPORAL DATA AUGMENTATION ON THE GAZE+ DATASET FOR 44 CATEGORIES.

detecting the primary region. We also report the results of our approach on each evaluated split on Table III and Table IV.

c) Visual Data Augmentation: We measured the classification performance of the visual data augmentation by comparing it against the original sequence. We considered all the splits from the GAZE+ dataset using the first level LSTM model. The accuracy was obtained using the normal and weighted average prediction for each video shot. The results of this experiment are shown in Table VI.

d) Temporal Data Augmentation: In order to understand the effect of temporally augmenting the data sequences, we measure the accuracy performance of our model on the GAZE+ dataset. We trained over all the splits of the dataset using the visually augmented frames. For each split, we trained using as initial weights the best resulting models from the first level LSTM. In addition to calculating the average prediction per shot, we also measured the linearly weighted average prediction. Experimental results are shown in Table VII.

D. Comparison with the state-of-the-art

We tested the proposed approach on five datasets from three well-known benchmarks (GTEA, Gaze, Gaze+) and performed comparisons to five state-of-the-art methods. On average, each competitive method has been tested on 3 datasets. We achieved a gain of 6.11%, 5%, on the GTEA Gaze, and GTEA Gaze+, respectively. On the GTEA dataset we achieved a similar performance to [4]. On this latter dataset, we observed that often the predictions at frame level within one shot were all incorrect so that the HLSTM is not able to improve the results for such shots. We stress that with respect to the state-of-the-art, our method does not rely on motion information, and shot boundary information is required solely during training. These results demonstrate the effectiveness of reasoning about the spatial arrangement of relevant regions at frame level on the one side, and how important is to exploit the temporal structure of videos on the other side. We observed that, in case of failure, very often either the action verb or the action object are correct. Specifically, a false verb prediction can often be attributed to the lack of motion modeling. We also provide some examples of qualitative results in Fig. 5, showing true and false predictions. We observed that, in case of failure, very often either the action verb or the action object are correct. Specifically, a false verb prediction can often be attributed to the lack of motion modeling.

V. CONCLUSIONS

This paper has introduced a novel approach for egocentric action recognition able to reason at a spatial level about semantically meaningful regions of the image and at the

temporal level about the sequence of actions being performed. The proposed neural network architecture consists of a CNN module, followed by an HLSTM able to capture temporal dependencies within and across shots. A detailed ablation analysis of the various components of our network validates the proposed architecture. The proposed model achieves state-of-the-art results on several standard benchmarks. In future work, we will improve the input to the HLSTM framework, we will exploit motion information and will test our method on the recently introduced EPIC-Kitchens dataset [47].

ACKNOWLEDGMENTS

This work was partially supported by CONACYT grant 366596, TIN2018-095232-B-C21, SGR-2017 1742, Nestore project of the European Commission Horizon 2020 programme (Grant N°769643), Validithi EIT Health program and CERCA Programme/Generalitat de Catalunya, MINECO/ERDF-EU through the program Ramon y Cajal, projects PID2019-110977GA-I00 and RED2018-102511-T. We thank the support of NVIDIA Corporation for hardware donation.

REFERENCES

[1] R. Poppe, "A survey on vision-based human action recognition," *Image and vision computing*, vol. 28, no. 6, pp. 976–990, 2010.
[2] M. Dimiccoli, "Computer vision for egocentric (first-person) vision," in *Computer Vision for Assistive Healthcare*. Elsevier, 2018, pp. 183–210.
[3] A. Fathi, A. Farhadi, and J. M. Rehg, "Understanding egocentric activities," in *ICCV*, 2011.
[4] M. Ma, H. Fan, and K. M. Kitani, "Going deeper into first-person activity recognition," in *CVPR*, 2016.
[5] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, "Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions," in *ICCV*, 2015.
[6] Y. Li, Z. Ye, and J. M. Rehg, "Delving into egocentric actions," in *CVPR*, 2015.
[7] T. McCandless and K. Grauman, "Object-centric spatio-temporal pyramids for egocentric activity recognition." in *BMVC*, 2013.
[8] H. Pirsiavash and D. Ramanan, "Detecting activities of daily living in first-person camera views," in *CVPR*. IEEE, 2012.
[9] F. Baradel, N. Neverova, C. Wolf, J. Mille, and G. Mori, "Object level visual reasoning in videos," in *ECCV*, June 2018.
[10] A. Behera, D. C. Hogg, and A. G. Cohn, "Egocentric activity monitoring and recovery," in *ACCV*. Springer, 2012, pp. 519–532.
[11] S. Sudhakaran and O. Lanz, "Attention is all we need: Nailing down object-centric attention for egocentric activity recognition," in *BMVC*, 2018.
[12] A. Fathi, Y. Li, and J. M. Rehg, "Learning to recognize daily actions using gaze," in *ECCV*. Springer, 2012.
[13] S. Singh, C. Arora, and C. V. Jawahar, "First person action recognition using deep learned descriptors," in *CVPR*, 2016.
[14] Y. Poleg, C. Arora, and S. Peleg, "Temporal segmentation of egocentric videos," in *CVPR*, 2014.
[15] A. Gupta, P. Srinivasan, J. Shi, and L. S. Davis, "Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos," in *CVPR*, 2009.
[16] B. Xiong, G. Kim, and L. Sigal, "Storyline representation of egocentric videos with an applications to story-based search," in *ICCV*, 2015.
[17] M. S. Ryoo and L. Matthies, "First-person activity recognition: Feature, temporal structure, and prediction," *International Journal of Computer Vision*, vol. 119, no. 3, pp. 307–328, 2016.
[18] G. Gkioxari, R. Girshick, and J. Malik, "Contextual action recognition with r*cnn," in *ICCV*, 2015.
[19] D. Weinland, R. Ronfard, and E. Boyer, "A survey of vision-based methods for action representation, segmentation and recognition," *Computer Vision and Image Understanding*, vol. 115, no. 2, pp. 224–241, 2011.
[20] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *NIPS*, 2014.
[21] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *CVPR*, 2015.
[22] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *CVPR*, 2015.
[23] X. Peng and C. Schmid, "Multi-region two-stream r-cnn for action detection," in *ECCV*, 2016.
[24] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *CVPR*, 2017.
[25] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013.
[26] G. Varol, I. Laptev, and C. Schmid, "Long-term temporal convolutions for action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 6, pp. 1510–1517, 2018.
[27] Y. Shi, B. Fernando, and R. Hartley, "Action anticipation with rbf kernelized feature mapping rnn," in *ECCV*, 2018.
[28] S. Buch, V. Escorcia, B. Ghanem, L. Fei-Fei, and J. C. Niebles, "End-to-end, single-stream temporal action detection in untrimmed videos," in *BMVC*, 2017.
[29] A. F. Bobick and Y. A. Ivanov, "Action recognition using probabilistic parsing," in *CVPR*. IEEE Computer Society, 1998, pp. 196–202.
[30] C. Li and K. M. Kitani, "Pixel-level hand detection in egocentric videos," in *CVPR*, 2013.
[31] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov 2004.
[32] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005.
[33] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *ECCV*, 2010.
[34] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *CVPR*, 2017.
[35] J. Hosang, R. Benenson, and B. Schiele, "How good are detection proposals, really?" in *BMVC*, 2014.
[36] J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping for image segmentation and object proposal generation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 1, pp. 128–140, 2017.
[37] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, 2013.
[38] A. Fathi and J. M. Rehg, "Modeling actions through state changes," in *CVPR*, 2013.
[39] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014.
[40] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Val Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *ECCV*, 2016.
[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
[42] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.
[43] A. Fathi, X. Ren, and J. M. Rehg, "Learning to recognize objects in egocentric activities," in *CVPR*, 2011.
[44] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "Labelme: A database and web-based tool for image annotation," *International Journal of Computer Vision*, vol. 77, no. 1, pp. 157–173, 2008.
[45] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*, 2014.
[46] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, vol. 10, no. 8, 1966, pp. 707–710.
[47] 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 *ECCV*, 2018.