Micro Actions and Deep Static Features for Activity Recognition

Sameera Ramasinghe, Jathushan Rajasegaran, Vinoj Jayasundara, Kanchana Ranasinghe,
Ranga Rodrigo and Ajith Pasqual
Department of Electronic and Telecommuication Engineering
University of Moratuwa
Sri Lanka
Email: {samramasinghe, brjathu, vinojjayasundara, kahnchana}@gmail.com, ranga@uom.lk, pasqual@ent.mrt.ac.lk

***Abstract*—A complex activity is a temporal composition of sub-events, and a sub-event typically consists of several low level micro-actions, such as body movement of different actors. Extracting these micro actions explicitly is beneficial for complex activity recognition due to actor selectivity, higher discriminative power, and motion clutter suppression. Moreover, considering both static and motion features is vital for activity recognition. However, optimally controlling the contribution from static and motion features still remains uninvestigated. In this work, we extract motion features at micro level, preserving the actor identity, to later obtain a high-level motion descriptor using a probabilistic model. Furthermore, we propose two novel schemas for combining static and motion features: Cholesky-transformation based and entropy-based. The former allows to control the contribution ratio precisely, while the latter obtains the optimal ratio mathematically. The ratio given by the entropy based method matches well with the experimental values obtained by the Choleksy transformation based method. This analysis also provides the ability to characterize a dataset, according to its richness in motion information. Finally, we study the effectiveness of modeling the temporal evolution of sub-event using an LSTM network. Experimental results demonstrate that the proposed technique outperforms state-of-the-art, when tested against two popular datasets.**

***Keywords*-Convolutional Neural Networks (CNN);Recurrent Neural Networks (RNN);Long Short-Term Memory (LSTM).**

I. Introduction

In video activity recognition, it is typical to focus on sub activities or sub events. For example, cooking may involve sub-events: cutting, turning on the cooker, and stirring. It may not always preserve this same lower order for the same action class. Instead, they may contain a higher-order temporal relationship among them, e.g., turning on the cooker and cutting may appear in reverse order in another video in the same class. On the other hand, there might be several actors responsible for performing such a sequence of micro-actions. A micro action refers to a low level action—such as body movements—done by an individual actor, within a small time frame of, say, 30 frames or less. A sub-event, which has a well-defined meaning in regular human interpretations, can comprise of several micro-actions. Encoding both the time evolution of sub-events and the set of micro-actions performed by the actors in the scene are important for classifying activities in video.

A complex activity typically comprises several sub-events. The sub-events in turn comprise micro-actions. Many existing approaches try to classify a video treating it as a single, high-level activity [25], [26], [20], [17]. As the action becomes complex, the behavior and the temporal evolution of its underlying sub-events become complicated.Therefore, this temporal pattern of sub-events is not easy to capture through a simple time series analysis. These patterns can only be identified by observing a large number of examples, using a system which has an infinite dynamic response. It is important to model this higher-order temporal progression, and capture temporal dynamics of these sub-events for better recognition of complex activities. Wang *et. al* [27] use temporal segment networks to model the long range temporal dynamics of a video, but it is not clear how the network captures the dynamics of each actor for modeling the dynamics.

When there are multiple actors in the scene, and if a collection of micro-actions are associated with each actor, there is actor selectivity. The level of discrimination achievable by focusing on micro-actions, in this context, is higher than that can be achieved using sub-events. The method of capturing sub-events to discriminate videos include linear dynamical systems [2], HMM [28], [29], and CRF-based methods [21]. Similarly, Rohrbach *et al.*[19], encode transition probabilities of a series of events statistically with a HMM model. However, these methods lack the power to attach an identity to what is tracked. If we track motion descriptors, like dense trajectories [25], the clustering simply lumps things that are moving in unison, e.g., a horse and a rider. However, if each actor has an associated identity, in the form of tracking micro-actions, a significant discriminative power becomes available. In addition, motion clutter will automatically be de-emphasized. Inview of this, taking micro-actions into consideration gives more discriminative power in video activity recognition, allows selections of actors, and de-emphasises motion noise.

Activity recognition requires static and motion features to simultaneously be considered. How to optimally combine or fuse these motion and static descriptors remains a challenge for three key reasons: both static and motion information provide clues regarding an action in a video, the contribution ratio from each domain affects the final recognition accuracy, and optimum contribution ratio of static and motion information may depend on the richness of the motion information. The combined descriptor should contain the essence of both domains, and should not have a negative influence on each other. Recently, there have been attempts to answer this question [20], [4], [17], [9], [22], [11]. However, existing methods lack the

978-1-5386-2839-3/17/$31.00 ©2017 IEEE

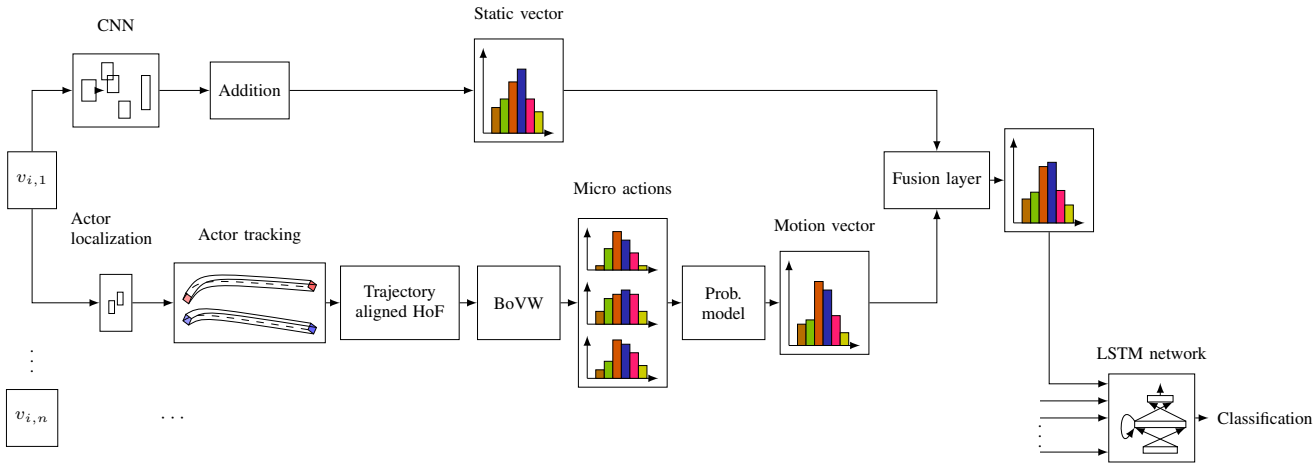

Fig. 1: **Overall methodology**. The whole process consists of eight major steps: (i) segmenting a video (ii) crafting static features, (iii) actor localization, (iv) actor tracking, (v) trajectory descriptor generation, (vi) mining micro actions (vii) fusing static and motion features, and (viii) capturing temporal evolution of sub events.

insight into how much this ratio affects the final accuracy, and have no control over this ratio.

In this paper, we present a video activity recognition method that involves CNN based static features, sub-events, and, more importantly, actors-associated micro actions. We address several problem areas: mining micro actions and combining them to obtain a higher level motion feature, crafting independent static and motion features, optimal fusion of the crafted features, and modeling temporal dynamics of sub-activities for high-level activity recognition. In order to examine the temporal evolution of sub-activities subsequently, we create video segments with constant overlapping durations. Afterwards, we create CNN-based-static and motion descriptors to represent each segment. For the motion descriptor, we first identify and localize candidate objects using Redmon *et al.* [18] and then use the method proposed in Possegger *et al.* [16] to track each object independently. In order to capture motion patterns of each actor, we use histogram oriented gradients (HOG) [3] on dense trajectories [25]. Then we apply a bag-of-words (BoW) method on the generated features to capture the distributional properties of micro actions. For generating the BoW, we use a k-means algorithm with a modified distance metric, presented in Aggarwal *et al.* [1], which performs better when clustering high demensional data. We then propose a probabilistic model to combine these micro actions and create high level, discriminative motion features. In order to model the temporal progression of sub events, we feed the fused vectors to a long short-term memory (LSTM) network. The LSTM network discovers the underlying temporal patterns of the sub events, and classifies high level actions. We compare it with a classifier which does not capture temporal dynamics to show that modeling temporal progression of sub events indeed gives better accuracy. Then, using a computationally efficient, yet powerful, mathematical model, we fuse static and motion feature vectors. We propose two such novel methods in this paper: based on Cholesky decomposition and entropy of motion and static vectors. Cholesky decomposition based method allows us to precisely control the contribution ratio of each domain, and entropy based method lets us obtain the optimum ratio mathematically. We show that these experimental and mathematical values match, and hence prove our hypothesis. The key contributions of this paper are as follows:

- Fused static and motion information based action classifier that models temporal evolution of sub-events.
- A micro-action based mechanism of tracking actors or objects independently, and probabilistically combining these learned dynamics leading to a motion descriptor.
- A method of combining static and motion features retaining the ability to govern the contribution of each feature type based on the Cholesky transformation, hence, establishing static and motion features are complementary.
- Establishing that the recognition accuracy is dependent on the ratio of contribution of static and motion features, and that the optimum ration depends on the dataset.
- Modeling the underlying temporal evolution of sub-events for complex activity recognition using an LSTM network. We experimentally prove that capturing these dynamics indeed benefits the final accuracy.

With the proposed technique we outperform the existing best results for the popular datasets UCF-11 [12] and Hollywood2 [14].

II. Methodology

Our activity classifier classifies video snippets based on their descriptors. In order to compute descriptors, initially, we segment a video into small snippets of 30 frames each. Then we carry out feature construction pipeline for each of these snippets, as shown in Fig. 1. We compute features for

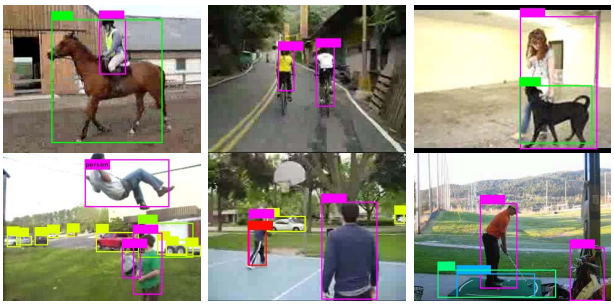

Fig. 2: **Initialization of candidate areas**. Top row: detection of multiple actors. Bottom row: cases where static objects are also detected as candidate areas. We disregard such areas based on the number of raw motion feature points generated in those areas.

each snippet that describe both motion and static domains. For extracting motion features, we localize and track objects independantly across frames. We initialize candidate areas, by applying Redmon *et al.* [18] in the first frame of each snippet.

Then, we track the candidate areas across the snippets by applying the state-of-the-art tracking mechanism provided in Posseger *et al.*[16]. Tracking each actor separately provides the ability to create motion features for each actor, and model micro actions independantly using a BoW method. We use a modified metric in k-means algorithm, to improve the clustering accuracy. In high dimensional space the data become sparse, and distance metrics can fail in the process of clustering, from an efficiency or effectiveness perspective, as proved in Aggarwal *et al.* [1]. They show that in high dimensional space, the concepts of proximity, distance, or nearest neighbor may not even be qualitatively meaningful. Therefore, we re-implement the k-means algorithm using the distance metric proposed in Aggarwal *et al.* [1]. We verify their claim by comparing the performance of several k-means implementations with different distance metrics. After modeling the micro actions, we combine the micro actions within the snippet using a probabilistic model to gain the final high level motion descriptor for that snippet.

For extracting static features we train a deep CNN on ImageNet. Then we apply this CNN on the frames of each video snippet to retrieve deep features—output vector from the final softmax layer of the CNN—from it. Then we use these features to create a static descriptor for the video segment. Afterwards, we combine these motion and static descriptors using one of the fusion models described in sub-section II-C.

The system can then represent a video as a vector time series, $C = [c_{t_0}, c_{t_1}, \ldots, c_{t_{n-1}}]$, where n is the number of segments. Then we apply an LSTM network on these features and exploit the dynamics of time evolution of the combined vector. Finally, we classify the dynamics of this time series and predict actions.

A. Motion Features

1) Actor Localization: In order to capture the temporal progression of the entire video, we first split the video in to snippets of 30 frames each. Then we apply the pre-trained detector presented in Redmon *et al.* [18] on the first frame of each segment and localize the objects and actors. It divides the image into regions and predicts bounding boxes and probabilities for each region. Then, these bounding boxes are weighted by the predicted probabilities to detect objects. In this step, we work under the assumption that the moving objects and actors do not change inside a duration of 30 frames. One drawback of this is the detector may detect still objects, which do not contribute to the motion activities in the video segment. It does not make sense to track still objects across the video and encode motion behaviours. We overcome this issue by discarding the candidate areas which do not contain significant motion information as discussed in Section II-A2. Fig. 2 shows some example candidate areas proposed by the detector.

TABLE I: Per-class accuracy comparison with state-of-the-art on UCF-11 (percent accuracy values).

Euclidean	Chi Square	F.M.(f=0.5)	F.M.(f=0.3)	F.M.(f=0.1)	F.M.(f=0.05)	F.M.(f=0.01)
87%	82%	67%	72%	81%	**92%**	88%

2) Tracking proposed candidate areas: After localizing probable moving actors and objects, in the first frame in each snippet, we track them using Posseger *et al.* [16] up to 30 frames. Then, we create dense trajectories [25] inside the bounding boxes, for each actor or object separately. Along these trajectories, we create 36-dimensional HOF features. Fig. 3 shows examples of created dense trajectories for each actor independently. We keep only the candidate areas with more than 50 HOF features within a snippet and discard the others. The value 50 was chosen by observation.

3) Modified K-Means for BoW: After creating HOF features, we choose $100,000$ feature points randomly from the pool of HOF features, which was generated from all the bounding boxes, in all the video segments, of all classes. This was done in order to reduce the time taken for k-means clustering to identify cluster heads needed for creating BoWs.

As mentioned in Section I, we modify the traditional k-means clustering algorithm, and replace the Euclidean distance by the the metric proposed in Aggarwal *et al.* [1], as the distance measuring function. Aggarwal *et al.* [1] proves, that the standard L_2 norm may be less effective in higher dimensions. Therefore, they propose a particular distance measuring function, which is a extension of L_k norm to the fractional distance metrics, for such cases as shown in Eq. 1. We verified this behavior using the following approach:we re-implemented the k-means algorithm using three different distance functions: Euclidean, Chi-squared distance, and the fractional metric, and evaluated it on the dataset provided in Fränti *et al.* [5]. Results are illustrated in Table I. The results show that the fractional metric, with $f = 0.05$, performs better. Therefore, we used the fractional metric as the distance measure function for clustering the HOF features.

$$\text{dist}_d^f(x, y) = \sum_{i=1}^{d} [((x^i - y^i))^f]^{\frac{1}{f}} \quad (1)$$

After clustering the randomly pooled $100,000$ HOF feature points, we obtain 1000 cluster heads. We choose the number of

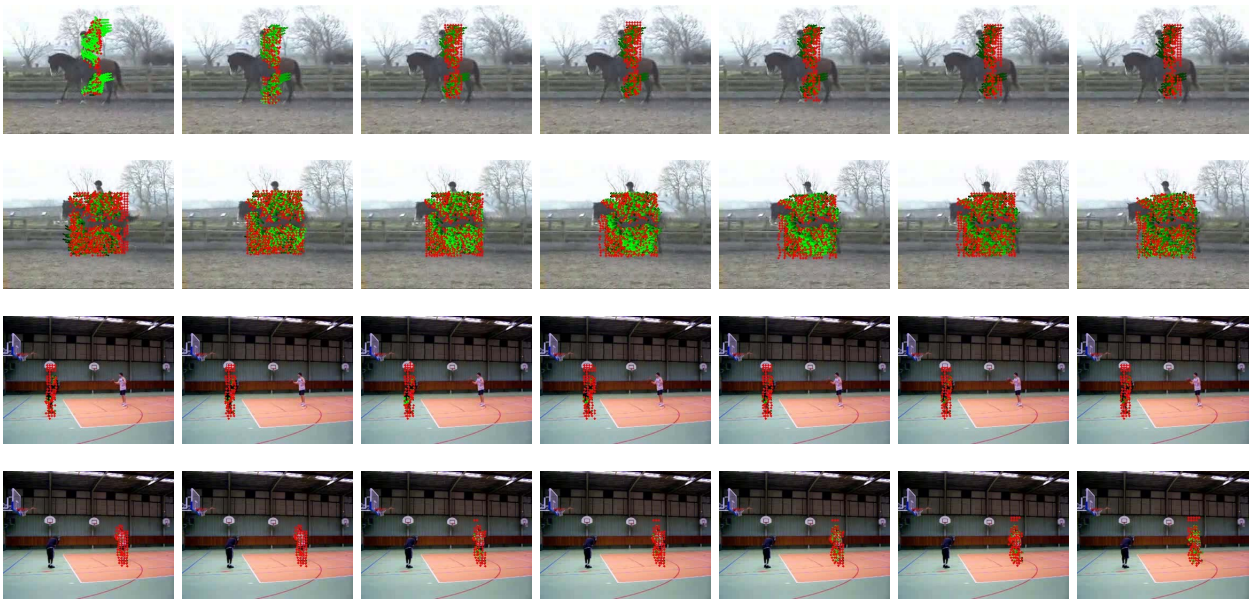

Fig. 3: **Modeling micro actions independently for each actor**. The top two rows illustrate the dense trajectories created for the rider and the horse. The bottom two rows illustrate the dense trajectories created for the two basketball players. The trajectories for the each actor were created in separate iterations, in order to simplify the visualization.

cluster heads as 1000, because the dimensions of final motion descriptors are needed to be the same as the static descriptors, which is explained in Section II-B. Then, for each individual object in each video segment $v_{i,n}$, a histogram is calculated as follows.

We calculate,

$$p_m = \underset{j \in J}{\operatorname{argmin}}_j(\text{dist}_{36}^{0.05}(T_j, h_{n,k,m})), J = \{1, 2, \ldots, 1000\} \tag{2}$$

for each $m \in \{1, 2, \ldots, z\}$, where $h_{n,k,m}$ is the m^{th} HOF vector of k^{th} actor of the n^{th} video segment, and T_j is the j^{th} cluster head. Then we increment the histogram values as $H_{n,k}(p_m) = H_{n,k}(p_m) + 1$, where $H_{n,k}(p_m)$ is the p_m^{th} value, $1 \leq p_m \leq 1000$, of histogram of the k^{th} actor of n^{th} video segment $v_{i,n}$. After calculating k histogram vectors $H_{n,k}$ representing every actor, we obtain $H_n = [H_1, H_2, \ldots, H_k]$ representing the set of micro actions in the n^{th} video segment.

4) High Level Actions from Micro Actions: The number of 1000 dimensional vectors is equal to the number of actors or moving objects in a particular video segment. In other words, each of these vectors represent a particular micro action done by a particular actor or object. These vectors should be merged in a mathematically meaningful way to obtain a descriptor for the high level action, consisting of these micro actions. We know,

$$H_{n,k} = P(M|A_{k,n}) \tag{3}$$

where M is the micro action and A_k is the k^{th} actor in n^{th} video segment. Therefore we can obtain high level motion descriptor M_{hn},

$$M_{hn} = P(M|A_{1,n}, A_{2,n}, ..., A_{l,n}) = \frac{\prod_{k=1}^{l} P(M|A_{k,n}) + \alpha}{\left[\frac{1}{k}\sum_{k=1}^{l} P(M|A_{k,n}\right]^{(k-1)} + \alpha k} \tag{4}$$

We empirically use the value $\alpha = 1$.

B. Static Features

In order to obtain static descriptors, we create a CNN of 1000 output classes and train it on the ImageNet dataset. The architecture is shown in Fig. 4. After training, we apply the trained model on each individual frame of each video segment. Then we average the output vectors of the CNN along indices and obtain a static descriptor s_i for each video segment v_i. Following the same procedure for every v_i, we develop a vector time series, $S = [s_1, s_2, \ldots, s_n]$, representing the static time evolution of the whole video.

C. Fusing of Static and Motion Features

This work depends on three factors; both static and motion information are vital for action recognition, but the final accuracy depends on the ratio of contribution of each domain, and the optimum contribution may depend on the richness of the motion information. We derive our fusion models addressing all these aspects. Although the static vector is extracted from the decision level of the CNN, we treat it as a feature level-vector which describes the probability distribution of each object appearing in a sub-event. Similarly, the motion vector represents the probability distribution of micro actions during

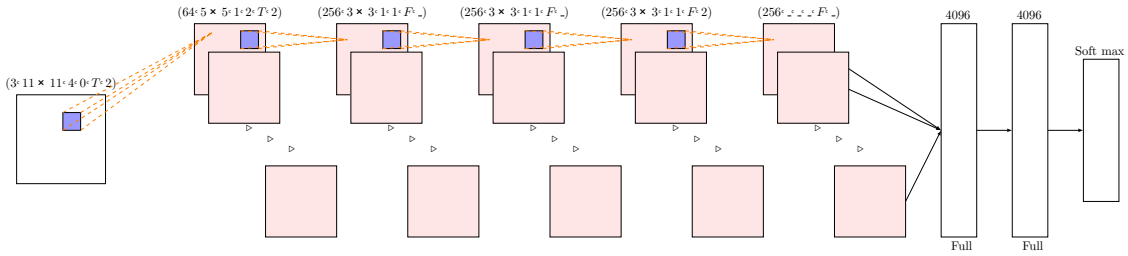

Fig. 4: CNN architecture used for generating static features. The CNN consists of five convolution layers, two fully connected layers, and one softmax layer. The details of each convolutional layer are provided on top of each layer according to the following format: (number of convolution layers × filter width × filter height, convolution stride, spatial padding, is Local Response Normalization added, max-pooling factor). Value above fully connected layers indicates the dimensionality of the layer. We use ReLu as the activation function.

a sub-event is hence treated as a feature level-vector. The three models we use to fuse the static and motion vectors are described next.

1) Cholesky Transformation Based Fusion: This derivation is based on the Cholesky transformation [6]. Let S and M be static and motion vectors, respectively. Cholesky transformation can be applied to the two vectors S and M with the correlation value ρ_1.

$$\begin{bmatrix} Y \\ Z \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \rho_1 & \sqrt{1-\rho_1^2} \end{bmatrix} \times \begin{bmatrix} S \\ M \end{bmatrix} \tag{5}$$

$$Y = S, Z = \rho_1 S + \sqrt{1-\rho_1^2} M \tag{6}$$

Similarly, the transformation can be applied to M and S with the correlation value ρ_2.

$$\begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \rho_2 & \sqrt{1-\rho_2^2} \end{bmatrix} \times \begin{bmatrix} M \\ S \end{bmatrix} \tag{7}$$

$$A = M, B = \rho_2 M + \sqrt{1-\rho_2^2} S \tag{8}$$

Cholesky transformation guarantees two properties: 1) The correlation between S and Z is ρ_1. 2) The correlation between M and B is ρ_2.

Therefore, if the values of ρ_1 and ρ_2 are chosen to obey $\rho_2 = \sqrt{1-\rho_1^2}$, it can be guaranteed that $Z = B$, $\forall S, M, \rho_1, \rho_2$. Hence, the resultant vector C can be obtained by $C = Z = B$

The correlation between C and S is ρ_1 and the correlation between C and M is ρ_2. Here S and M represent the static and the motion vectors whereas C represents the resultant vector. This derivation leads us to an important intuition: by choosing the value of ρ_1, we can choose the degree to which the static features and the motion features contribute, in deriving the resultant vector. In section 4, it is shown, how this property is used to explore, the optimal contribution of static and motion domain information for recognizing actions.

2) Entropy Based Fusion: In this method, we model each vector as a histogram, and fit a gaussian distribution for each. Using the histogram model, the mean and variance of each vector are calculated in order to fit the two vectors into Gaussian distributions. The joint Gaussian distribution is then computed based on this data.

$$G_{sm}(N) = \frac{1}{2\pi\sigma_m\sigma_s} e^{-\left[\frac{[N-\mu_s]^2}{2\sigma_s^2} + \frac{[N-\mu_m]^2}{2\sigma_m^2}\right]} \tag{9}$$

Then, we calculate the entropies of the both distributions using Eqn. 10. According to information theory, the amount of information contained in a histogram is proportional to the entropy of the distribution.

$$E(X) = -\sum(P(x_i) log_2(P(x_i))) \tag{10}$$

Then the axis of the joint probability can be scaled using the scaling matrix,

$$\text{Scaling matrix} = \begin{bmatrix} \frac{E(S)}{E(M)+E(S)} & 0 \\ 0 & \frac{E(M)}{E(M)+E(S)} \end{bmatrix}$$

Where $E(M)$ and $E(S)$ are entropy of motion and static vectors respectively. This enables us to give weighted contribution from each domain to the final vector, based on the information content. The parameters of the scaling matrix defines how much each each domain contributes to the evaluation line of the joint distribution. Therefore, the optimum ratio for combination of motion and static components of a given dataset can be mathematically evaluated. With regards to the UCF-11, 13% of motion vector and 87% of static vector constitute this parameter on average. For Hollywood2, it is 17% and 83%. This mathematical inference is further verified through the experiment results in Section III. Defining new parameters N_s and N_s as follows, we build a new distribution which is a scaled version of the joint Gaussian distribution obtained previously.

$$N_s = N\frac{E(S)}{E(M)+E(S)}, N_m = N\frac{E(M)}{E(M)+E(S)}$$

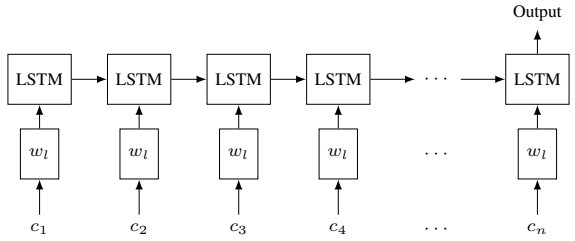

Fig. 5: The process of feeding fused vectors to the LSTM network. c_i indicates the fused vector representing the i_{th} video segment.

Finally, we define the following distribution representative of the combined vector.

$$G_{sm}(N) = \frac{e^{-\frac{1}{2(1-\rho^2)}\left[\frac{[N_s-\mu'_s]^2}{2\sigma'^2_s}+\frac{[N_m-\mu'_m]^2}{2\sigma'^2_m}-\frac{2\rho[N_s-\mu'_s][N_m-\mu'_m]}{\sigma'_s\sigma'_m}\right]}}{2\pi\sigma'_m\sigma'_s\sqrt{1-\rho^2}} \tag{11}$$

The corresponding mean and variance of this newly computed distribution are denoted by μ'_s, σ'_s and μ'_m, σ'_m for the static and motion vector respectively.

D. Capturing Temporal Evolution

In our work, we represent each video as n fixed-length segments (n differs for videos of different lengths) with overlapping frames. Each segment is represented with a fused vector c_t. Therefore, each video can be represented as a vector time series. We model and classify this time series by feeding the fused vectors to a LSTM network, using many to one apporach, as shown in Fig. 5. The network consists of an input layer, a 128 unit LSTM layer with 0.8 dropout, and a fully connected softmax output layer.

III. Experiments and Results

We use two benchmark datasets to evaluate our system: UCF-11 and Hollywood2. Our results outperform the state-of-the-art in both datasets. We demonstrate that the optimum contribution may vary depending on the richness of motion information, by varying the contribution of static and motion features for the calculation of the combined vector series. We also explore what is the optimum contribution is from each domain. Furthermore, we highlight the importance of considering time evolution of sub activities for identifying complex events, by comparing the performance of LSTM and Random Forest classifiers.

A. Contribution of Static and Motion Domains

We use the Cholesky Transformation based method for controlling the contribution of motion and static domains (subsection II-C1). Results for different contribution ratios for UCF-11 and Hollywood2 datasets, are shown in Table II and Table III. We use accuracy and mean average precision as performance metrics, for UCF-11 and Hollywood2, respectively. The overall optimum contribution ratio between static and motion vectors for both datasets stands at 80:20.

TABLE II: Per-class accuracy for different contribution of static and motion vectors for UCF-11. The vectors are fused using Cholesky method. Ratios are indicated in the format static:motion. Highest accuracy for UCF-11 is achieved using a 80:20 ratio between static and motion vectors (percent accuracy values).

Class	100:0	80:20	60:40	50:50	40:60	20:80	0:100
B_shooting	92.4	90.7	95.6	97.3	96.4	94.9	92.3
Biking	94.3	98.0	**98.3**	98.0	98.0	97.3	91.7
Diving	90.3	**99.6**	99.4	99.2	98.7	89.6	88.3
G_swinging	93.2	**98.1**	97.5	96.6	96.6	96.4	90.5
H_riding	94.0	**98.7**	98.1	97.5	96.7	96.1	84.2
S_juggling	92.4	**97.5**	96.8	95.4	95.3	94.6	88.3
Swinging	89.3	**98.3**	98.0	97.9	95.7	95.8	89.9
T_swinging	92.3	**99.2**	94.5	95.5	97.9	97.0	90.6
T_jumping	93.7	96.8	**98.9**	97.9	96.2	96.2	89.6
V_spiking	88.2	**98.3**	97.5	97.3	95.0	94.7	92.3
W_dog	90.2	**97.7**	96.6	96.4	96.4	96.2	92.2
Accuracy	91.8	**97.5**	97.4	97.2	96.6	96.3	90.0

TABLE III: AP for each class for different contribution of static and motion vectors to the fused vector for Hollywood2. Ratios are indicated in the format static:motion. Highest mAP for Hollywood2 is achieved using a 80:20 ratio between static and motion vectors (average precision values).

Class	100:0	80:20	60:40	50:50	40:60	20:80	0:100
AnswerPhone	52.3	**82.6**	51.3	44.4	39.7	39.7	34.5
DriveCar	64.6	**96.1**	53.2	47.5	43.4	39.7	36.2
Eat	50.0	**66.3**	55.6	55.2	53.2	48.1	45.0
FightPerson	72.2	**94.3**	82.2	76.4	68.9	53.4	44.2
GetOutCar	56.9	**83.4**	62.2	54.2	52.3	44.5	38.7
HandShake	42.2	74.9	**82.2**	70.7	59.4	42.3	39.2
HugPerson	49.9	**77.1**	64.3	62.6	57.2	50.9	40.6
Kiss	49.9	85.3	**86.4**	70.2	64.7	59.5	43.2
Run	60.2	78.2	**92.8**	84.2	81.5	66.4	59.5
SitDown	80.2	82.8	**90.5**	76.4	76.4	67.3	50.8
SitUp	58.7	82.0	**84.2**	76.2	65.9	48.5	42.1
StandUp	55.5	78.2	**93.4**	69.6	62.5	42.7	33.4
mAP	58.3	**81.8**	62.8	65.6	50.3	51.9	42.3

Distributions for the overall performance over different contribution levels, from static and motion domains, for datasets UCF-11 and Hollywood2 are shown in Table II and Table III. The performance change for different contribution percentages of motion and static domain is evident in these distributions. The optimum contribution is depends on the scenario, e.g., if motion patterns are indistinguishable across actions, static information plays a critical role for determining the action, and vise versa. In Table III it is evident that, for action classes which do not highly interact with external objects—such as, kiss, run, sitdown, situp, standup handshake—the optimum motion:static ratio is 40:60. For other action classes, which interact with objects, optimum motion:static ratio is 20:80. This highlights our hypothesis, that being able to control this contribution explicitly, is vital for an action recognition system.

Our mathematical derivation of optimum contribution ratio as 87:13 and 83:17 for UCF-11 and Hollywood2 datasets (Section II-C2) closely aligns with the experimental values of

TABLE IV: Comparison of our method with the state-of-the-art methods in the literature. Motion:static ratios are 20:80 for both UCF-11 and Hollywood2.

UCF-11		Hollywood2	
Liu *et al.*[12]	71.2	Vig *et al.*[24]	59.4
Ikizler-Cinbis *et al.*[7]	75.2	Jiang *et al.*[10]	59.5
Wang *et al.*[25]	84.2	Mathe *et al.*[15]	61.0
Ramasinghe *et al.*[17]	93.1	Jain *et al.*[8]	62.5
		Wang *et al.*[25]	58.3
		Wang *et al.*[26]	64.3
Our method(Cholesky)	**97.5**	Our method (Cholesky)	**81.8**
Our method(Entropy)	90.9	Our method (Entropy)	69.9

TABLE V: Per-class accuracy comparison with state-of-the-art on UCF-11 (percent accuracy values).

Class	Ours (Cholesky)	Ours (Entropy)	KLT[13]	Wang *et al.*[25]	Ikizler-Cinbis[7]	Ramasinghe *et al.*[17]
B_shooting	**90.7**	91.1	34.0	43.0	48.5	95.6
Biking	**98.0**	91.7	87.6	91.7	75.17	93.1
Diving	**99.6**	92.8	99.0	99.0	95.0	92.8
G_swinging	**98.1**	92.4	95.0	97.0	95.0	95.0
H_riding	**98.7**	89.2	76.0	85.0	73.0	94.3
S_juggling	**97.5**	89.6	65.0	76.0	53.0	87.8
S_winging	**98.3**	91.1	86.0	88.0	66.0	92.4
T_swinging	**99.2**	85.8	71.0	71.0	77.0	94.9
T_jumping	**96.8**	91.5	93.0	94.0	93.0	94.0
V_spiking	**98.3**	93.6	96.0	95.0	85.0	93.2
W_dog	**97.7**	91.7	76.4	87.0	66.7	91.4
Accuracy	**97.5**	90.9	79.0	84.2	75.2	93.1

80:20 (Table IIII and Table III), further verifying our results.

B. Comparison with the State-of-the-Art

We compare our results using a motion:static ratio of 20:80 with state of the art for UCF-11 and Hollywood2 datasets (IV). On UCF-11, we signicantly outperform the state-of-the-art by 4.4% while on Hollywood2 we outperform the state-of- the-art by 23.5%. We also compare per-action class results in Table V and Table VI. In UCF-11, our method excels in all the 11 classes, while in Hollywood2 we excel in 10 out of 12 classes.

C. Effectiveness of Capturing Time Evolution

A complex action is composed of multiple sub activities. The sub activities conform to a temporal pattern. We try to capture those patterns using an LSTM network, and measure their impact on the accuracy of classication. For comparison, we also directly feed the fused vectors to a random forest classier, which does not capture sequential dynamic patterns. It is evident from the results in Fig. 6 and Fig. 7, that the LSTM network signicantly outperforms the random forest classier for both datasets. In Hollywood2, the LSTM network wins by a 9.6% margin, while in UCF-11, the LSTM network wins by an 14.1% margin. Thus, we verify the significant impact of temporal patterns of sub activities on complex action classication.

TABLE VI: Per-class mAP comparison with state-of-the-art on Hollywood2 (average precision values).

Class	Ours (Cholesky)	Ours (Entropy)	KLT[13]	Wang et al.[25]	Ullah[23]
AnswerPhone	**82.6**	66.3	18.3	32.6	25.9
DriveCar	**96.1**	81.2	88.8	88.0	85.9
Eat	66.3	58.4	**73.4**	65.2	56.4
FightPerson	**94.3**	82.1	74.2	81.4	74.9
GetOutCar	**83.4**	66.7	47.9	52.7	44.0
HandShake	**74.9**	64.3	18.4	29.6	29.7
HugPerson	**77.1**	70.2	42.6	54.2	46.1
Kiss	**85.3**	74.2	65.0	65.8	55.0
Run	78.2	66.7	76.3	**82.1**	69.4
SitDown	**82.8**	72.1	59.0	62.5	58.9
SitUp	**82.0**	70.6	27.7	20.0	18.4
StandUp	**78.2**	66.3	63.4	65.2	57.4
mAP	**81.8**	69.9	54.6	58.3	51.8

Fig. 6: Accuracy comparison between Random Forest Classifier and LSTM for UCF-11 dataset. Motion:static ratio of 20:80 is used. Accuracy is significantly higher when the temporal dynamics of sub events are captured.

Fig. 7: mAP comparison for Random Forest Classifier and LSTM for Hollywood2 dataset. Motion:static ratio of 20:80 is used. mAP is significantly higher when the temporal dynamics of sub events are captured.

IV. Conclusion

In this paper we proposed a fused static and motion descriptor based action classifier that models temporal progression of sub-events. We use an actor-based tracking mechanism and extract micro actions using dense-trajectory aligned HOF features. Then we obtain a high-level motion descriptor by combining micro actions, with the aid of a probabilistic model. Our static descriptor is based on deep features.

We presented two novel methods—based on entropy and Cholesky transformation—to fuse static and motion vectors. Cholesky method provides the ability to control the contribution of each domain and the entropy method obtains the optimum ratio based on the information content. We showed that the experimental and and mathematically derived values agree. Furthermore, we showed that the optimum contribution of static and motion domains may vary depending on the richness of the motion information.

We modeled the temporal progression of sub-events using an LSTM network. Experimental results indicate that this is indeed beneficiary, compared to using models which do not capture temporal dynamics. Comparison of our work with multiple state-of-the-art algorithms, on the two popular data sets, UCF-11 and Hollywood2, show that our system performs better.

Acknowledgment

This research was supported by National Research Council of Sri Lanka, under grant No: 12-018.

References

[1] C. C. Aggarwal, A. Hinneburg, and D. A. Keim. On the surprising behavior of distance metrics in high dimensional space. In *International Conference on Database Theory*, pages 420–434. Springer, 2001. 2, 3

[2] S. Bhattacharya, M. Kalayeh, R. Sukthankar, and M. Shah. Recognition of complex events: Exploiting temporal dynamics between underlying concepts. pages 2235–2242, Columbus, OH, Jun 2014. 1

[3] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *European conference on computer vision*, pages 428–441. Springer, 2006. 2

[4] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. *Computing Research Repository (CoRR)*, abs/1604.06573, Apr 2016. 1

[5] P. Fränti, O. Virmajoki, and V. Hautamäki. Fast agglomerative clustering using a k-nearest neighbor graph. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(11):1875–1881, 2006. 3

[6] G. H. Golub and C. F. V. Loan. *Matrix Computations*. The Johns Hopkins Univeristy Press, 3rd edition edition, 1996. 5

[7] N. Ikizler-Cinbis and S. Sclaroff. Object, scene and actions: Combining multiple features for human action recognition. pages 494–507, Crete, GRC, Sep 2010. Springer. 7

[8] M. Jain, H. Jegou, and P. Bouthemy. Better exploiting motion for better action recognition. pages 2555–2562, Portland, OR, Jun 2013. 7

[9] S. Ji, W. Xu, M. Yang, and K. Yu. 3D convolutional neural networks for human action recognition. 35(1):221–231, Jan 2013. 1

[10] Y.-G. Jiang, Q. Dai, X. Xue, W. Liu, and C.-W. Ngo. Trajectory-based modeling of human actions with motion reference points. pages 425–438, Florence, ITA, Oct 2012. Springer. 7

[11] H.-J. Kim, J. Lee, and H.-S. Yang. Human action recognition using a modified convolutional neural network. In *Int. Symp. on Neural Networks*, pages 715–723, Nanjing, CHN, Jun 2007. 1

[12] J. Liu, J. Luo, and M. Shah. Recognizing realistic actions from videos in the wild. pages 1996–2003, Miami, FL, Jun 2009. 2, 7

[13] B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. pages 674–679, 1981. 7

[14] M. Marszalek, I. Laptev, and C. Schmid. Actions in context. pages 2929–2936, Miami, FL, Jun 2009. 2

[15] S. Mathe and C. Sminchisescu. Dynamic eye movement datasets and learnt saliency models for visual action recognition. pages 842–856. Springer, Florence, ITA, Oct 2012. 7

[16] H. Possegger, T. Mauthner, and H. Bischof. In defense of color-based model-free tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2113–2120, 2015. 2, 3

[17] S. Ramasinghe and R. Rodrigo. Action recognition by single stream convolutional neural networks: An approach using combined motion and static information. In *3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 101–105, Kuala Lumpur, MAS, Nov 2015. 1, 7

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016. 2, 3

[19] M. Rohrbach, M. Regneri, M. Andriluka, S. Amin, M. Pinkal, and B. Schiele. Script data for attribute-based recognition of composite activities. pages 144–157, Florence, ITA, Oct 2012. 1

[20] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. pages 568–576, Montreal, CAN, Dec 2014. 1

[21] Y. Song, L.-P. Morency, and R. Davis. Action recognition by hierarchical sequence summarization. pages 3562–3569, Portland, OR, Jun 2013. 1

[22] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3D convolutional networks. pages 4489–4497, Santiago, CHI, Dec 2015. 1

[23] M. M. Ullah, S. N. Parizi, and I. Laptev. Improving bag-of-features action recognition with non-local cues. volume 10, pages 95–1, Aberystwyth, GBR, Aug 2010. 7

[24] E. Vig, M. Dorr, and D. Cox. Space-variant descriptor sampling for action recognition based on saliency and eye movements. pages 84–97, Florence, ITA, Oct 2012. Springer. 7

[25] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. Colorado Springs, CO. 1, 2, 3, 7

[26] H. Wang and C. Schmid. Action recognition with improved trajectories. pages 3551–3558, Sydney, AUS, Nov 2013. 1, 7

[27] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision*, pages 20–36. Springer, 2016. 1

[28] Y. Wang and G. Mori. Hidden part models for human action recognition: Probabilistic versus max margin. 33(7):1310–1323, Jul 2011. 1

[29] D. Wu and L. Shao. Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition. pages 724–731, Columbus, OH, Jun 2014. 1