Micro Actions and Deep Static Features for Activity Recognition

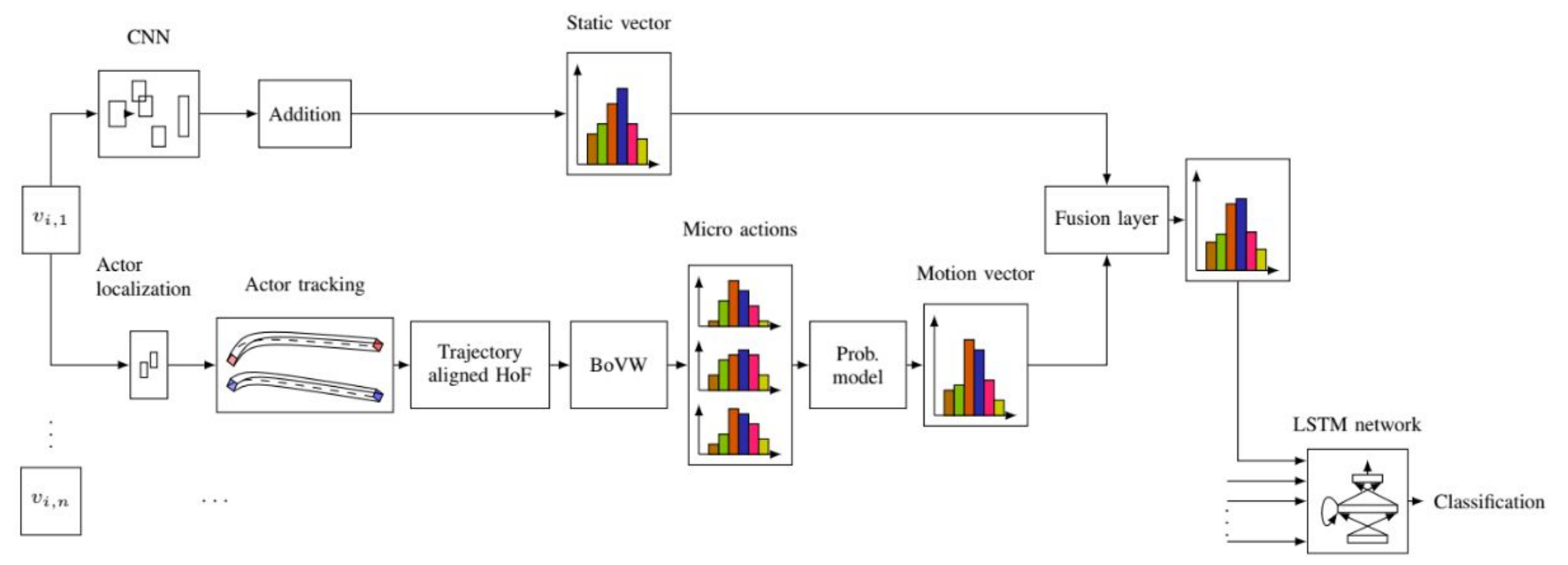
Sameera Ramasinghe, Jathushan Rajasegaran, Vinoj Jayasundara, Kanchana Ranasinghe,Ranga Rodrigo and Ajith Pasqual

Presented at DICTA 2017

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 schemes for combining static and motion features. This analysis also provides the ability to characterize a dataset, according to its richness in motion information.

Methodology



Our Methodology contains three stages of pipeline. First, we extract motion and static features from the video. Second, we fuse both feature vectors into a single vector, and finally we feed it through a pre-trained RNN to classify the video.

Feature Extraction

In the motion feature extraction, we segment each video by 30 frames. Then for each segment, we find objects in the first frame and track those objects throughout the segment using dense trajectories for each actor or object separately, along with 36-dimensional HOF features. Then we Keep only the candidate areas with more than 50 (chosen by observation) HOF features within a snippet and discard the others.

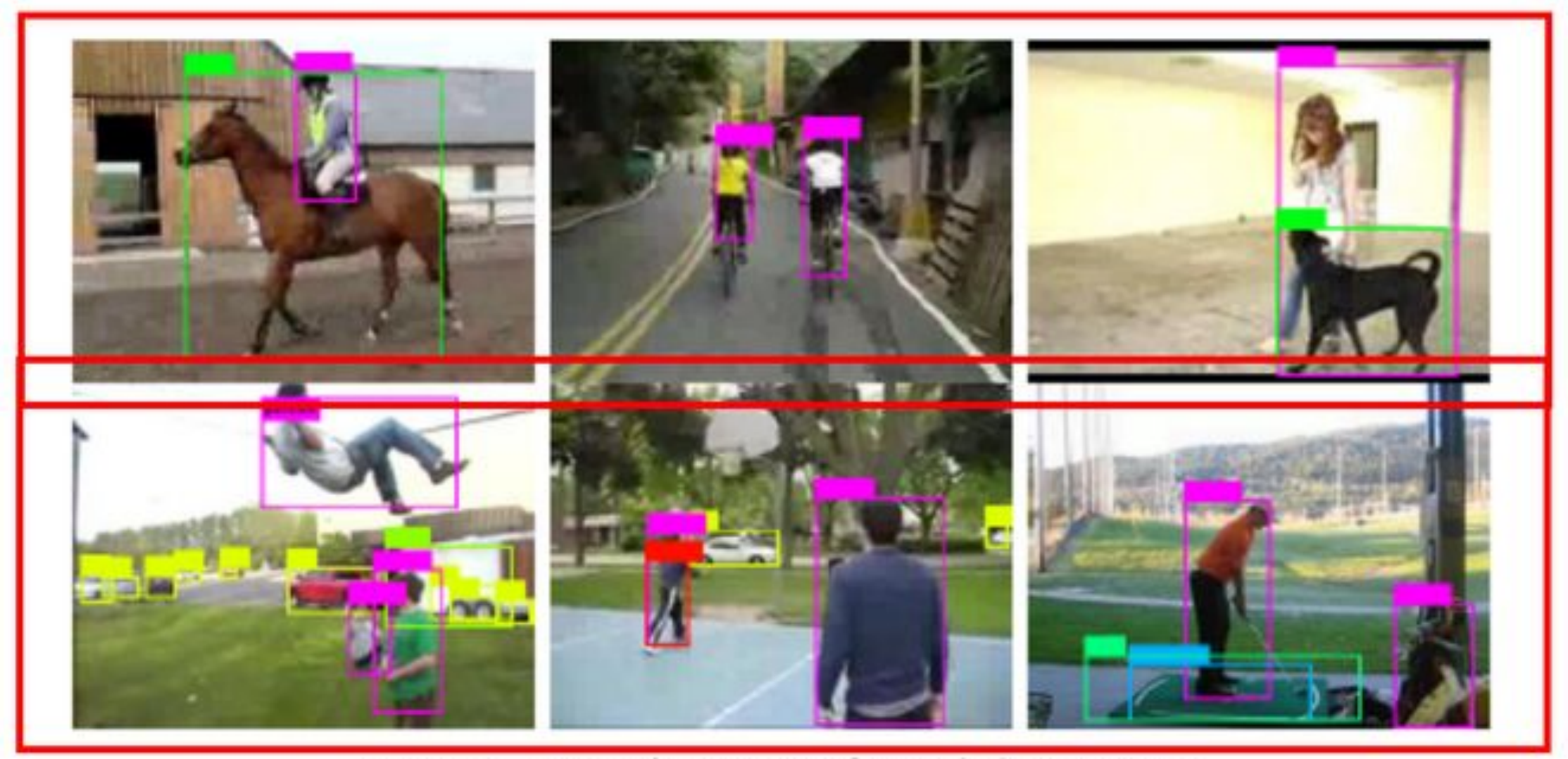


Figure – Initialization of candidate areas

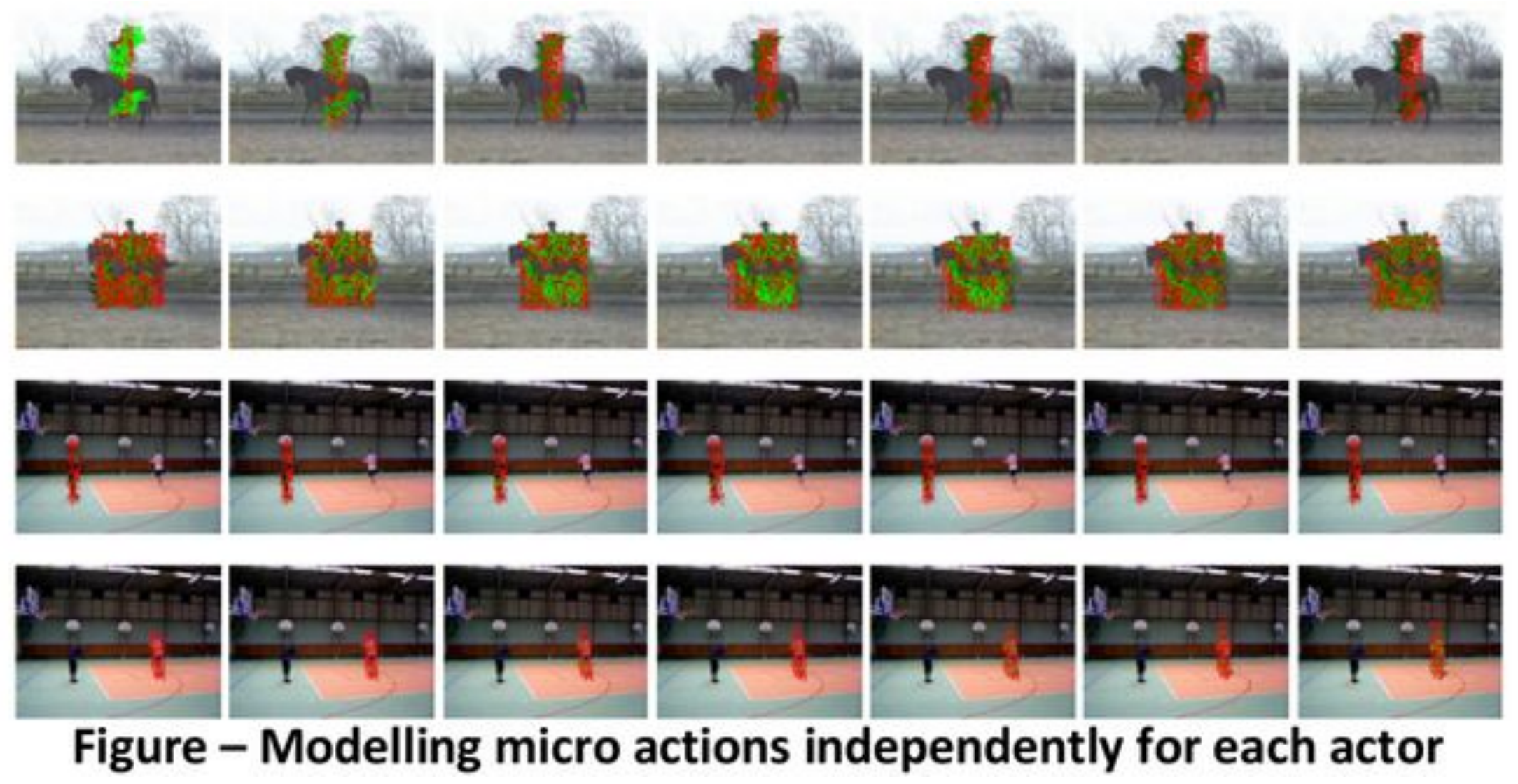
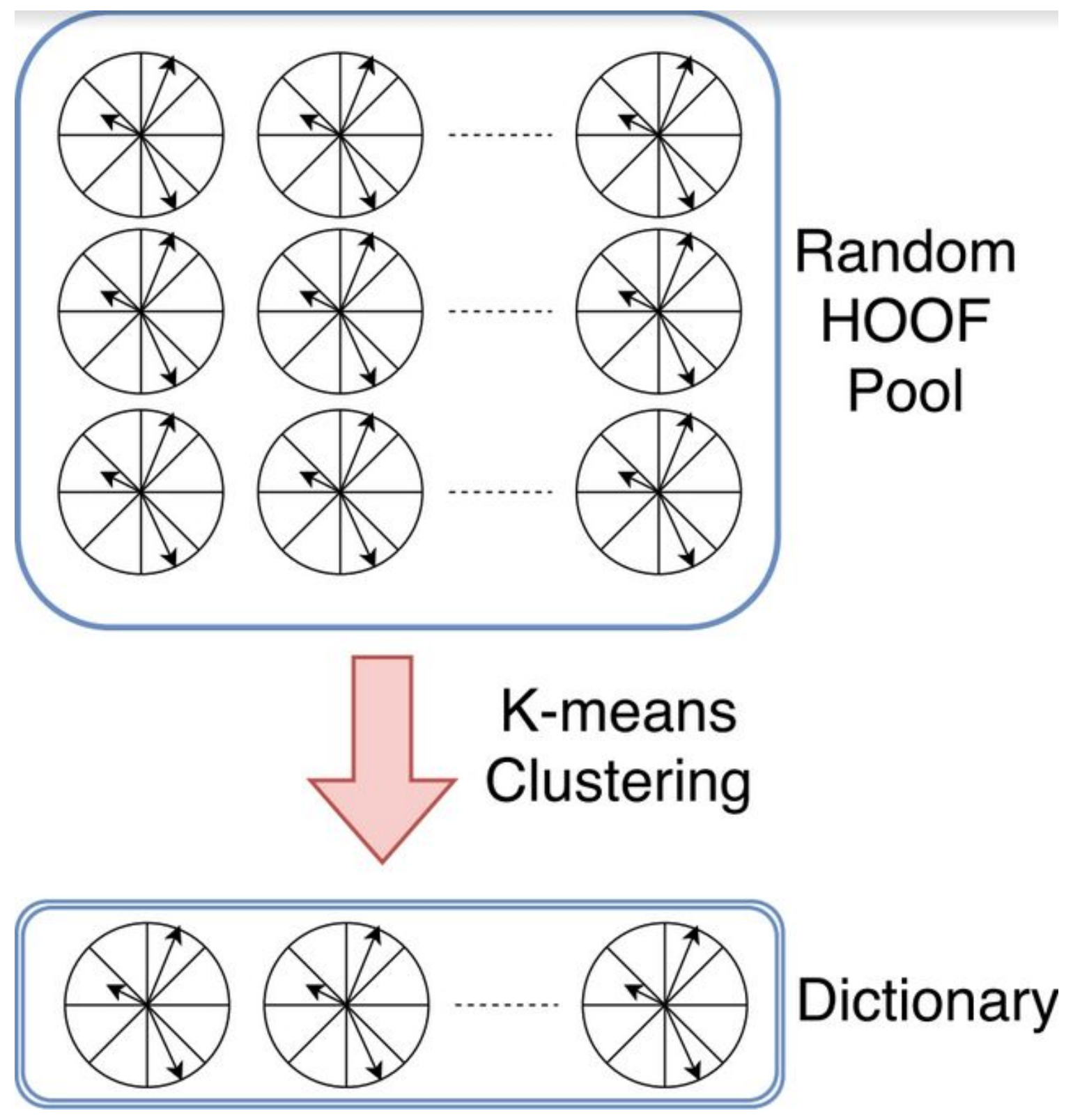


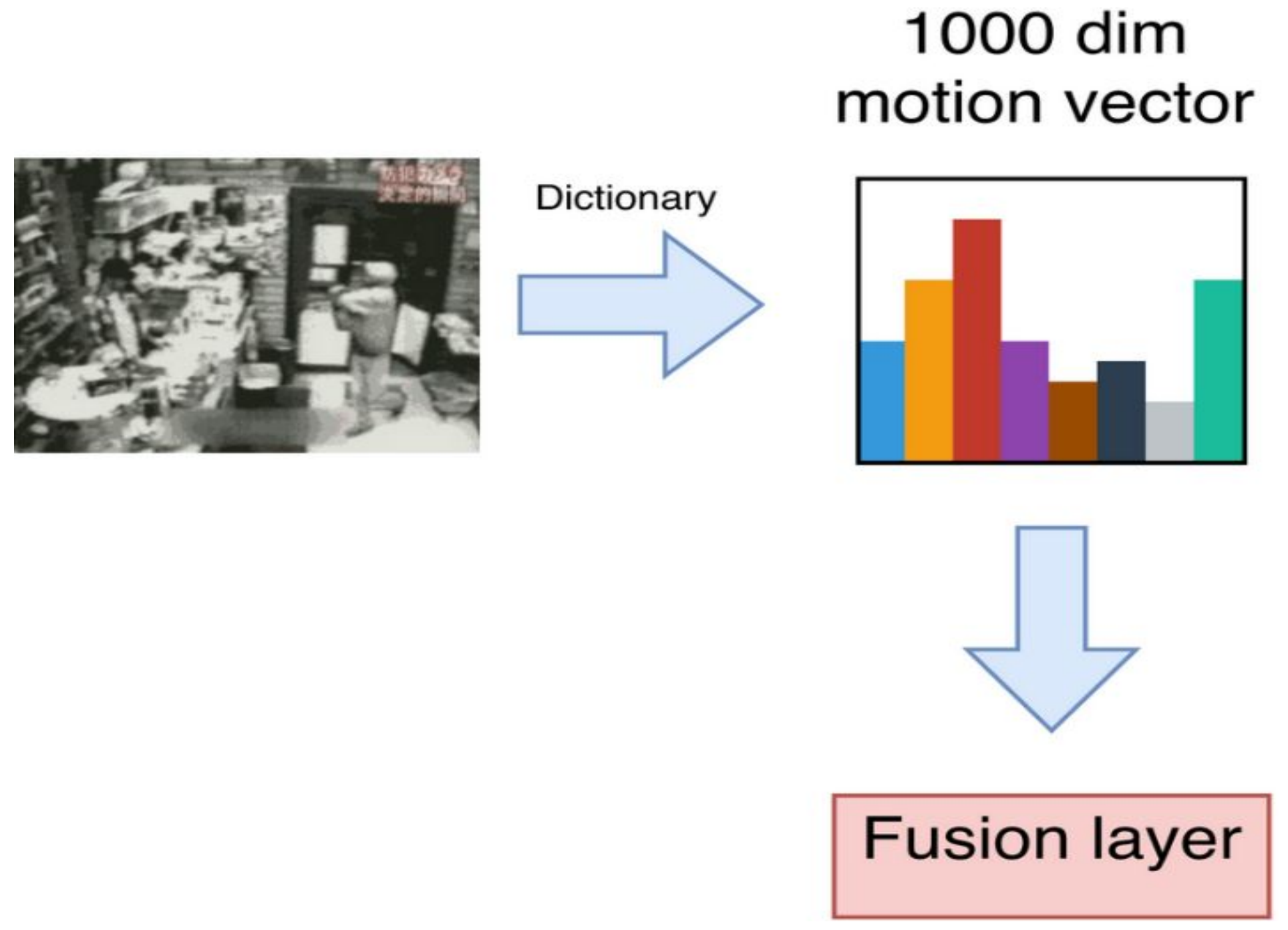
Figure – Modelling micro actions independently for each actor

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

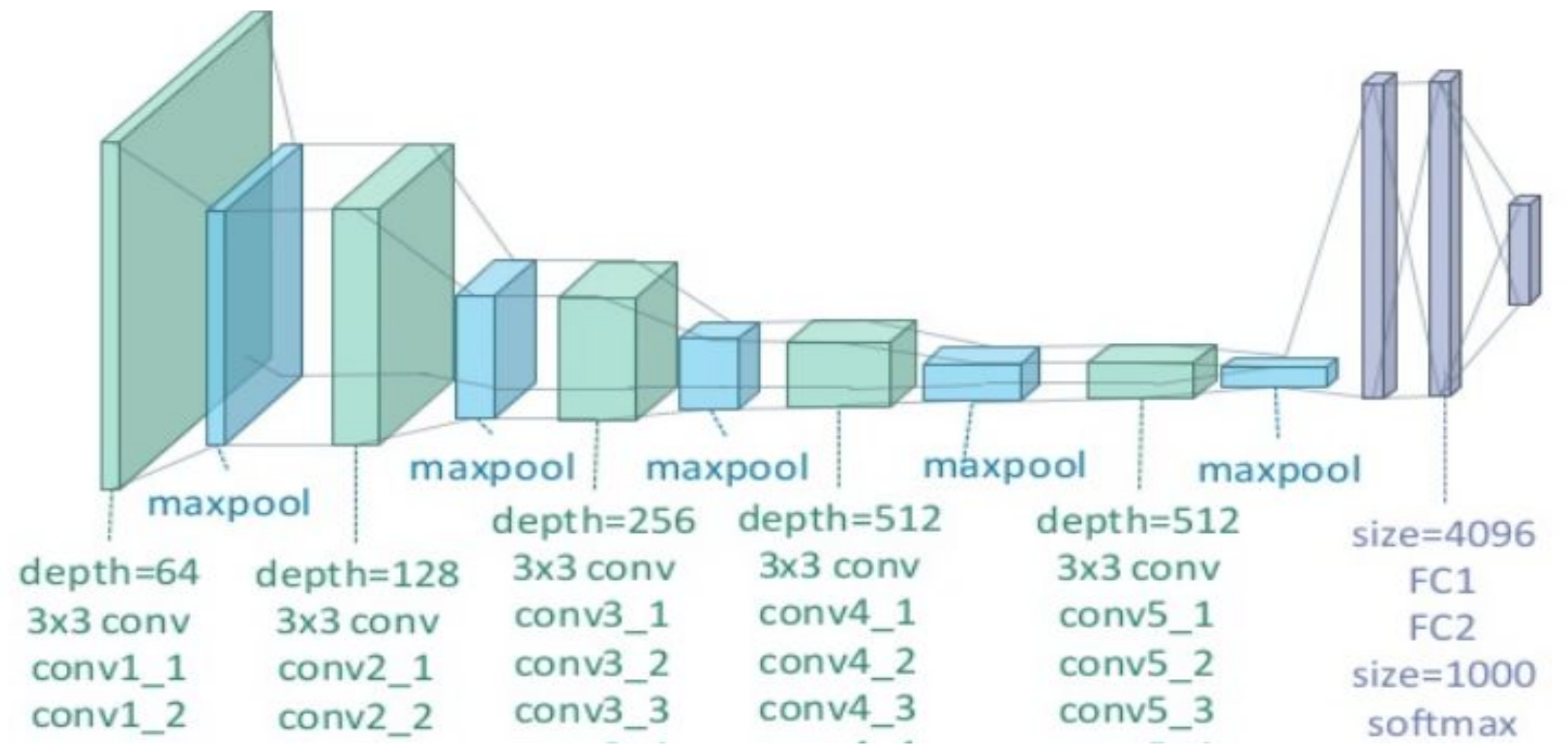
We randomly took 100,000 HOF features and applied k-means to get 1000 clusters. Here we used fractional metric instead of L_2 norm due to the fact that this won't perform well in the high dimensional space.



If there are **n** micro action vectors in a video segment, it means that there are **n** moving objects tracked in the segment. Then we join these micro action vectors to get a high level action vector. we applied some smoothing to the final vector to reduce the noise.



To get the static features we feed each frame of the segment to a pre-trained Convolutional Neural Net, and got the static vector on the softmax layer (with dimension 1000). Then we got the average of static vectors from each of the frames.



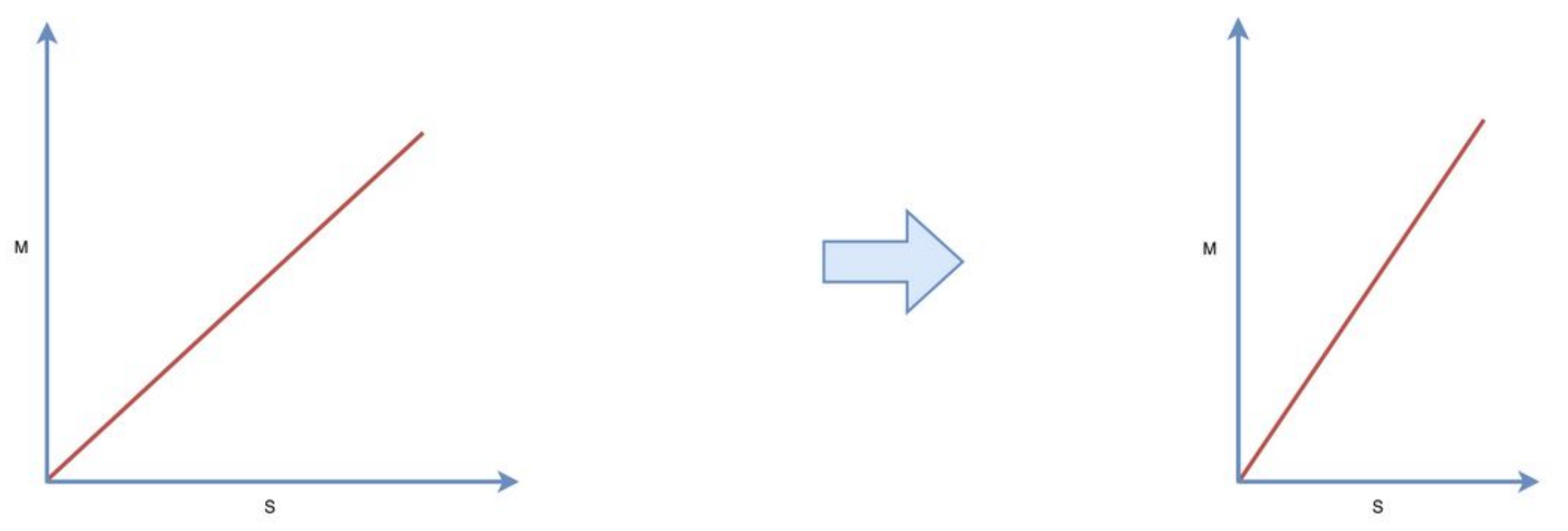
Fusion of Motion and static vectors

We fuse both motion and static vectors to get one vector representing a video segment. By using cholesky transformation we can get a correlated vector for two uncorrelated vectors. We apply this transformation to the motion and static vectors and get a fused vector. The fused vector contains information about both vectors according to the correlation ratio. This give us the freedom to manually tune the correlation of these vectors. By doing this we were able to obtain optimum accuracies.

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

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

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

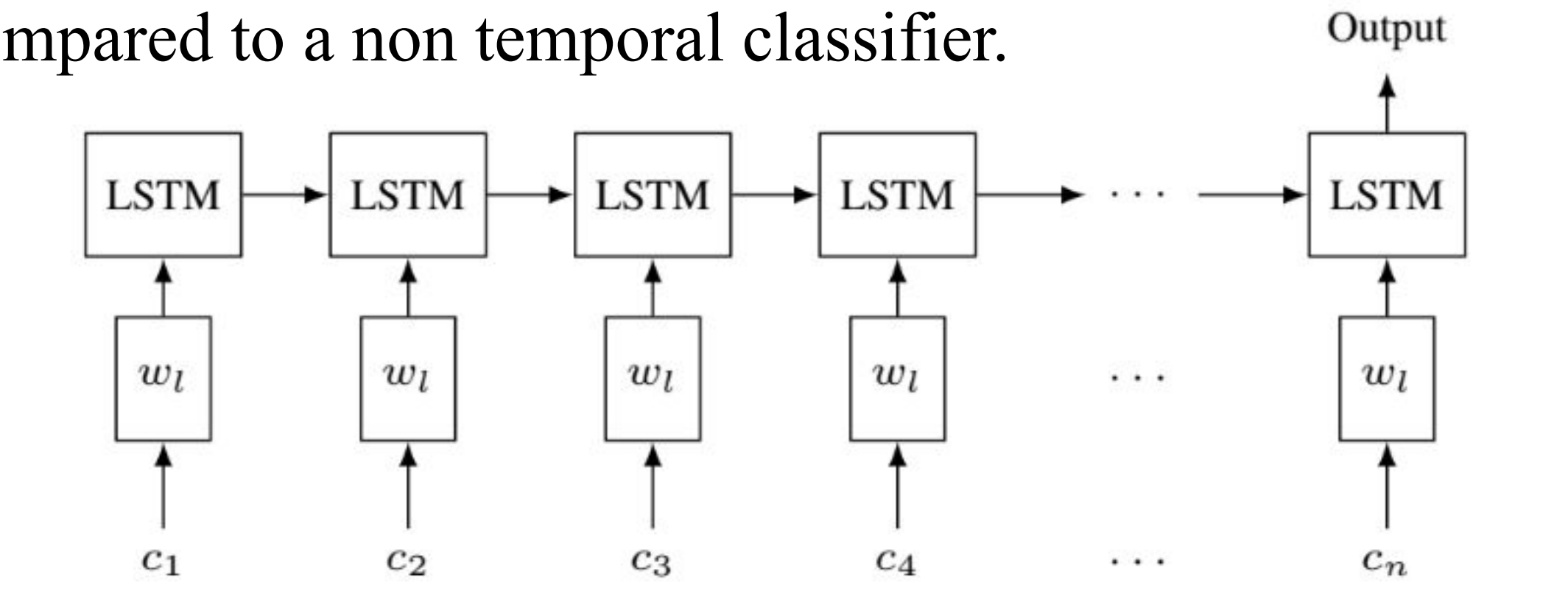


In the second method, we first measured the amount of information captured in each motion and static vectors. Then we fit a joint Gaussian distribution to these vectors and got a fused vector along the line which is closer to the vector with more information. This gradient is calculated by the ratio between the entropies of the two distributions.

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

Capture temporal evaluation

We use these fused vectors to train a RNN with LSTM layers. We also show that the temporal data has more information about the video by comparing it with a random forest classifier, which shows that LSTMs are performing better in capturing temporal evaluation compared to a non temporal classifier.

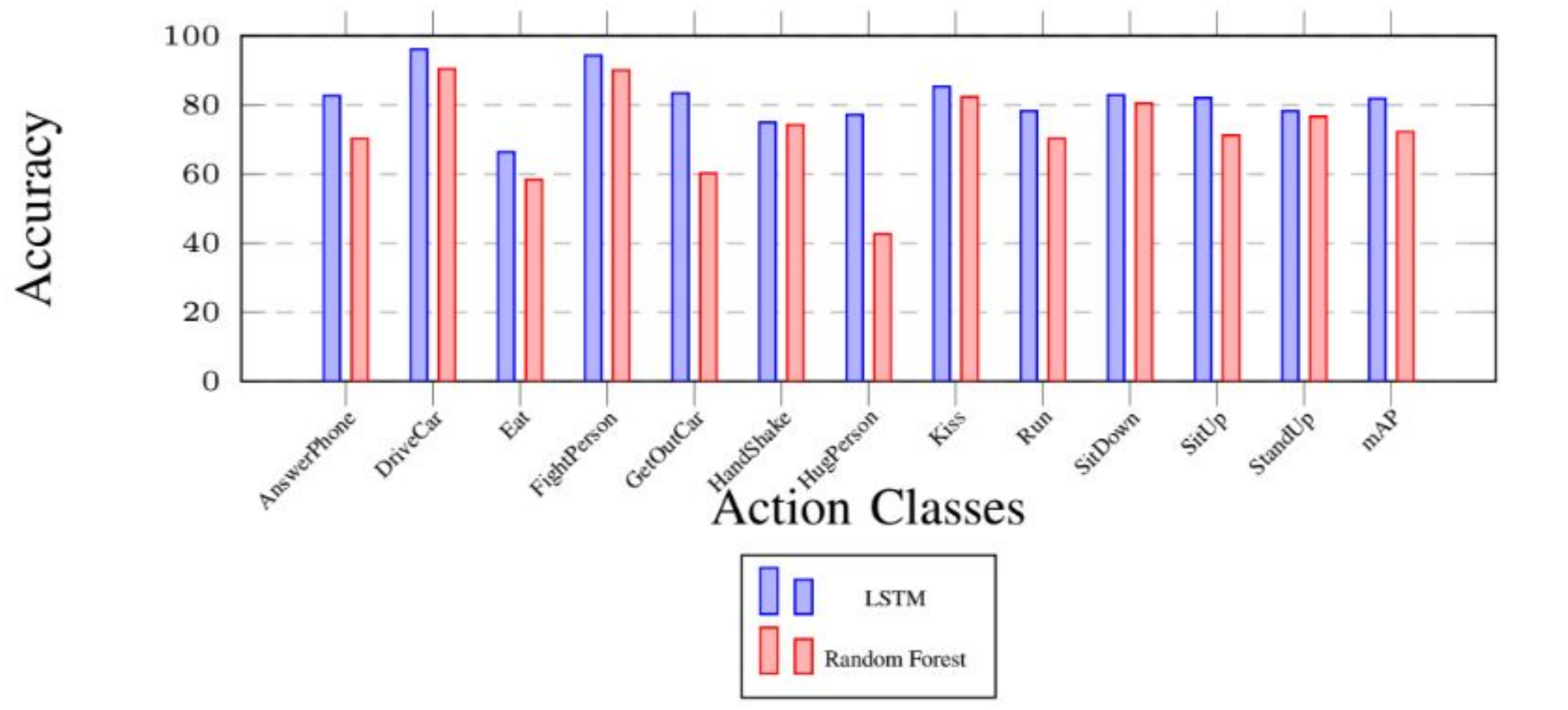


UCF-11 dataset

Results

We tested our framework on Hollywood2 dataset and UCF-11 Dataset. The figure below shows a comparison of overall accuracies with the state-of-art results.

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



Acknowledgment

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