Detection and Classification of Interacting Persons

Scott Blunsden[(1)], Robert Fisher [(2)]
(1) European Commission Joint Research Centre, Italy
(2) University of Edinburgh, Scotland

ABSTRACT

This chapter presents a way to classify interactions between people. Examples of the interactions we investigate are; people meeting one another, walking together and fighting. A new feature set is proposed along with a corresponding classification method. Results are presented which show the new method performing significantly better than the previous state of the art method as proposed by [Oliver et al., 2000].

INTRODUCTION

This chapter presents an investigation into classification of multiple person interactions. There has been much previous work upon identifying what activity individual people are engaged in. [Davis and Bobick, 2001] used a moment based representation based on extracted silhouettes and [Efros et al., 2003] modeled human activity by generating optical flow descriptions of a person's action. Descriptions were generated by first hand-tracking an individual, re-scaling to a standard size and then taking the optical flow of a persons actions over several frames. A database of these descriptions was created and matched to novel situations. This method was extended by [Robertson, 2006] who also included location information to help give contextual information to a scene. Location information is of assistance when trying to determine if someone is loitering or merely waiting at a road crossing. Following on from flow based features [Dollar et al., 2005] extracted spatial-temporal features to identify sequences of actions.

Ribeiro and Santos-Victor [Ribeiro and Santos-Victor, 2005] took a different approach to classify an individual's actions in that they used multiple features calculated from tracking (such as speed, eigenvectors of flow) and selected those features which best classified the persons actions using a classification tree with each branch using at most 3 features to classify the example.

The classification of interacting individuals was studied by [Oliver et al., 2000] who used tracking to extract the speed, alignment and derivative of the distance between two individuals. This information was then used to classify sequences using a coupled hidden Markov model (CHMM). [Liu and Chua, 2006] expanded the two person classification to three person sequences using a hidden Markov model (HMM) with an explicit role attribute. Information derived from tracking was used to provide features such as the relative angle between two persons to classify complete sequences. Xiang and Gong [Gong and Xiang, 2003] again used a CHMM to model

interactions between vehicles on an aircraft runway. These features are calculated by detecting significantly changed pixels over several frames. The correct model for representing the sequence is determined by the connections between the separate models. Goodness of fit is calculated by the Bayesian information criterion. Using this method a model representing the sequences actions is determined.

Multi-person interactions within a rigid formation was also the goal of [Khan and Shah, 2005] who used a geometric model to detect rigid formations between people, such an example would be a marching band. [Intille and Bobick, 2001] used a pre-defined Bayesian network to describe planned motions during American football games. Others such as [Perse et al., 2007] also use a pre-specified template to evaluate the current action being performed by many individuals. Pre-specified templates have been used by [Van Vu et al., 2003, Hongeng and Nevatia, 2001] within the context of surveillance applications.

SPECIFICALLY WHAT ARE WE TRYING TO DO?

Given an input video sequence the goal is to automatically determine if any interactions between two people are taking place. If any are taking place then we want to identify the class of the interaction. Here we limit ourselves to pre-defined classes which have been previously labeled. To make the situation more realistic there is also a 'no interaction' class. We seek to give each frame a label from a predefined set. For example a label may be that person 1 and person 2 are walking together in frame 56.

The ability to automatically classify such interactions would be useful in cases which are typical of many surveillance situations. Such an ability to automatically recognize interactions would also be useful in video summarization where it could be possible to focus only on specific interactions.

FEATURES AND VARIABLES

Video data is rich in information with the resolution of modern surveillance cameras capable of delivering megapixel resolution at a sustained frame rate of greater than 10fps. Such data is overwhelming and mostly unnecessary for classification of interactions. As a first step, tracking of the individuals is employed. There is a rich body of work upon tracking of people and objects within the literature. See the review by Yilmaz et al. for a good survey of current tracking technology [Yilmaz, A et al. 2006]. For all experiments performed in this chapter the bounding box method of identifying a person was used. The positional information was calculated based upon the centre of this box. This process is illustrated in Figure 1. Such tracking information is typical of the output of many tracking procedures and it will be assumed that such a tracker is available throughout all experiments carried out in this chapter.

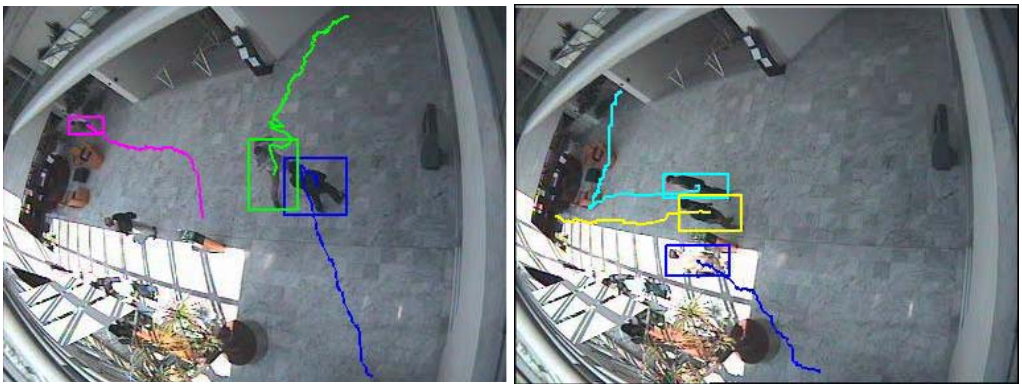

Figure 1 - Bounding box tracking. Colored lines show the previous position of the centre of the tracked object.

Here three types of features are used as input to a classifier. We make use of movement, alignment and distance based features. More details are given in the following sections.

Movement Based features

Movement plays an important role in recognizing interactions. The speed of an individual is calculated as shown in equation (1.1). The double vertical bar (||..||) represents a vector L2 norm as given by $\|\mathbf{x}\|$ = sqrt($\mathbf{x}_1^2+\mathbf{x}_2^2+.....+\mathbf{x}_n^2$) where $\mathbf{x}_n$ refers to the n*th* component of vector **x**.

$$s_i^t = \frac{\left\|\mathbf{p}_i^t - \mathbf{p}_i^{t-w}\right\|}{w} \quad (1.1)$$

Here $\mathbf{p}_i^t$ refers to the position of the tracked object at time *t* for object *i*. Within this work only the two dimensional ($\mathbf{p}_i^t = \left[x_i^t, y_i^t\right]$) case is considered due to tracking information only being available in two dimensions. The temporal offset *w* is introduced due to the high rates which typify many modern video cameras. With frame rates of around 25 frames per second this can mean that differences between the current and last frame (w=1) are very small and may be dominated by noise.

The absolute difference in speed ($\varepsilon_{i,j}^t$) between two tracks is also calculated ($\left|s_i^t - s_j^t\right|$). The vorticity (v_t^i) is measured as a deviation from a line. This line is calculated by fitting a line to a set of previous positions of the trajectory $\mathbf{P}_i^t = \left[\mathbf{p}_i^{t-w},..,\mathbf{p}_i^t\right]$. The window size *w* is the same as that used in equation (1.1). At each point the orthogonal distance to the line is found. The total distance of all points are then summed and normalized by window length and so give a measure of the vorticity.

Alignment Based features

The alignment of two tracks can give valuable information as to how they are interacting. The degree of alignment is common to [Gigerenzer et al., 1999; Oliver et al., 2000; and Liu and Chua, 2006] who all make use of such information when classifying trajectory information. To calculate the dot product the heading (**h**) of the object is taken as in equation (1.2) and the dot product (1.3) is calculated from the directions of tracks *i* and *j*.

$$\hat{\mathbf{h}}_i^t = \frac{\mathbf{p}_i^t - \mathbf{p}_i^{t-w}}{\left\|\mathbf{p}_i^t - \mathbf{p}_i^{t-w}\right\|} \tag{1.2}$$

$$a_{i,j}^t = \hat{\mathbf{h}}_i^t \cdot \hat{\mathbf{h}}_j^t \tag{1.3}$$

In addition to the alignment between two people the potential intersection ($\gamma_t^{i,j}$) of two Trajectories is also calculated. Such features are suggested in [Gigerenzer et al., 1999] and [Liu and Chua, 2006]. We first test for an intersection of the headings. This is achieved as shown in Algorithm 1.

$\mathbf{d}_{i,j}^t = \mathbf{p}_j^t - \mathbf{p}_i^t$ //distance between targets j and i at time t

$c_{i,j}^t = \mathbf{h}_i^t \otimes \mathbf{h}_j^t$ //cross product of the headings of i and j at time t

if $c_{i,j}^t \neq 0$ //non zero cross product

$$\delta_{i,j}^t = \frac{\mathbf{d}_{i,j}^t \times \mathbf{h}_j^t}{c_{i,j}^t}$$

$l_{i,j}^t = \mathbf{p}_i^t + \delta_{i,j}^t \bullet \mathbf{h}_i^t$ //they intesect at this point

elseif $c_{i,j}^t = 0$ //parallel but may be in opposite direction

if $\left\|\mathbf{d}_{i,j}^t\right\| > 0$ and $\mathbf{h}_i^t = -\mathbf{h}_j^t$ and $\left\|\mathbf{h}_i^t\right\| \times \left\|\mathbf{h}_j^t\right\| > 0$

$$r_i^t = \frac{\left\|\mathbf{h}_i^t\right\|}{\left\|\mathbf{h}_i^t\right\| + \left\|\mathbf{h}_j^t\right\|}$$

$$\delta_{i,j}^t = \frac{\mathbf{d}_{i,j}^t \times \mathbf{r}_i^t}{c_{i,j}^t}$$

$l_{i,j}^t = \mathbf{p}_i^t + \delta_{i,j}^t \bullet \mathbf{h}_i^t$ //they intesect at this point

else they do not meet

end

Algorithm 1- Algorithm to determine the meeting of two trajectories

Distance Based features

Distance is a good measure for many types of interaction, for example meeting is not possible without being in close physical proximity. First the Euclidean distance is taken and is used as given in equation (1.4) below.

$$d_{i,j}^{t} = \left\| \mathbf{p}_i^t - \mathbf{p}_j^t \right\| \quad (1.4)$$

The derivative of the distance was also calculated. This is the difference in distance at contiguous time steps. It is calculated as shown in equation (1.5) below.

$$\hat{d}_{i,j}^{t} = \frac{\sum_{t-w}^{t} d_{i,j}^{t}}{\left\| t - (t-w) \right\|} \quad (1.5)$$

An instantaneous measure such as the distance and the derivative of the distance can both be prone to short term tracking errors. In an effort to remove this effect a window size containing *w* points was averaged (as in $\mathbf{P}_i^t$ in the previous section). The distance was calculated for every point (as in equation (1.4)) in this window.

Final feature vector

The final feature vector for each pair of people is given in equation (1.6) below:

$$\mathbf{r}_{i,j}^{t} = \left[s_i^t, s_j^t, \hat{s}_i^t, \hat{s}_j^t, \varepsilon_{i,j}^t, a_{i,j}^t, d_{i,j}^t, \hat{d}_{i,j}^t, v_i^t, v_j^t, \gamma_{i,j}^t \right] \quad (1.6)$$

The vector between persons *i* and *j* at time *t* is made up of the speed of each person (s_i^t, s_j^t) and the change in speed $\hat{s}_i^t, \hat{s}_j^t$. The alignment, distance and change in distance at a particular point in time is given by $a_{i,j}^t, d_{i,j}^t$ and $\hat{d}_{i,j}^t$. The vorticity at a particular point in time for each person is given by v_i^t, v_j^t whilst the possibility of an intersection between two trajectories is given by $\gamma_{i,j}^t$. This gives a final feature vector containing 11 elements. A further processing step of normalizing the training data to have zero mean and unit variance was also taken.

Observation Window Size

Throughout these experiments we investigated the role of varying the number of video frames used before making a decision as to what is happening within the frame. Figure 2 (below) shows how this is achieved. Throughout this work we used information from before and after the current frame in order to classify it. Typically the frames relate to a small quantity of time (1-2 seconds) and help with the lag problem when your decision is biased by the large amount of previous information used in the classification. The window size variation throughout this work is equivalent to a few seconds delay. This was not foreseen as a problem if such an approach was taken in a real surveillance application. The fact that there would be a lag in classification if making use of only previous information seems an appropriate trade-off for an increase in accuracy.

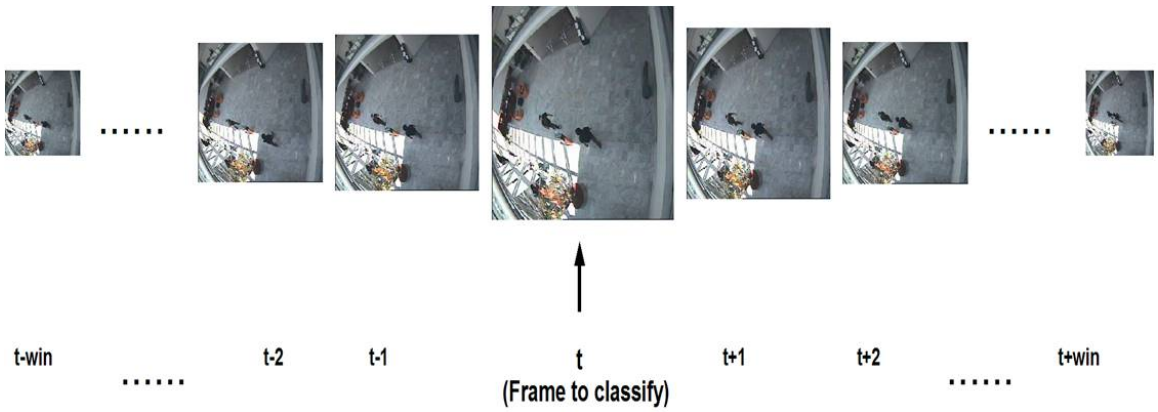

Figure 2 - The frame to classify (t) uses information from w frames around the current frame in order to classify the frame.

CLASSIFICATION

This section introduces the classifiers which are used throughout subsequent experiments. We make use of a simple linear discriminant classifier (LDA) which is non-probabilistic and provides a baseline for performance. This is introduced in the next section. We then briefly introduce the hidden Markov model (HMM) which is widely used throughout the literature. We then introduce a newer model, the conditional random field (CRF). Finally the previous best method as suggested by [Oliver, 2000] is briefly reviewed.

Linear Discriminant Classifier

Linear discriminant analysis (LDA) seeks to maximize the objective function given in equation (1.7) below. Here S_W is the within class scatter matrix with S_B being the between class scatter matrix.

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} \quad (1.7)$$

The objective function (equation (1.7)) is often referred to as the signal to noise ratio. What we are trying to achieve is a projection (**w**) which maximizes the distance of the class means relative to the (sum of) variances of a particular class. To generate a solution it is noted that equation (1.7) has a property whereby it is invariant with respect to scaling of the **w** vectors (eg $\mathbf{w} \to \alpha \mathbf{w}$ where α is some arbitrary scaling). Therefore **w** can be chosen such that $\mathbf{w}^T \mathbf{S}_W \mathbf{w} = 1$. It is also common (and indeed the case here) to perform a whitening step (zero mean and unit variance) on the data prior to input into this method.

This maximization can be turned into a regular eigenvalue problem. Projections are found for each class (ie one class vs all others) and the mean and variance (**C**) of the class projection are found. In order to classify a novel point the new point is projected with S_W^{-1}. The class label is determined by taking the smallest Mahalonobis distance between the calculated class model's mean and variance and the new test point.

Hidden Markov Model

Hidden Markov models (HMM's) have been introduced by (among others) Rabiner [Rabiner,1990]. The model is parameterized by the prior distribution $\prod$ with each element π_i representing $\pi_i = p(x = i)$ across all hidden states $i \in [1,..,N]$. The stationary state transition matrix **A** is used to represent the probability of a transition from one state (i) to another (j) through time. An entry within the stochastic matrix **A** is referenced by $a_{i,j} = p(x_t = i \mid x_{t-1} = j)$. Within this work we are concerned with continuous real valued inputs ($\mathbf{r}_t$) which can be accommodated within the model by using a Gaussian mixture model to represent the input distribution $p(\mathbf{r}_t \mid x_t = j)$.

$$b_j(\mathbf{r}_t) = \sum_{m=1}^{M} c_{j,m} \mathbf{N}(\mathbf{r}_t, \mu_{j,m}, \mathbf{C}_{j,m}) \quad (1.8)$$

Here the observed data **R** is the vector being modeled, $c_{j,m}$ is the mixture coefficient for the m*th* mixture in state *j*. **N** is a Gaussian distribution with mean vector $\mu_{j,m}$ and covariance $\mathbf{C}_{j,m}$ **f**or the m*th* mixture component in state j. The mixture coefficient c_j must sum to 1.

The hidden Markov model's parameters can thus be represented as $\lambda = (\Pi, \mathbf{A}, \Theta)$ where Θ represent the parameters of the mixture model.

Conditional Random Field

In this section the workings of a conditional random field (CRF) are explained and then the specific formulation as applied in this chapter is given. The structure of the CRF we use is shown in Figure 3. The CRF can be configured to resemble HMM like models. However they can be more expressive in that arbitrary dependencies on the observations are allowed. Using the feature functions of the CRF allows any part of the input sequence to be used at any time during the inference stage. It is also possible that different state's (classes) can have differing feature functions (though we do not make use of this here). The feature functions describe the structure of the model. The CRF is also a discriminative model where as the HMM is a generative one. A potential advantage of the discriminative CRF over generative models is that they have been shown to be more robust to violations of independence assumptions [Lafferty et al., 2001].

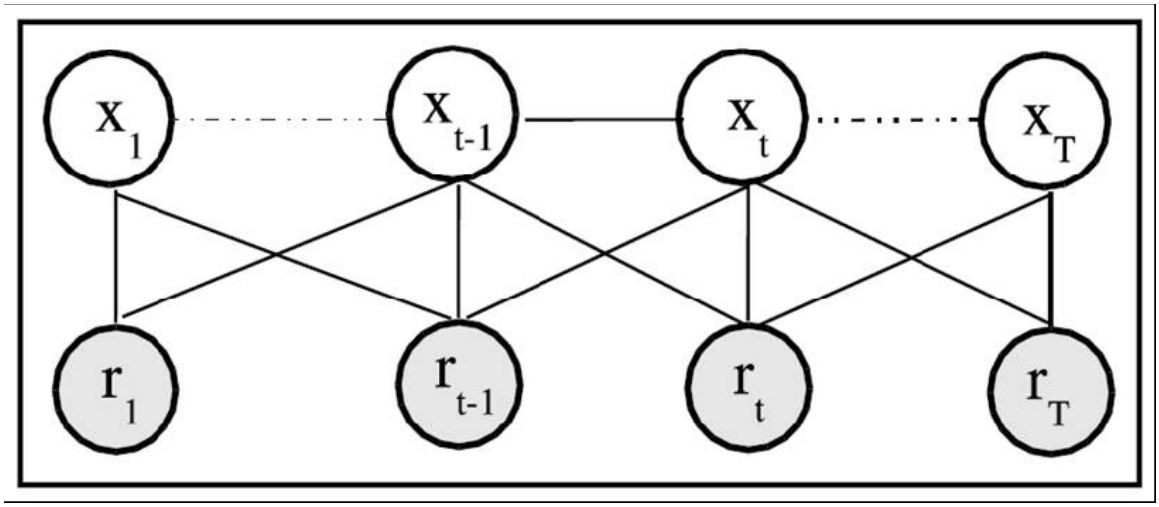

Figure 3- CRF model. The observations (r) are shown for each timestep. The class label x is also shown.

The discrete temporal state at a particular timestep *t* is given by x_t which takes a value from the set of all possible class labels $x \in X = \{1, 2, .., C\}$. Here C is the maximum number of class labels whilst t represents the time with T being the maximum length of the sequence. Observations at time t are denoted as r_t with the joint observations given as $\mathbf{R}_t = (\mathbf{r}_1, .., \mathbf{r}_t)$. Likewise the joint state is given by $\mathbf{X}_t = (x_1, .., x_t)$. For notational compactness we shall refer to $\mathbf{X}_t$ as $\mathbf{X}$ and $\mathbf{R}_t$ as $\mathbf{R}$ in accordance with other author's [Sminchisescu et al.,2005, and Wallach, 2004].

The distribution over joint labels $\mathbf{X}$ given observations $\mathbf{R}$ and parameters Θ are given by:

$$p_\theta(\mathbf{X} \mid \mathbf{R}) = \frac{1}{Z_\theta(\mathbf{R})} \prod_{c \in C(\mathbf{X}, \mathbf{R})} \phi_\theta^c(\mathbf{X}_c, \mathbf{R}_c) \tag{1.9}$$

Within this equation ϕ_θ^c is a real valued potential function of the clique c and $Z_\theta(\mathbf{R})$ is the observation dependent normalization (sometimes referred to as a partition function). C(**X**,**R**) is the set of maximal cliques.

Here a first order linear chain is used (as show in Figure 3). The cliques are pairs f neighboring states (x_t, x_t+1), whilst the connectivity among observations is restricted to that shown in the graph (Figure 3- CRF model. The observations (r) are shown for each timestep. The class label x is also shown.). A CRF model with T timesteps, as used here, can be re-written in terms of exponential feature functions F_θ computed in terms of weighted sums over the features of the cliques. This exponential feature formulation is given below in equation (1.10)

$$p_\theta(\mathbf{X} \mid \mathbf{R}) = \frac{1}{Z_\theta(\mathbf{R})} \exp(\sum_{t=1}^{T} F_\theta(x_t, x_{t-1}, \mathbf{R})) \quad (1.10)$$

The conditional log likelihood of the CRF is given below. Assuming that the training data is fully labeled $\left\{\mathbf{X}^d, \mathbf{R}^d\right\}_{d=1,..,D}$ the parameters of the model are obtained by optimization of the following function:

$$L_\theta = \sum_{d=1}^{D} \log p_\theta(\mathbf{X}^d \mid \mathbf{R}^d) = \sum_{d=1}^{D} (\sum_{i=t}^{T} F_\theta(x_t^d, x_{t-1}^d, \mathbf{R}^d) - \log Z_\theta(\mathbf{R}^d)) \quad (1.11)$$

In order to make parameter optimization more stable the problem often makes use of a regularized term (R_θ). The problem then becomes one of optimizing the penalized likelihood ($L_\theta + R_\theta$). The regularizing term used here was chosen to be $R_\theta = -\|\theta\|^2$.

Once trained novel input is given to a CRF model and a probability distribution is given throughout all timesteps for all classes. In this case we choose the highest probability as being the classification label of the new example. A Gaussian prior over the input data was used throughout all experiments.

Oliver's Coupled Hidden Markov Model

Here we briefly cover Oliver's method ([Oliver et al., 2000]) of classifying interacting individuals. This work is reviewed as it provides a state of the art method to compare our results with. Oliver used coupled hidden Markov models to model five different interactive behaviors.

Oliver's work is used for comparison with the work presented here. Oliver et al use two feature vectors (one for each chain). These feature vectors are made up of the velocity of the person, the change in distance between the two people and the alignment between the two people. This gives two feature vectors, one for each chain.

For each class the parameters of a two chain coupled hidden Markov model are trained. When classifying the model which produces the highest likelihood for a test sequence is taken as the class label.

RESULTS

This section presents the results obtained by using the methods described in the preceding chapters. Results using a conditional random field (CRF) , hidden Markov model (HMM) and its coupled variation (CHMM) are presented. Results using a linear discriminant model (LDA) are also presented and used as a baseline non-probabilistic method to which results are compared. We present the result of classification over many training and testing subsets to give an indication of the standard deviation and the expected performance when using a method. Results are presented over many different window sizes. The graphs show the averaged performance of the classifier over 50 runs. The standard deviation is given by the shaded regions.

Experimental setup

The CAVIAR dataset has been previously used in [Dee and Hogg, 2004a, Wu and Nevatia, 2007] however there is not a universally agreed training and testing set. Therefore it was deemed that in order to characterise an algorithm's true performance upon a dataset it should be tested with different subsets of the entire data. This will give an indication of the expected performance of the algorithm rather than finding a particularly good (in terms of classification accuracy) subset.

We are interested in comparing the four methods as described in the previous section. Furthermore the role of time is investigated. We seek to investigate what is the optimal length of time a sequence should be watched before a decision is made. Results comparing each method and the effects of time are given in the following sections.

Throughout the training procedure the testing set was kept separate from the training data and was unseen by the learning algorithm until testing time. Therefore only the training set was used when determining the parameters of the learning model. Training and testing sets were split 50/50 on a per class basis. Partitioning was done per-class rather than over the whole dataset due to the uneven distribution of classes. Such a step means that in the training stage the learning algorithm will have examples of every type of class. We would not expect a correct classification on unseen classes and so this measure can stop misleading results. In order to show the average performance this procedure was repeated over 50 different partitions of the training and test data.

The dataset contains examples of complete sequences, for example a sequence consisting of two people walking together may be hundreds of frames long. Our goal is to classify each frame correctly. If we were to take this sequence and split it up as training and testing frames then the classification task would be much simpler as training and testing points would be simply a matter of interpolation between highly similar points. It is for this reason that when deciding on the training and testing data we partition based on the complete sequences. This means that an entire walking together sequence will be assigned to the training set whilst another complete walking together sequence will be assigned to the testing set. This should avoid the pitfall of having training and testing data which is essentially the same. This is especially true as the data has a very strong temporal coupling.

What we are aiming to do is try to give a class label to two interacting individuals. This is shown in Figure 4-. The two people are approaching one another. Between the first and second frame the distance between them decreases. The classifier would assign one label for this interaction (covering both people) to indicate 'approaching'.

Figure 4-Two people approaching one another.

Classification Results

CAVIAR Dataset

The CAVIAR dataset [EC Funded CAVIAR.. (2004)] contains 11,415 frames in which some have labeled examples of interactions. Within this set there are 5 distinct classes which we seek to identify and classify. The 5 classes consist of examples of people: walking together (2,543), approaching (942), ignoring (4,916), meeting one another (1,801), splitting up (879) and fighting (334). The numbers in brackets indicates how many frames contain this behavior.

Overall results for the dataset are shown in Figure 5. These results are the further broken down and shown for each class in Figure 6, Figure 7 and Figure 8.

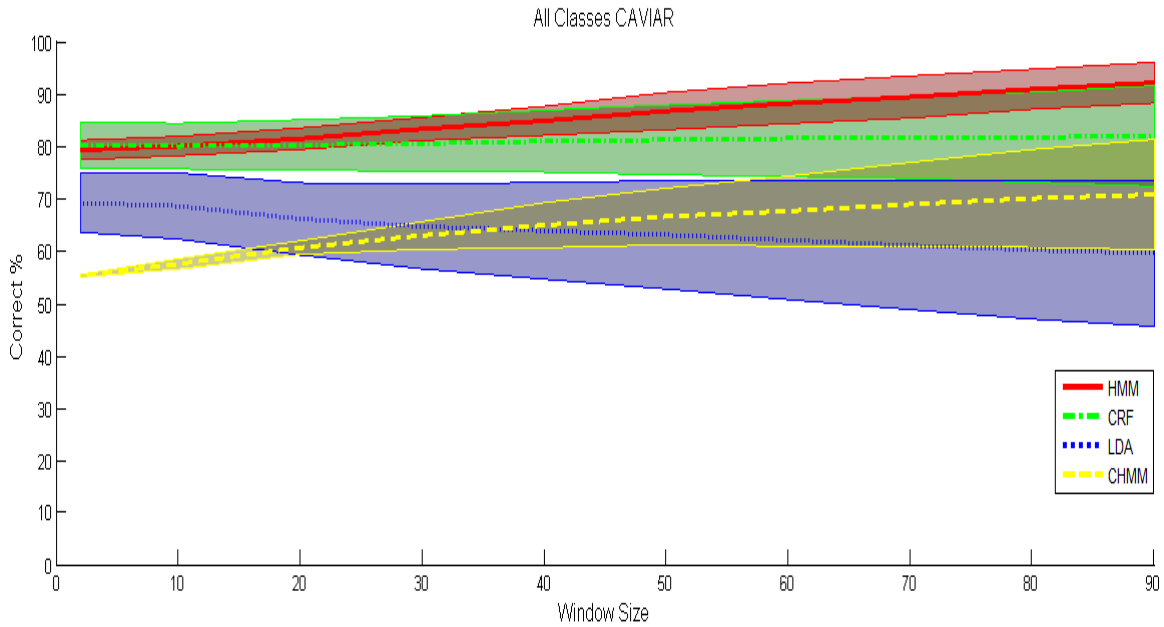

Figure 5- Overall results on the CAVIAR dataset for each method. Lines show averaged results (over 50 runs) whilst the shaded regions show one standard deviation.

It is visible that for small window sizes the CRF method offers the best performance (except perhaps when classifying the approaching behavior). However in this dataset the HMM model gives the best possible average performance. Both the HMM and the CRF using the new proposed features show superior performance to Oliver's method. A particular problem for all methods is in the classification of fighting behavior. A small sample size and the short timescale where any fighting actually occurs contribute towards this. We see that for some cases the window is too small (such as walking together and ignore) or too large (such as ignore and approach). A per class time window or an enhanced feature set would help this problem

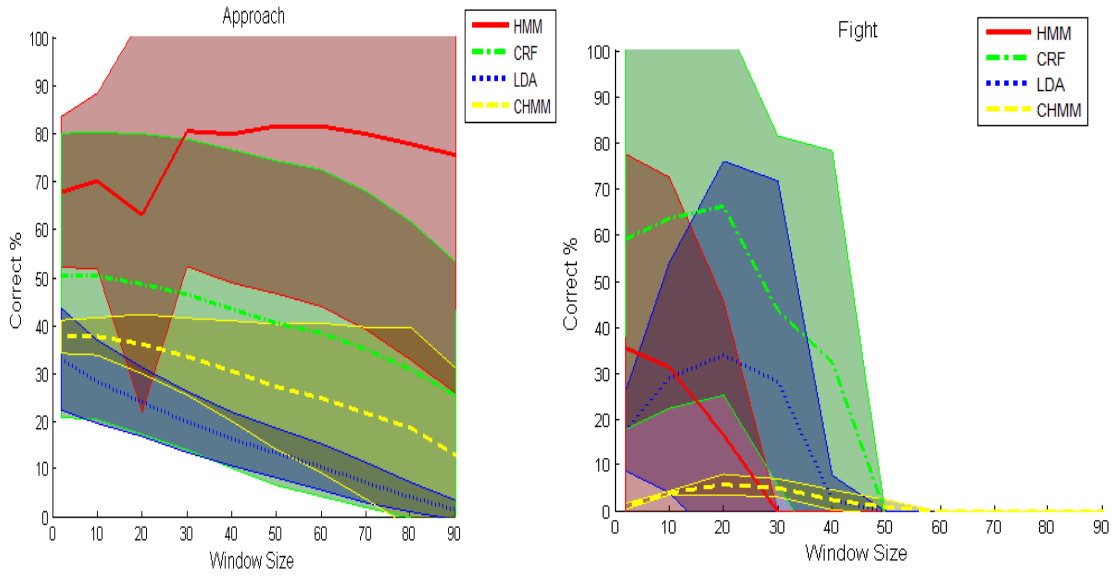

Figure 6 – Results on the 'approaching' and 'fighting' class, for the CAVIAR data.

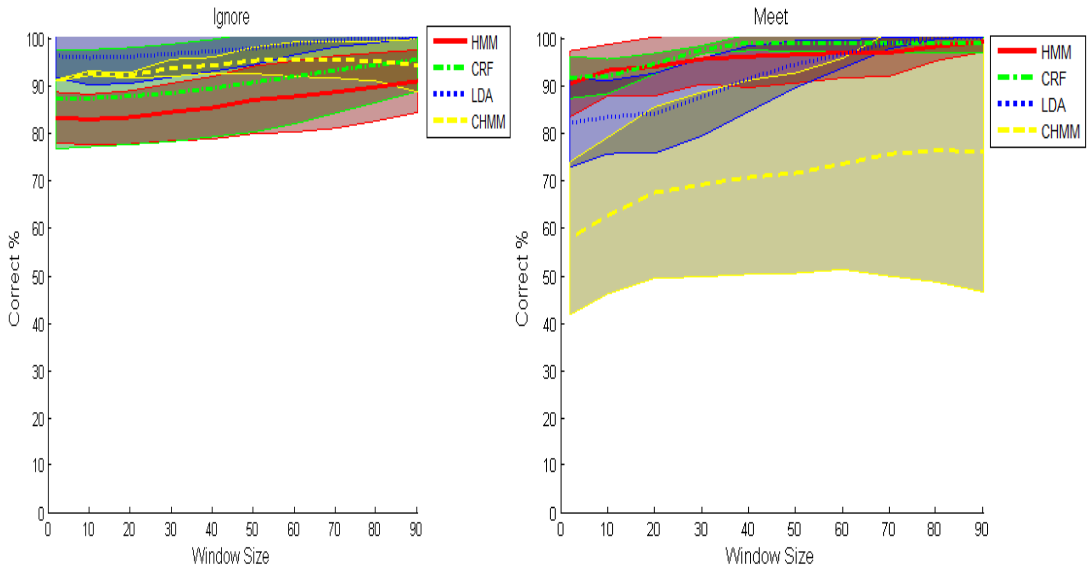

Figure 7 – Results on the 'ignore' and 'meet' class, for the CAVIAR data.

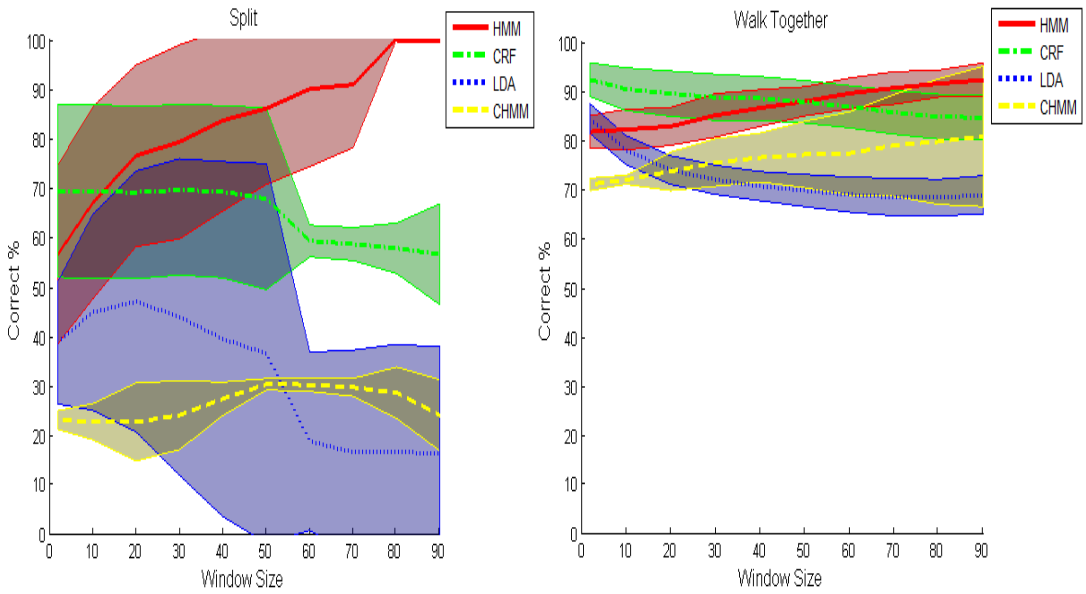

Figure 8 – Results on the 'split' and 'walk together' class, for the CAVIAR data

One of the other features of this dataset is that for "approaching", "splitting" and "fighting" there were perhaps not enough examples to get a build a sufficiently generalisable model. A simple answer would be to say that more data is required. However in many real surveillance applications such an approach is not possible so showing how a method performs using only limited data is still of value.

BEHAVE Dataset

The BEHAVE dataset [Blunsden et al. 2007] contains 134,457 frames which have labeled examples of interactions. Within this set there are 5 distinct classes which we seek to identify and classify. The 8 classes consist of examples of people: In a group (91,421), Approaching (8,991), walking together (14,261), splitting (11,046), ignoring (1,557), fighting (4,772), running (1,870) and chasing (539) one another. The numbers in brackets indicates how many frames contain this behavior.

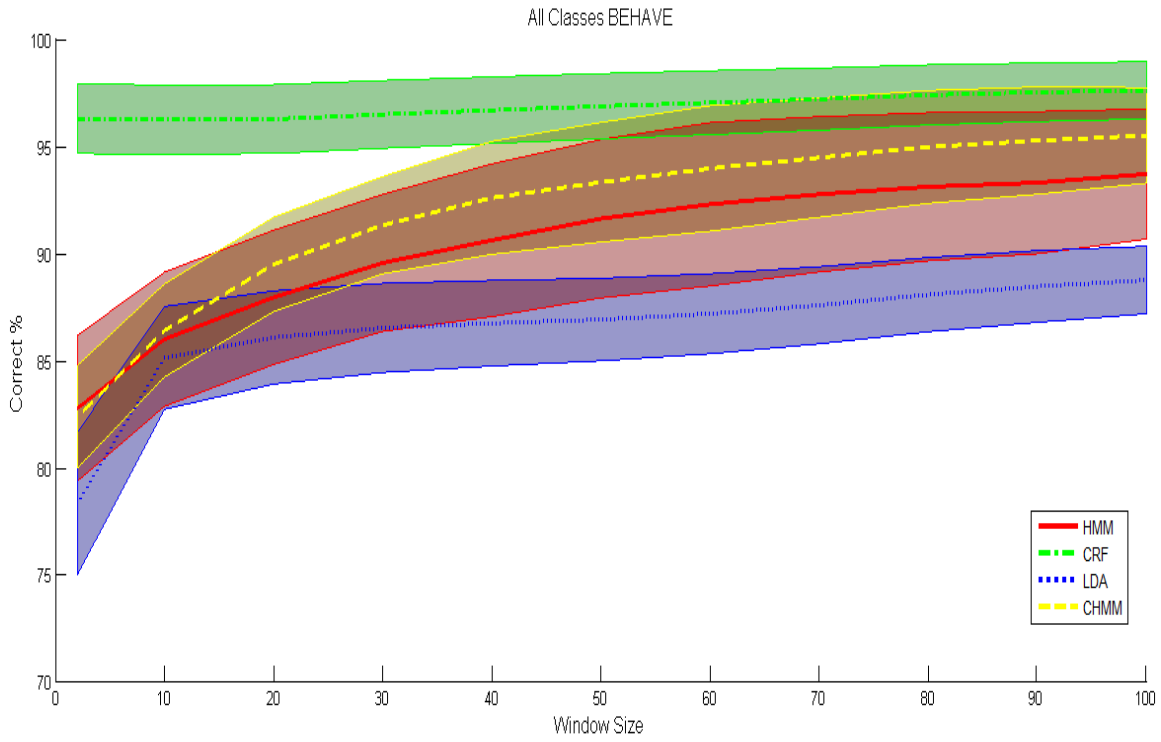

Figure 9- Overall performance on the BEHAVE dataset for each method. Lines show averaged results (over 50 runs) whilst the shaded regions show one standard deviation.

The overall averaged classification is shown in Figure 9. The CRF clearly performs better than all other methods on this dataset for all window sizes. There is a slight increase in performance when the window size is increased when using a CRF. However the effect is more dramatic for both the CHMM, HMM and the LDA method. All three of these methods (HMM, CHMM, LDA) increase in performance as the window size is increased. Significant performance increases are observed between window sizes of 1 and 20.

Per class results (figures Figure 10, Figure 11 and Figure 12) display a similar story where increasing the window size has little effect upon CRF classification. When classifying splitting, approaching (Figure Figure 10) and fighting (Figure 11) increasing the window size improves the performance of the HMM, CHMM and the LDA classifiers. The HMM classifier gives the best performance of all methods when classifying fighting (Figure 6). Increasing window size also gives a similar increase in performance for both the in group classification and the walking together class (Figure 12). However when classifying people in a group the LDA method decreases in performance.

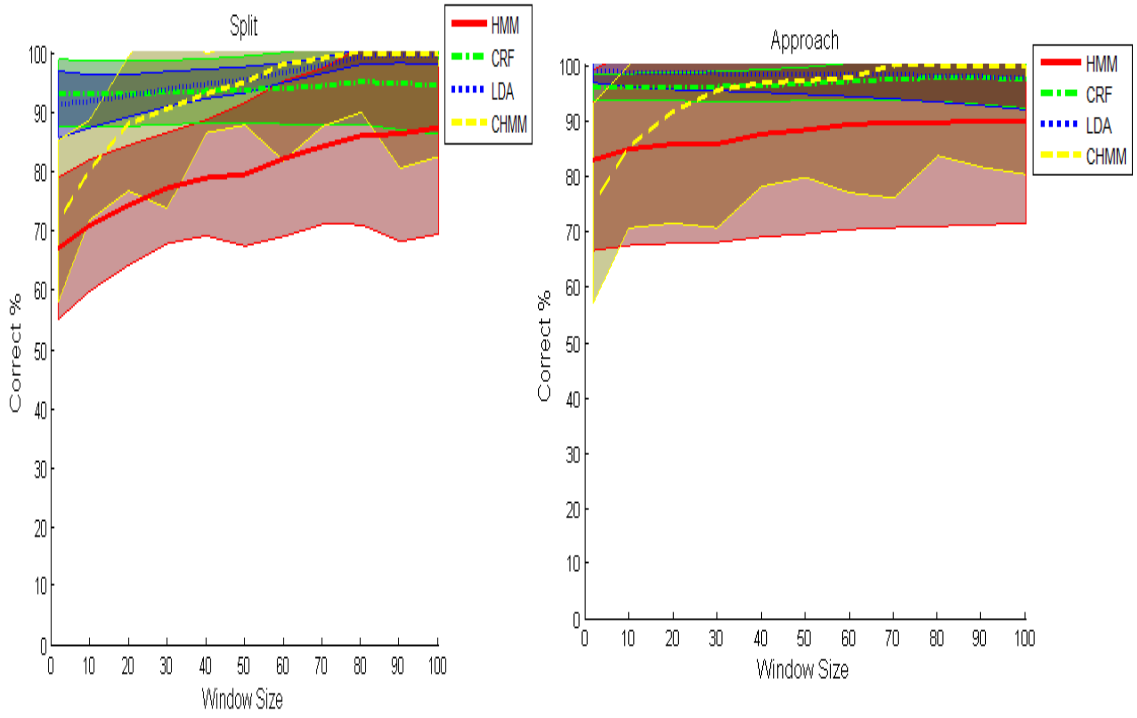

Figure 10-Results on the 'split and 'approach' classes for the BEHAVE dataset.

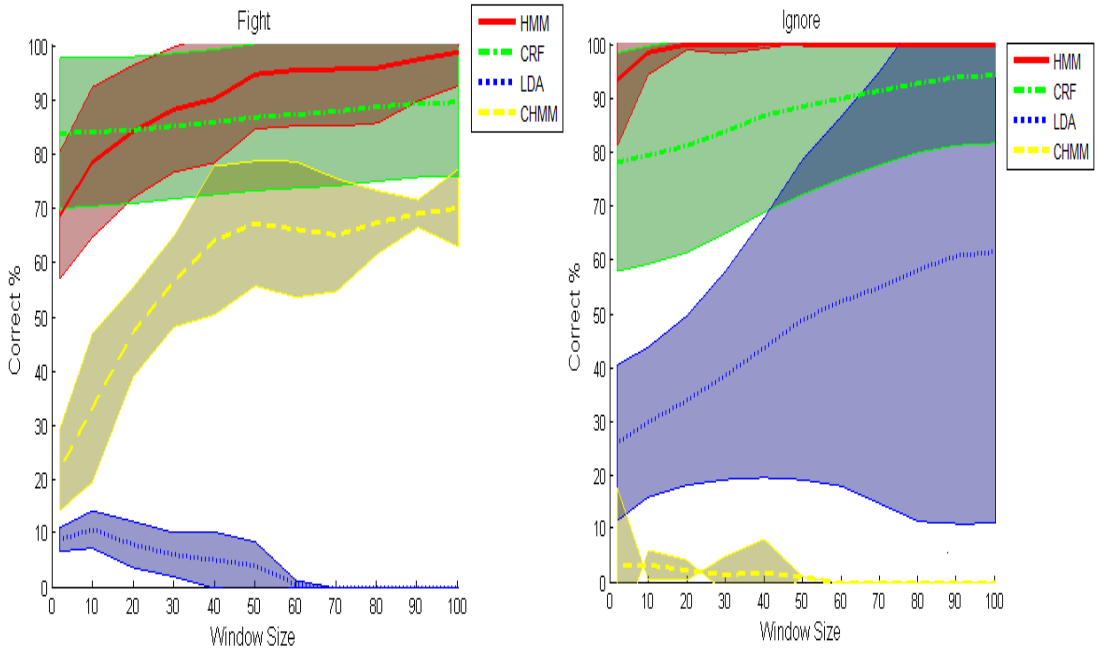

Figure 11 - Results on the 'fight' and 'ignore' classes for the BEHAVE dataset.

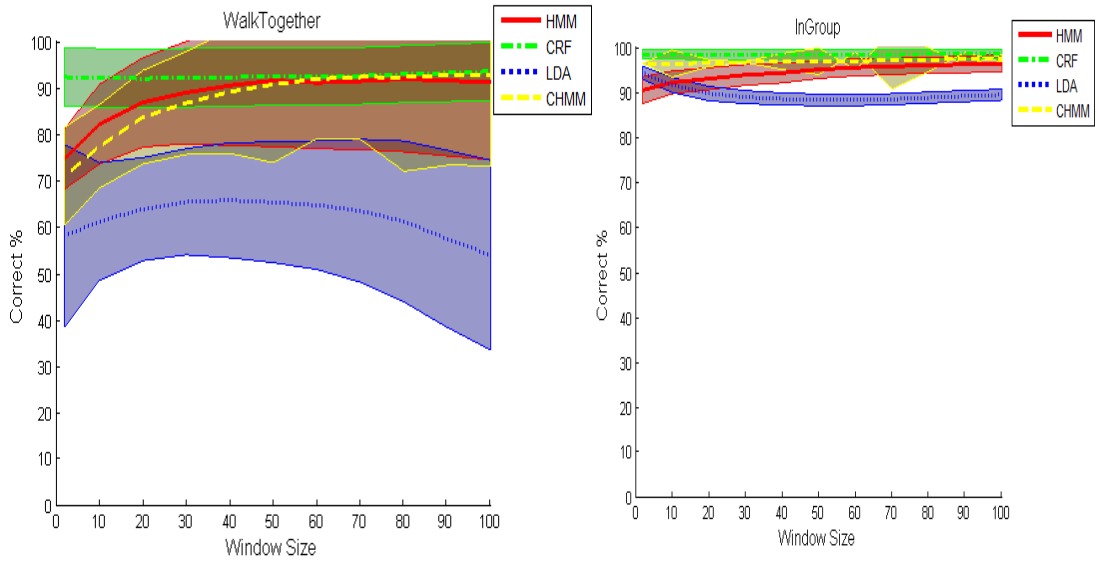

Figure 12- Results on the 'walk together' and 'in group' classes for the BEHAVE dataset.

CONCLUSION

Over all datasets the CRF classifier performs well (80-95%) when using limited information. The previous best method suggested by Oliver [Oliver, 2000] is improved upon using the CRF classifier in conjunction with the new proposed feature set. The new proposed feature set also outperforms Oliver's method when using a HMM model for a great many cases.

The CRF classifier displays an ability to better classify data in the short term compared to the HMM. In contrast the HMM model improves more rapidly when the observation window size is increased suggesting it is better at smoothing the signal over longer sequences. The forward algorithm used to determine the likelihood seems to smooth the signal much better than the CRF. The case of the CHMM the more gradual improvement could be attributable to the larger number of parameters which requires more data in order to represent the data adequately. A suggestion for future work would be therefore to improve the long term temporal model of the CRF that we are using. It should be noted that these comments about the CRF model apply for the single chain structure which is used here. There are many configurations which can be used within the CRF framework. A higher order CRF may produce a better temporal model and so we would expect to see larger improvements when the observation window size is increased. However a CRF model will always be discriminative compared to the HMM and CHMM's generative ability. The ability to generate samples may be important in certain cases (such as estimating a model's complexity [Gong and Xiang, 2003]) however this ability is not required for the classification tasks as presented here.

Throughout all experiments on all of the datasets it is visible that there seems to be an optimal window size for classification of a particular class. For some activities such as fighting in the CAVIAR dataset (Figure 6) the window size is quite short (due to the speed of a fight) where as

for other classes such as the ignore behavior (Figure 7) a longer window size improves performance.

FUTURE WORK

The significance of the role of time within this work has been demonstrated. Future work should seek to exploit this in a principled way. It would also be fruitful to investigate in a principled way automatic feature selection. It would be envisaged that such a procedure could incorporate lots of features and chose them automatically rather than, as is the case here, where it is only possible to use a limited number of features. By having a comparative study of the classifiers on the dataset is also possible to know how much you should rely on them, if for instance you were using some classification criteria in feature selection or to establish an optimal time window.

REFERENCES

S. Blunsden, E. Andrade, A. Laghaee, and R. Fisher. Behave interactions test case scenarios, epsrc project gr/s98146, On Line, September 2007. URL: http://groups.inf.ed.ac.uk/vision/BEHAVEDATA/INTERACTIONS/index.

EC Funded CAVIAR project/IST 2001 37540. found at url: http://homepages.inf.ed.ac.uk/rbf/caviar/, 2004.

J. W. Davis and A. F. Bobick. The representation and recognition of action using temporal templates. In IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 23, pages 257–267. IEEE Computer Society, 2001.

H. M. Dee and D. C. Hogg. Detecting inexplicable behaviour. In British Machine Vision Conference, pages 477–486, September 2004.

P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In PETS, pages 65–72, China, 2005.

A. Efros, A. Berg, G. Mori, and J. Malik. Recognising action at a distance. In In 9th International Conference on Computer Vision, volume 2, pages 726–733, 2003.

G. Gigerenzer, P. M. Todd, and ABC Research Group. Simple Heuristics That Make Us Smart. Evolution and Cognition Series. Oxford University Press, 1999.

S. Gong and T. Xiang. Recognition of group activities using a dynamic probabilistic network. In IEEE International Conference on Computer Vision, pages 742–749, October 2003b.

S. S. Intille and A. F. Bobick. Recognizing planned, multiperson action. CVIU, 81(3): 414–445, March 2001.

X. Liu and C. S. Chua. Multi-agent activity recognition using observation decomposed hidden markov models. Image and Vision Computing, pages 166–175, 2006.

N. M. Oliver, B. Rosario, and A. P. Pentland. A Bayesian computer vision system for modeling human interactions. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8):831–843, 2000.

M. Perse, M. Kristan, J. Pers, and S. Kovacic. A template-based multi-player action recognition of the basketball game. In J. Pers and D. R. Magee, editors, ECCV Workshop on Computer Vision Based Analysis in Sport Environments, pages 71–82, Graz, Austria, May 2006.

L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. Readings in speech recognition, pages 267–296, 1990.

P. Ribeiro and J. Santos-Victor. Human activities recognition from video: modeling, feature selection and classification architecture. In Workshop on Human Activity Recognition and Modelling (HAREM 2005 - in conjunction with BMVC 2005), pages 61–70, Oxford, September 2005.

N. M. Robertson. Automatic Causal Reasoning for Video Surveillance. PhD thesis, Heartford College, University of Oxford, June 2006.

C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas. Conditional models for contextual human motion recognition. In International Conference on Computer Vision, 2005.

T. Van Vu, F. Bremond, and M. Thonnat. Automatic video interpretation: A recognition algorithm for temporal scenarios based on precompiled scenario models. In ICVS, 2003.

H. M. Wallach. Conditional random fields: An introduction. Technical report, University of Pennsylvania, 2004. CIS Technical Report MS-CIS-04-21.

Bo Wu and Ram Nevatia. Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors. International Journal of Computer Vision, 75(2):247–266, 2007.

Yilmaz, A., Javed, O., and Shah, M. (2006). Object tracking: A survey. ACM Comput. Surv. 38, 4 (Dec. 2006)