Recognition of coordinated multi agent activities, the individual vs the group

Scott Blunsden, Robert Fisher and Ernesto Andrade

Institute for Perception Action and Behaviour, School of Informatics, University of Edinburgh, Scotland
{s.j.blunsden, rbf, eaneto}@inf.ed.ac.uk
http://www.ipab.inf.ed.ac.uk/

Abstract. The problem of identifying coordinated group activity has received relatively little attention compared to the many approaches of recognising individual actions. Here a comparison between a global and individual modelling approach to classifying coordinated group activity is presented. Results show that by taking a global perspective to classifying coordinated activity in the sports domain better classification performance can be achieved within a relatively simple framework.

1 Introduction

The area of video sequence analysis with regard to automatic interpretation of human activity has recently received a lot of attention [1–4]. Traditionally researchers have been interested in identifying what an individual person is doing. Examples of this include gesture recognition [5] and understanding how a monitored area is utilised by an individual [6].

Within this paper the emphasis is upon trying to interpret what a group of people are doing. There are certain classes of activities that can only be defined when considering the interactions of multiple individuals. Examples of such activities include meeting, following and conversing. Although it may be possible to recognise such activities by only considering an individual's behaviour (such as are they standing still and talking), there are limitations to this approach. Within the remainder of the paper the focus is upon classifying what the group as a whole is doing.

Here the focus is upon classifying group behaviour within the sports domain. Specifically the game of European handball is used to investigate coordinated group behaviour. Within this paper two approaches to classifying group action are compared. One approach takes a bottom up approach to classification by modelling individuals. The other considers the group as a whole as forming the input signal. These two approaches are described in subsequent sections and the results are compared in the results section (6). By considering the group as a whole better results can be achieved when trying to classify the team activity.

2 Previous Work

The work most closely resembling the domain of interest here is that of Intillie and Bobick [1]. In [1] the goal was to classify pre-defined plays within the game of American football. They used predefined Bayesian networks to evaluate the likelihood of an observed play. Based on visual evidence a goal network was used to determine the belief that certain actions had been committed. This provides input into a multiagent network combining many agent's visual goal networks to compute the likelihood that a particular play was being observed.

In addition to the static Bayesian network architectures employed by [1] dynamic Bayesian networks have also been applied to multiagent problems. In [5] Oliver et al. use coupled hidden Markov models to model two person interactions. Again separate models are trained for each class of interaction, however there was no explicit a priori domain knowledge within the models. The likelihoods of observing a given interaction was determined from examples. Coupled models have also been employed by Xiang and Gong [3] who modelled vehicle interactions on an airport loading bay. All possible couplings of the model were enumerated and evaluated using the Bayesian information criterion [7] to determine the best connection architecture.

Howard and Jebara [2] use a hierarchical tree architecture to aggregate several persons' trajectories (again within the domain of American football) into a Markov process describing the team activity. Although they only present limited results the approach presents a plausible interpretation of the structure within the game. Each player is modelled, then the team, then the game as a whole. Others such as Zhang et al [8] have also used an aggregating approach for group action clustering in meetings. However in this instance a hierarchical architecture is not as explicit.

3 Data Preparation

The data set used throughout this paper is from the publicly available CVBASE dataset [9]. Only the handball sequences are used in the methods and results presented here. The handball dataset consists of 3 separate video cameras recording a 10 minute long game. Court coordinates of each of the players for one team throughout this sequence are available (first 1000 frames are shown in figure 1(a)). The activity the whole team is engaged in and the starting and end times of the activity is also annotated.

The timeout class from the original dataset was removed as this is not regarded as a coordinated activity. In addition some classes have been merged together. This step was primarily taken due to the high similarity between the activities themselves. The classes 'offence against set-up defence, setting up an offence'(nfpn) and 'offence against set-up defence, ending an offence' (nfan) were merged. These classes are highly similar in description and in appearance and require extra information about the game (ie the ability to detect an attempt on goal) to distinguish them. The classes 'defence, basic defence, against preparation of an offence'(obz) and 'defence, basic defence against ending offence'(obg) were merged for similar reasons. The classes 'defence returning' (ovpp) and 'defence slowly returning' (ovpc) were merged as the actual speed difference distinguishing each class was not well defined. These activities may have looked

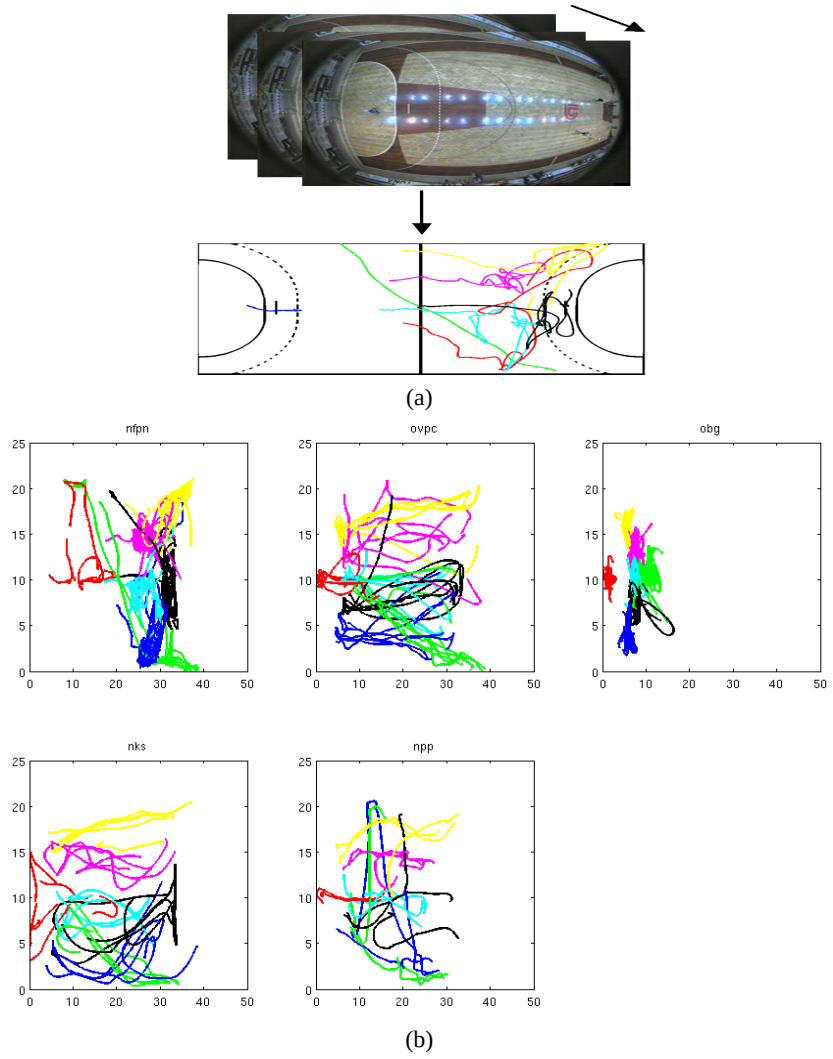


Fig. 1. (a) The first 1000 frames from the video sequence and the associated player positions as given in court coordinates. (b) Positional information of each player (different colour) plotted in court co-ordinates. The classes are: nfpn - offence against set-up defence , ovpc - defence, returning, obg - defence, basic defence, nks - offence, fast break, npp - offence, slowly going into offence.

different if information on the other team was also available, particularly in distinguishing between trying to stop a fast break. This gives 5 final classes shown in figure 1 (b).

4 Extracted Features

In addition to the original $x_{n,t}$ and $y_{n,t}$ court position provided in the original data set [9] (for the n^{th} player at the t^{th} timestep), three additional features were also calculated. A speed feature ($s_{n,t}$) was calculated for every player of every frame. The speed ($s_{n,t}$) of a player was calculated over the past w frames (equation 3). Such a feature helps to distinguish between fast and slow breaks. Here a temporal window size (w) is used due to low per frame movement. If the change in position was calculated based upon the previous frame only then frequently that change would be either zero or very close to zero.

The directional change of a player is also encoded in the feature vector with the direction of change in x ($c^x_{n,t}$ equation 1) and y ($c^y_{n,t}$ equation 2) being calculated, again over the past w frames. The window size (w) for calculating the change in position was empirically set to 100 frames (4 seconds) throughout all experiments. To prevent features from one complete activity sequence overlapping with that from another class (ie immediately after transitions) the first w frames from the sequence are removed. In addition to removing any overlap between features this also helps with eliminating the transitional period where one activity turns into another.

$$c^x_{n,t} = sign(x_{n,t-w} - x_{n,t}) \tag{1}$$
$$c^y_{n,t} = sign(y_{n,t-w} - y_{n,t}) \tag{2}$$
$$s_{n,t} = |(x_{n,t-w}, y_{n,t-w}) - (x_{n,t}, y_{n,t})| \tag{3}$$

This gives a final feature vector for the n^{th} player at the t^{th} timestep as given by equation (4).

$$\mathbf{p}_{n,t} = \left[x_{n,t}, y_{n,t}, c^x_{n,t}, c^y_{n,t}, s_{n,t}\right]^T \tag{4}$$

5 Classification

Each test/training sample was taken from the original data annotations and consists of a sequence of contiguous frames containing the calculated features as described in section 4. These sequences are taken directly from the annotated sequences. Within this classification scheme the minimum sample length varied due to the length of time a certain activity was being performed. This caused the number of available complete samples to vary between 55 and 12. For the training phase only the training samples were used and for the testing phase only the unseen test samples were used. The training and test samples did not overlap. The individual and group classification methods are now detailed and results are presented in section 6.

5.1 Individual modelling approach

Here each player is individually modelled as they evolve through time. Each individual model is then used as input into a higher level process which models the current activity of the team. The approach implemented here is that of Howard and Jebara [2]. This approach is chosen to contrast to the global view where all the players are considered as forming the input signal (section 5.2). The dynamical systems tree (DST) was chosen as it does not require an explicit domain model (cf. [1]) and is capable of generating a class label for a given sequence (cf. [3]). This hierarchical model also represents the problem well in that we hypothesise that each player forms input into a higher level process representing the team activity.

This particular DST consists of a top level aggregating process and leaf processes arranged in a hierarchical manner. An aggregating process is a Markov chain which has at most one parent process. For the problem represented here there is one aggregating process which represents the group activity. The states of the aggregating process states are represented by $s^a = \{s_0^a, .., s_T^a\}$, where T represents the maximum possible time. The activity aggregating process, without any parents, has the conditional distribution:

$$p(s^a) = p(s_0^a) \prod_{t=1}^{T} p(s_t^a | s_{t-1}^a) \tag{5}$$

Each player is modelled as a switching linear dynamical system (SLDS) and is referred to as a leaf process within this framework. A leaf process has an aggregating process as a parent and has the following conditional distribution:

$$\begin{aligned} p(s^i, x^i, y^i | s^{\pi(i)}) = {} & p(s_0^i | s^{\pi(i)}) p(x_0^i | s_0^i) p(y_0^i | x_0^i) \\ & \times \prod_{t=1}^{T} p(s_t^i | s_{t-1}^i, s_t^{\pi(i)}) p(x_t^i | x_{t-1}^i, s_t^i) p(y_t^i | x_t^i) \end{aligned} \tag{6}$$

In this conditional distribution $s^i = \{s_0^i, .., s_T^i\}$ represents the ith leaf processes discrete Markovian hidden state. $x^i = \{x_0^i, .., x_T^i\}$ is the continuous Markov hidden state, with $y_i = \{y_0^i, .., y_T^i\}$ being the observations of the ith leaf process. The leaf processes parent process is given by $\pi(i)$ with discrete hidden states $s^{\pi(i)} = \{s_0^{\pi(i)}, .., s_T^{\pi(i)}\}$.

This gives the conditional distribution over all variables in the DST with $\mathcal{A}$ aggregating processes and $\mathcal{L}$ leaf processes as:

$$P(\mathcal{S}, \mathcal{X}, \mathcal{Y}) = \prod_{a \in \mathcal{A}} p(s^a) \prod_{i \in \mathcal{L}} p(s^i, x^i, y^i | s^{\pi(i)}) \tag{7}$$

where $\mathcal{S}, \mathcal{X}, \mathcal{Y}$ represents the discrete hidden, continuous hidden and emission variables, respectively. The variational estimates of the parameters of the model are not detailed here due to lack of space, the equations are given in [2]. Input to each of the leaf processes at a particular timestep is given by equation (4).

The DST was set up so that each leaf node represents a player with an aggregating process modelling the overall team activity. For the purposes of classification a DST tree

(with the same structure) is trained on each class of activity (fig 1) using the training set. These trained models were then used to compute the likelihood for an unknown play. The trained model which produced the highest (bounded) likelihood was taken to be the class of the novel sample.

5.2 Group classification

In contrast to modelling each individual player the input vector is now made up of *all* the players in the team. That is the input signal per frame is given by equation (8). All the player's attributes are concatenated to form one 35 dimensional vector as input per frame. A different classification scheme is used as the DST approach is not suited to high dimensional single signal data as is used here.

$$\mathbf{q}_t = [\mathbf{p_{1,t}}, .., \mathbf{p_{N,t}}] \tag{8}$$

Here $\mathbf{p}_{n,t}$ is as defined in equation (4) and N is the size of the team (in this case N=7). A support vector machine (SVM) classifier [10] is then trained upon this data. Partitioning of training and testing data followed the same convention as in the individual case. The SVM classifier is briefly reviewed here before describing its application to the sports dataset. The SVM classifier was chosen after examining the eigenvector projections of the data (figure 4). The data is well separated in the space per class but not clustered around any obvious points. The decision boundaries between the classes are also non-linear, for this reason a SVM classification scheme was deemed an appropriate classifier to use.

A SVM classifier seeks to find a separating (hyper) plane such that the distance between the plane and each data point is maximised. This distance is given by the margin which is defined as being the minimum perpendicular distance from the plane to a data point. A point is then classified by the discriminant function:

$$f(\mathbf{x}) = sgn(\sum_{i}^{N} y_i \lambda_i . k(\mathbf{x}, \mathbf{x}_i) + b) \tag{9}$$

Within this equation $\mathbf{x}_i$ is the i^{th} datapoint with $y_i \in \{+1, -1\}$ being the corresponding class label and λ_i the Lagrangian term associated with the i^{th} data point. When the data point ($\mathbf{x}_i$) has a non-zero Lagrangian (λ_i) value it is considered a support vector (ie on the margin). The term b is a constant scalar representing the offset from the origin. The kernel function is represented by k. Throughout these experiments a Gaussian kernel function is used. The problem is to maximise :

$$W(\lambda) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i=1}^{N} \lambda_i \lambda_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \tag{10}$$

subject to $0 \leq \lambda_i \leq C, i = 1, ..., N$ and $\sum_{i=1}^{N} \lambda_i y_i = 0$

The constant C places an upper bound on the Lagrange parameters, thereby limiting the influence of individual patterns. In experiments presented here $C = 10$ as good

classification performance was obtained with this setting. Although using pairwise classification strategies have been shown to be preferable in some situations [11] the one against all method proved to work well for this dataset.

The SVM classifier is trained using individual points from $\mathbf{q}_t$ (equation (8)). Each frame in the training set was assigned an activity label corresponding to the class of the complete sequence. This effectively removed the temporal ordering of the data (which is an area of future research). The publicly available Torch libraries [12] were used to compute the results.

To classify a test sequence each point in that sequence is individually labelled by the SVM classifier. This produces a vector of tokens for each (variable length) sample ($\mathcal{L} = [l_1 = c_1, .., l_t = c_1 .., l_{samplen} = c_i]$). Here l_t represents the label at time t and is an element from the set of all possible classes as determined by the SVM classifier. To classify the entire sample from the labelled vector the most frequent label in this labelling is taken to be the label of the whole sample.

6 Results

To generate classification results each method was run for a total of 100 runs. On each run samples were randomly partitioned into non overlapping complete training and test activity sequences using a 50-50 split. The testing data was unseen until classification. The minimum size of sequence length was also increased from 100 frames to 275 frames in increments of 25 frames. Once the minimum size was increased beyond 275 frames the number of complete sequences dropped to 12, which was deemed too small a subset for testing and training. The poor classification results when fewer samples are available can be seen in figure 2.

Figure 2 shows the results of both the DST classification and the SVM group classification. Both methods attain peak performance when the minimum window size is around 150-250 frames in size. This indicates that there may be a time scale implicit in the game which has to be passed in order for decisions to be made about what activity the team is undertaking.

It is immediately visible that the SVM group classifier performs significantly better than the DST classifier on this problem. Both the averaged classification performance is higher and the standard deviation is lower. This indicates the DST model was not able to correctly fit the data as well.

In addition to the classification of pre-segmented sequences a trained model was run over the whole video sequence (all 15000 frames). Models were trained on a randomly selected sample of complete activity sequences. The number of training sequences accounted for less than 50% of the total video length. This random sampling approach was taken in preference to simply dividing the complete video in half as the distribution of activities is not uniform and some only occur in the second half of the game.

The models were trained using the procedures described in the preceding sections. As this process results in a continuous labelling of the complete sequence the time out class was also used for training. For labelling an individual frame the past n frames were used as input into the classifier. Throughout the experiments n was set to 100.

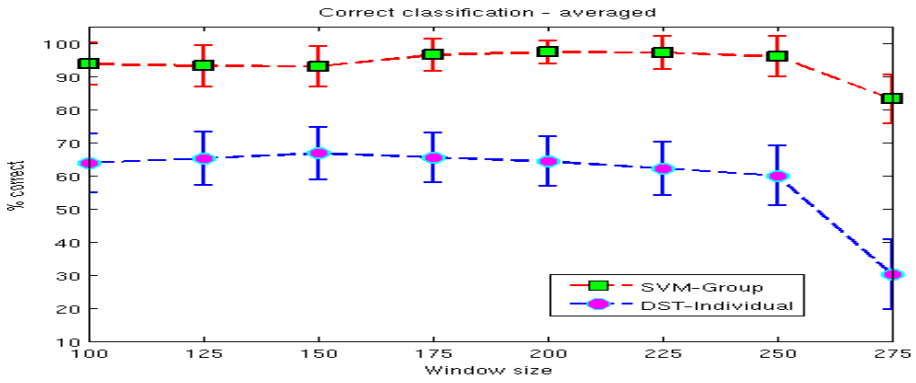


Fig. 2. Correct classification rates for group classification using support vector machines(squares) and dynamical systems tree (circle)

This produces a labelling for every frame in the sequence (after n frames have passed). The results are shown in figure (3).

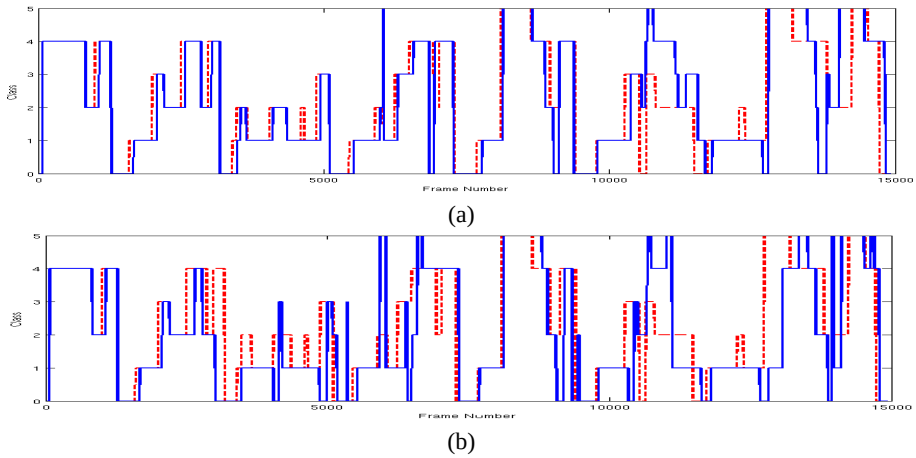


Fig. 3. Classifications for every frame in the handball sequence. Dotted line denotes correct (ground truth) class. Continuous line denotes label determined by the classifier. (a) Labelling as given by the SVM-group classifier. (b) Labelling as given by the DST classifier.

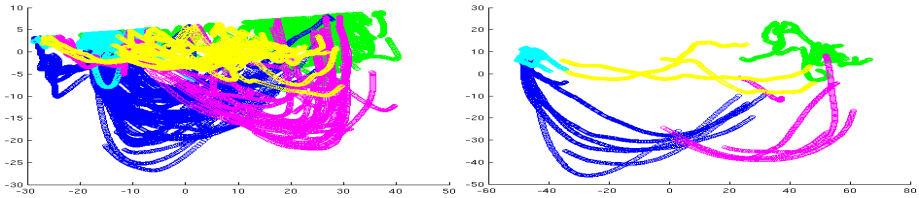
It is clearly visible that there is some delay in making a correct classification when the activity switches in both the SVM-group and the DST classifiers. The overall correct classification rate when using the SVM-group classifier is 72.3%, with the DST giving a performance of 61.9%. When looking at the plot of correct and prediced there is a high correlation, with most of the misclassifications being due to delaying the deci-

sion or between frames 10000-12000. The team activity changes fairly rapidly in quick succession during this time period, which may explain the poor performance.

7 Conclusion

The higher complexity of the DST model coupled with the relatively sparse complete samples may explain its inferior performance. When looking at the predicted labels of the model its most frequent predictions seem to be from models which have more examples, and are thus better trained. The model seems capable of discriminating between offence and defence but has trouble classifying the specific type of offence/defence (due to their high similarity) and frequently chooses the one with more examples (in the original paper [2] the low number of samples was not such a problem). Also the relative simplicity of the SVM classifier means training and classification can be done in under a minute where as a DST can take several hours to train and classify the data.

A rationale behind the superior performance of taking a global view may be due to the data itself. Figure 4 shows PCA projections of the data for both (a) individual and (b) group input. It is clear that when one considers the whole group the data shows more interclass overlap (a). When taking the global view (b) for co-ordinated group activities in this case superior results can be achieved with much simpler methods. Indeed the simpler method reduces the number of parameters which are required to model the problem. Reducing the learning of such parameters is particularly important when dealing with sparse datasets which are common to vision problems due to the high overhead of acquiring data.



(a) (b)

Fig. 4. PCA projections using the 2 highest eigenvectors for (a) Individual and (b) group datapoints. (a) shows individual players whilst (b) shows the team as a whole (35 Dim vector). Each colour refers to a different class. The data in (b) is significantly more separable

When applying these methods to a more realistic scenario with unsegmented activities both methods suffer. However it is clear that they show a high correlation with the ground truth data (figure 3).

8 Future Work

When taking a global view as in this instance the question of roles arises. Within this dataset players stick closely to their position with no significant swapping of position

(ie attacker and defender swap roles). This situation causes problems not just with the SVM global classifier but also with the DST classifier or any which assumes constant roles throughout training and testing. This problem would need to be addressed to make such methods more general and able to understand more complex sequences and games.

Future work could address delaying the decision or recognising transitions are occurring when labelling the unsegmented data. Also more data (such as another game) would be a better test of the algorithms generalisation capabilities.

The question of how much extra information the other team could give us is also the focus of future work. Certain situations such as preventing a fast break or a player seemingly going out of position may become clearer if information for the other team was available. Extra information is also available in the form of temporal dependencies from frame to frame. At present the group classification approach does not incorporate such dependencies into the classification procedure. Our future work will address these issues in more detail.

9 Acknowledgements

We would like to thank Andrew Howard for kindly making his code available.

References

1. Intille, S.S., Bobick, A.F.: A framework for recognizing multi-agent action from visual evidence. MIT Media Laboratory (1999)
2. Howard, A., Jebara, T.: Dynamical systems trees. In: Uncertainty in Artificial Intelligence. (2004)
3. Gong, S., Xiang, T.: Recognition of group activities using a dynamic probabilistic network. In: IEEE International Conference on Computer Vision. (2003) 742–749
4. Jug, M., Pers, J., Dezman, B., Kovacic, S.: Trajectory based assessment of coordinated human activity. In: Computer Vision Systems, Proceedings of Third International Conference ICVS. (2003) 534–543
5. Oliver, N.M., Rosario, B., Pentland, A.P.: A bayesian computer vision system for modelling human interactions. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 831–843
6. Dee, H.M., Hogg, D.C.: Detecting inexplicable behaviour. In: British Machine Vision Conference. (2004) 477–486
7. Schwarz, G.: Estimating the dimension of a model. Annals of Statistics **6** (1978) 461–464
8. Zhang, D., Gatica-Perez, D., Bengio, S., McCowan, I., Lathoud, G.: Modelling individual and group actions in meetings: A two-layer hmm framework. IDIAP Research Institute, Martigny, Switzerland (2000)
9. Machine Vision Group, U.o.L.: Cvbase '06 workshop on computer vision based analysis in sport environments, found at url: http://vision.fe.uni-lj.si/cvbase06/ (2006)
10. Scholkopf, B.: Statistical learning and kernel machines. Technical report, Microsoft Research Cambridge, Guildhall Street, Cambridge CB2 3NH, UK (2000) Technical report.
11. Duan, K., Keerthi, S.S.: Which is the best multiclass svm method? an empirical study. In: Neural Information Processing Systems. (2003)
12. Collobert, R., Bengio, S., Marithoz, J.: Torch: a modular machine learning software library. Technical report, IDIAP (2002) Technical report IDIAP-RR 02-46.