Hidden Markov Models with Kernel Density Estimation of Emission Probabilities and their Use in Activity Recognition

Massimo Piccardi
Faculty of Information Technology
University of Technology, Sydney
massimo@it.uts.edu.au

Óscar Pérez
Computer Sciencie Department - GIAA
Universidad Carlos III de Madrid,
Colmenarejo, Spain
oscar.perez.concha@uc3m.es

Abstract

In this paper, we present a modified hidden Markov model with emission probabilities modelled by kernel density estimation and its use for activity recognition in videos. In the proposed approach, kernel density estimation of the emission probabilities is operated simultaneously with that of all the other model parameters by an adapted Baum-Welch algorithm. This allows us to retain maximum-likelihood estimation while overcoming the known limitations of mixture of Gaussians in modelling certain probability distributions. Experiments on activity recognition have been performed on ground-truthed data from the CAVIAR video surveillance database and reported in the paper. The error on the training and validation sets with kernel density estimation remains around 14-16% while for the conventional Gaussian mixture approach varies between 15 and 24%, strongly depending on the initial values chosen for the parameters. Overall, kernel density estimation proves capable of providing more flexible modelling of the emission probabilities and, unlike Gaussian mixtures, does not suffer from being highly parametric and of difficult initialisation.

1. Introduction

Automatic recognition of activities in videos is paramount to many applications such as multimedia annotation and visual surveillance. As a consequence, it has been widely investigated to date (see [1-5] and several others). Activities are often modelled as the states of entities, either at given times or along time intervals. For instance, the state of a person at a given frame may be recognised as either "inactive" or "active". By considering sequences of state values, we may want to recognise more complex patterns such as "raising an arm" or "collecting an object". In many cases, the values of the state variables cannot be directly measured, due to noise and other non-idealities such as occlusions and illumination changes, and have to be estimated from the available observations. Hidden Markov models and their several variations have been used extensively for activity recognition since they provide a smoothed estimate of the state values [6,7,3]. It may be argued that smoothed state estimators introduce a delay between an observation and the corresponding state estimate: however, such a delay is often negligible to the purpose of the application. In particular, this is the common case for video analysis where observations occur at video rate and even a delay of a few hundred observations translates into a relatively short time delay.

A hidden Markov model (HMM) is fully described by three sets of quantities: the state transition probabilities, A; the emission (or observation) probabilities, B; and the probabilities of the initial states, π. The states are only allowed to assume discrete values; let us say, N. Therefore, A can be represented by an N x N matrix and π by an N-dimensional vector. Instead, observations are often drawn from continuous variables and, as such, emission probabilities need to be modelled by probability density functions. The most common approach for modelling emission probabilities is by use of mixtures of Gaussians [6]. In such a case, B is fully described by the weights, means and covariances of all the Gaussian components. Once given one or more sequences of observations, the Baum-Welch algorithm can be used for learning a corresponding HMM with maximum likelihood. This algorithm is an expectation-maximization algorithm that learns A, B and π in a simultaneous manner and is guaranteed to converge to a local optimum in the parameter space.

Alternatives to the use of mixtures of Gaussians (GMs) for modelling the emission probabilities have been proposed in the literature. Bourlard and Morgan in [8] proposed to replace the Gaussian mixtures by Artificial Neural Networks in a hybrid ANN/HMM model. A number of variations on hybrid ANN/HMM models is presented by Trentin in [9]. In a recent work [10], Krüger *et al*. proposed to replace the Gaussian mixtures with mixtures of Support Vector Machines. However, these approaches typically train the emission probabilities in a supervised manner, requiring knowledge of the ground-truth values of the hidden state variable. Even though they

1-4244-1180-7/07/$25.00 ©2007 IEEE

may potentially achieve higher accuracy than maximum-likelihood methods, they cannot be applied in the general case where state ground-truth is not available. Conversely, maximum-likelihood HMMs can be trained just with a sequence of observations that are, by definition, available and therefore we focus our attention on them in the following. In our work, we want to retain the maximum-likelihood simultaneous estimation of all parameters offered by the Baum-Welch algorithm, while overcoming the known limitations of mixture of Gaussians. Gaussian mixtures suffer from limitations in modelling pdf's in at least two well-known circumstances: a) when the number of modes in the pdf is greater than that of the Gaussian components in the mixture, b) when the pdf has uniform regions. Kernel density estimation (KDE) has generally proven superior to GMs in these cases. On the other, hand the estimation of the optimal *kernel bandwidth* in KDE is extremely critical for its performance. Several criteria for optimality and corresponding methods have been proposed to this aim [11].

In this paper, we present a model for the emission probabilities based on KDE that can be directly plugged in the Baum-Welch algorithm (referred to as KDE/HMM in the following). For estimation of the kernel bandwidth, we propose an expectation-maximization algorithm under a maximum pseudo-likelihood criterion. Experiments are performed on videos from the CAVIAR database and accuracy is measured against the state ground truth provided by expert annotation [12]. The experimental results show the improved performance of KDE/HMM over conventional HMMs based on GMs. While this paper limits its analysis to video data, the proposed KDE/HMM lends itself to general application and can improve performance in other domains. Experimental results on synthetic data omitted from this paper due to space limitations reassure in this direction.

The rest of the paper is organised as follows: Section 2 describes kernel density estimation by expectation-maximization and compares modelling of pdf's with that provided by a mixture of Gaussians with a fixed number of components. Section 3 extends the kernel density estimation to the modelling of emission probabilities in HMMs. Section 4 presents the experiments performed and discusses results. The conclusions summarise the main contributions of this work.

2. Kernel density estimation with an expectation-maximization algorithm

Gaussian mixtures can be used to model the probability density function of a random variable, $p(x)$, as:

$$p(x) = \sum_{l=1}^{M} \alpha_l \, G\left(x; \mu_l, \sigma_l^2\right) \tag{1}$$

with $G(\cdot)$ the Gaussian function and M the number of Gaussian components. We consider here the univariate case for the sake of simplicity of notation, but we will eventually extend results to the multivariate case. The parameterization of such a GM requires the estimate of the weight, α, mean, μ, and variance, σ^2, for each of the M Gaussian components (the sum of weights has to be unitary). This estimate is typically performed so as to maximize the likelihood, L, over a set of samples, x_i, $i=1,...,N$, independently drawn from this distribution:

$$L(x_1,..,x_N) = \prod_{i=1}^{N} p(x_i). \tag{2}$$

Operationally, the likelihood is conveniently computed in log form with the main advantage of avoiding rapid underflow. Maximization of (2) can be obtained by estimating the GM parameters through an expectation-maximization (EM) algorithm. EM is an iterative algorithm that improves the estimate of the parameters at each iteration and is guaranteed to converge to a local maximum of the likelihood (or a saddle point in the multivariate case). The equations used to estimate the parameters at each iterations are:

$$\alpha_l^{new} = \frac{1}{N} \sum_{i=1}^{N} p(l \mid x_i, \Theta) \tag{3}$$

$$\mu_l^{new} = \frac{\sum_{i=1}^{N} x_i \, p(l \mid x_i, \Theta)}{\sum_{i=1}^{N} p(l \mid x_i, \Theta)} \tag{4}$$

$$\sigma_l^{2\,new} = \frac{\sum_{i=1}^{N} \left(x_i - \mu_l^{new}\right)^2 p(l \mid x_i, \Theta)}{\sum_{i=1}^{N} p(l \mid x_i, \Theta)} \tag{5}$$

where Θ represents the current set of parameters and $p(l|x_i,\Theta)$ is simply the probability of the l-th Gaussian component at sample x_i:

$$p(l \mid x_i, \Theta) = \frac{\alpha_l \, G\left(x_i; \mu_l, \sigma_l^2\right)}{\sum_{k=1}^{M} \alpha_k \, G\left(x_i; \mu_k, \sigma_k^2\right)} \tag{6}$$

Equation (6) is often referred to as a membership function, expressing the membership of x_i in each of the Gaussian components, and its computation is the expectation step of

an EM iteration. The computation of update equations (3-5) is its maximization step. It is important to note that each of (3-5) provides an optimum for the respective parameter independently of the other two.

Kernel density estimation models a pdf as:

$$p_{KDE}(x) = \frac{1}{Nh}\sum_{j=1}^{N} K\left(\frac{x - x_j}{h}\right) \quad (7)$$

where $K(\cdot)$ is a function with particular properties, called *kernel*, and h is the kernel bandwidth (or smoothing factor). By using the Gaussian kernel and noting the kernel bandwidth with σ, (7) becomes:

$$p_{KDE}(x) = \frac{1}{N}\sum_{j=1}^{N} G\left(x; x_j, \sigma^2\right) \quad . \quad (8)$$

Although (8) reduces KDE to another Gaussian mixture, the similarity between (1) and (8) is mainly apparent: first, in (1) the number of Gaussian components, M, is typically very low compared to the number of samples, N. Moreover, the weight, position and width (α, μ, σ) of each component are determined as a trade-off over the sample set. In (8), instead, each Gaussian component is firmly located on a sample. The only parameter to be estimated is the variance, σ^2 (or, equivalently, the standard deviation, σ), common to all the Gaussian components.

Estimate of the optimal variance, σ^2, can be performed according to a number of different criteria (see [11] for a comprehensive review). It is interesting to note that maximizing the likelihood for the KDE case leads to an obvious but impractical solution:

$$\begin{aligned} L_{KDE}(x_1,..,x_N) &= \prod_{i=1}^{N} p_{KDE}(x_i) = \\ &= \prod_{i=1}^{N}\left(\frac{1}{N}\sum_{j=1}^{N} G\left(x_i; x_j, \sigma^2\right)\right) \to \infty \; as \; \sigma \to 0 \end{aligned} \quad (9)$$

While several criteria could be chosen to determine an optimal value for σ, here we are interested in retaining the maximum likelihood framework so that our results can be more easily transferred to the estimate of parameters of a hidden Markov model. Thus, we use the pseudo-likelihood defined as [11]:

$$PL_{KDE}(x_1,..,x_N) = \prod_{i=1}^{N}\frac{1}{N}\sum_{j=1, j\neq x_i}^{N} G\left(x_i; x_j, \sigma^2\right) . \quad (10)$$

Essentially, the probability of each x_i sample is computed by excluding the Gaussian component centred on x_i itself. Maximization of (10) can be performed in various ways. Duin in [13] suggested computing the first derivative of (10) with respect to σ and iteratively calculating its zero crossings. He reported that, by using a specially adapted version of the regula falsi algorithm, 5-20 iterations were needed to reach an accuracy of 10^{-3} in the value of σ over a set of experiments. Here, we use the expectation-maximization algorithm by adapting update equation (5) to the case of a common value for σ for all the Gaussian components. From [7], it can be easily proven that:

$$\sigma^{2\,new} = \frac{\sum_{l=1}^{M}\left(\sum_{i=1}^{N}\left(x_i - \mu_l^{new}\right)^2 p(l \mid x_i, \Theta)\right)}{\sum_{l=1}^{M}\left(\sum_{i=1}^{N} p(l \mid x_i, \Theta)\right)} \quad (11)$$

provides the optimal value for σ when such a value is constrained to be the same for all the M Gaussian components. Again, if we use (11) directly for estimating σ in the KDE case, we will converge to the undesired value $\sigma = 0$. Thus, for KDE we adjust (11) to reflect the definition of pseudo-likelihood given in (10) as:

$$\sigma^{2\,new} = \frac{\sum_{i=1}^{N}\sum_{j=1, x_j\neq x_i}^{N}\left(x_i - x_j\right)^2 p(j \mid x_i, \Theta)}{\sum_{i=1}^{N}\sum_{j=1, x_j\neq x_i}^{N} p(j \mid x_i, \Theta)} \quad . \quad (12)$$

Equation (12) can be derived as follows: for GM, the derivation of (3-5) is possible under the simplifying assumption that the generative model of each sample x_i is not the whole GM, but only the best Gaussian component indicated by an unobserved indicator variable. For KDE, under the further assumption of pseudo maximum likelihood, the probability at (6) is assumed null when $x_j = x_i$; thus, (12) derives from (11). For testing, we have run a set of experiments on a range of simulated data: convergence of σ was always obtained, and always to a local maximum of the pseudo-likelihood.

2.1. Comparing KDE and GM pdf estimation

GMs show limitations in modelling certain distributions. One limitation is in the modelling of distributions which show more modes than the Gaussian components. In this case, one single Gaussian component has to be fit over multiple modes, thus leading to poor modelling based on eye judgment and relatively low likelihood. Although estimating the "right" number of modes is possible through procedures such as the mean-shift vector, it is often unfeasibly time consuming. The same poor modelling occurs when the distribution shows uniform regions which

are inaccurately modelled by means of only a few Gaussians. KDE can overcome both these limitations. We argue that in some cases feature values obtained from human activities in videos such as speed and positions exhibit such uniform regions. Moreover, we argue that KDE could also lead to improved hidden Markov models of such activities. Figures 1 and 2 show an example of density estimation with a GM with two components based on (3-6) and with KDE based on our estimator for σ. For the latter case, the fitting of the distribution over the samples seems generally very good. As an obvious consequence, the likelihood obtained for KDE has always been greater or equal than that for GM in all our experiments. Obviously, this comes at an increased computational cost for the evaluation of (8) with respect to (1).

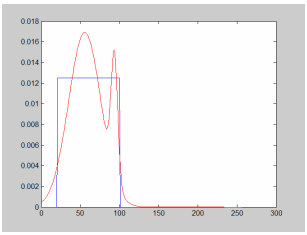


Figure 1: GM density estimation of a seemingly uniform distribution. Estimation seems generally inaccurate (initial parameters: $\alpha_1 = \alpha_2 = 0.5$; $\mu_1 = 85$, $\mu_2 = 170$; $\sigma_1 = \sigma_2 = 16$).

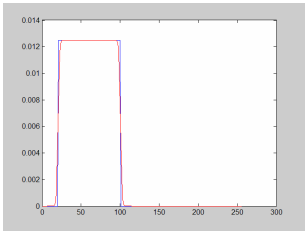


Figure 2: KDE from the same set of samples as Figure 1 with the proposed estimator for σ. Estimation seems generally accurate (initial parameter: $\sigma = 16$).

3. Hidden Markov models with kernel density estimation of emission probabilities

A hidden Markov model (HMM) offers a mean to estimate the joint probability of a sequence of time-discrete observations O_t, $t = 1..T$, and corresponding hidden states, $X_t \in \{1..N\}$ [7]. The model is fully described by the set of parameters $\lambda = \{A, B, \pi\}$:

$$A = \{a_{ij}\} = p(X_t = i \mid X_{t-1} = j) \;\; \forall i, j \tag{13}$$

$$B = \{b_j(o_t)\} = p(O_t = o_t \mid X_t = j) \;\; \forall o_t, j \tag{14}$$

$$\pi = \{\pi_i\} = p(X_1 = i) \;\; \forall i \tag{15}$$

A are called the state transition probabilities and express the Markovian hypothesis that the value, i, of the current state, X_t, is only dependent on the value, j, of the previous state, X_{t-1}. B are called the emission (or observation) probabilities, quantifying the probability of observing value o_t when the current state is j. Eventually, π are called the initial state probabilities and quantify the probabilities of values for the initial state. When observation values are continuous, HMMs typically use GMs to model their emission probabilities. Each state's value, $i = 1..N$, has a corresponding GM. Thus, the observation probability $b_i(o_t)$ is given by:

$$b_i(o_t) = \sum_{l=1}^{M} \alpha_{il} G\left(o_t; \mu_{il}, \sigma_{il}^2\right) . \tag{16}$$

The Baum-Welch algorithm provides update equations for the iterative optimisation of the model's parameters. Herewith, we concentrate on B. First, similarly to (6), we pose:

$$p_i(l \mid o_t, \Theta) = \frac{\alpha_{il} G\left(o_t; \mu_{il}, \sigma_{il}^2\right)}{\sum_{k=1}^{M} \alpha_{ik} G\left(o_t; \mu_{ik}, \sigma_{ik}^2\right)} . \tag{17}$$

to express the probability of the l-th component in the GM of state i. Then, the weights, means and variances of the emission probabilities are obtained in a way similar to (3-5) over the set of observed values, o_t, $t = 1..T$. The basic difference is that the terms in the numerators and denominators are multiplied by the probability of being in state i at time t, $\gamma_i(t)$:

$$\alpha_{il}^{new} = \frac{\sum_{t=1}^{T} p_i(l \mid o_t, \Theta)\, \gamma_i(t)}{\sum_{t=1}^{T} \gamma_i(t)} \tag{18}$$

$$\mu_{il}^{new} = \frac{\sum_{t=1}^{T} o_t \; p_i(l \mid o_t, \Theta)\, \gamma_i(t)}{\sum_{t=1}^{T} p(l \mid o_t, \Theta)\, \gamma_i(t)} \tag{19}$$

$$\sigma_{il}^{2\,new} = \frac{\sum_{t=1}^{T} \left(o_t - \mu_{il}^{new}\right)^2 \; p_i(l \mid o_t, \Theta)\, \gamma_i(t)}{\sum_{t=1}^{T} p(l \mid o_t, \Theta)\, \gamma_i(t)} \tag{20}$$

In turn, $\gamma_i(t)$ can be expressed from the current estimates of A and B (see [7] for details).

From (18-20) and the considerations addressed in Section 2, we can finally derive the update equations for the KDE case:

$$\alpha_{il}^{new} = \frac{\sum_{t=1, o_t \neq o_l}^{T} p_i(l \mid o_t, \Theta)\, \gamma_i(t)}{\sum_{t=1}^{T} \gamma_i(t)} \tag{21}$$

$$\mu_{il}^{new} = o_l \tag{22}$$

$$\sigma_i^{2\,new} = \frac{\sum_{t=1}^{T} \sum_{l=1, o_l \neq o_t}^{T} (o_t - o_l)^2\; p_i(l \mid o_t, \Theta)\, \gamma_i(t)}{\sum_{t=1}^{T} \sum_{l=1, o_t \neq o_t}^{T} p(l \mid o_t, \Theta)\, \gamma_i(t)} \tag{23}$$

In (22), the centres of the Gaussian components are not subject to update and sit, as usual, on the samples. (23) is the re-writing of (12) integrated by $\gamma_i(t)$. Again, we exclude the Gaussian component centred on the sample itself to prevent convergence to $\sigma^2 = 0$. We conveniently obtain this by setting $p_i(l \mid o_l, \Theta) = 0$ at the beginning of the iteration. Weight adjustment is needed also in the KDE case since observations need to be "dispatched" to the states in any case. To this aim, (21) is identical to (18) and just follows the way EM updates the GM weights. The only difference in (21) is that $p_i(l \mid o_l, \Theta)$ is, again, set equal to 0. In this way, the weights are essentially defined by the neighbouring kernels, not the one centred on the point itself, like in update equation (23). Alternatives for weight assignment are possible, such as a simple $\alpha_{il} = \gamma_i(l)$, but they have not been experimented in real data, just in the synthetic data showed in the section 4.1. Overall, equations at (21-23) define the KDE/HMM proposed in this paper. Merely to prove that these results obviously extend to the multivariate case, we conclude this section by showing (23) for the case of multivariate observations:

$$\Sigma_i^{new} = \frac{\sum_{t=1}^{T} \sum_{l=1, o_l \neq o_t}^{T} (o_t - o_l)(o_t - o_l)^T\; p_i(l \mid o_t, \Theta)\, \gamma_i(t)}{\sum_{t=1}^{T} \sum_{l=1, o_t \neq o_t}^{T} p(l \mid o_t, \Theta)\, \gamma_i(t)} \tag{24}$$

4. Experiments

In this section, we report results divided in two sets of experiments. First of all, in order to show the performance of the KDE/HMM, we present some results with synthetic data and the weight assignment as simple as simple $\alpha_{il} = \gamma_i(l)$. Subsequently, we carried out some tests for the human activity classification with the well known database of CAVIAR [12].

4.1. Experiments with synthetic data

The first set of experiments consists of a synthetic data distributed in three clusters of two-dimensional data:

- Class 1: A two-dimensional uniform function between *x, y* between 0 and 18.
- Class 2: Four two-dimensional independent Gaussians functions between *x, y* = 21 and 26 and σ=*2*.
- Class 3: Another two-dimensional uniform between *x, y* between 30 and 60

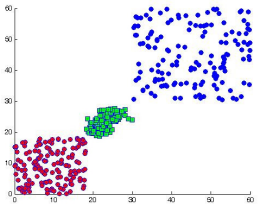


Figure 3. Distribution of the training data.

The initialization of the parameters for the KDE and GMs are set as follows:

- Means (only for the GMs case): μ_1 = [10 23 50]
- Covariance: $\Sigma_1 = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix}$, $\Sigma_2 = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix}$, $\Sigma_3 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, $\Sigma_4 = \begin{pmatrix} 2.5 & 0 \\ 0 & 2.5 \end{pmatrix}$, $\Sigma_5 = \begin{pmatrix} 5 & 0 \\ 0 & 5 \end{pmatrix}$, $\Sigma_6 = \begin{pmatrix} 10 & 0 \\ 0 & 10 \end{pmatrix}$
- Weight for each of the Gaussian components: random in the case of GMs, and 1 for KDE.
- Number of Gaussians per state (only GMs case): $M = 2$.
- State transition matrix (3 x 3): A = [0.6 0.2 0.2; 0.2 0.6 0.2; 0.2 0.2 0.6].
- Initial probabilities: Π = [1 0 0]
- Maximum number of iterations for the EM algorithm: *max-iter=50*

The experiments are carried out by changing the value of the initial covariance and using 2-fold cross validation. We took 150 points for the first and third class and 160 for the second one.

Table 1. Total classification error for the training and validation sets of synthetic data.

Error(%)		KDE	GMs
(1) Σ_1	Training	0.0	67.4
	Validation	0.0	67.4
(2) Σ_2	Training	0.0	67.4
	Validation	0.0	67.4
(3) Σ_3	Training	0.0	43.7
	Validation	0.0	46.3
(4) Σ_4	Training	2.4	43.7
	Validation	0.0	46.3
(5) Σ_5	Training	0.0	43.7
	Validation	0.0	46.3
(6) Σ_6	Training	0.0	43.7
	Validation	0.0	46.3

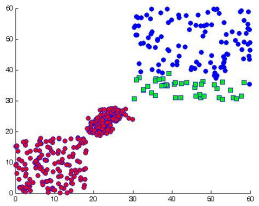


Figure 4. Picture of the classification and confusion matrix after the training for experiments from 3 to 6 with GMs.

We can check that the KDE outperforms in all the experiments the GMs method by modeling perfectly the two uniforms and the group of Gaussians. On the other hand, the GMs improves its performance with covariance values higher than 2, whereas the classification is very poor below this threshold.

Table 2. Confusion matrix for the classification with the GMs (experiments from 3 to 6)

GMs	Predicted			
Actual		**1**	**2**	**3**
	1	150	0	0
	2	160	0	0
	3	0	41	109

4.2. Experiments for the human activity recognition

Finally, we report results on the application of KDE/HMM to the classification of human activities. We used the CAVIAR video dataset [12] and selected the two videos named Fight_RunAway1.mpg and Fight_OneManDown.mpg. Among all the activities showed in these videos, we focused on three: {Inactive ("**in**"), Walking ("**wk**") and Running ("**r**")}. Both videos come accompanied by the ground truth provided by the dataset's authors. Each person in each frame is labelled with an activity value. This ground truth was determined by hand-labelling and we must take into account the subjectivity when classifying, especially between classes walking and running. The tool used for the experiments was the Kevin Murphy's Matlab Toolbox for HMM [14]. We later modified this toolbox to add the implementation of the KDE model.

The features that we selected in order to classify the activities are the magnitudes of the subject's speed measured over different time intervals. In particular, we computed the speed at 5 and 25 frame intervals as follows:

$$speed_f = \frac{1}{f}\sqrt{(x_i - x_{i-f})^2 + (y_i - y_{i-f})^2}\ , \qquad (25)$$

where f is set to 5 and 25, respectively, and (x_i, y_i) and (x_{i-f}, y_{i-f}) are the subject's positions in the image plane.

Figure 3 shows histograms of the two velocities for the three activities and how challenging the separation of the activities promises to be based on such features.

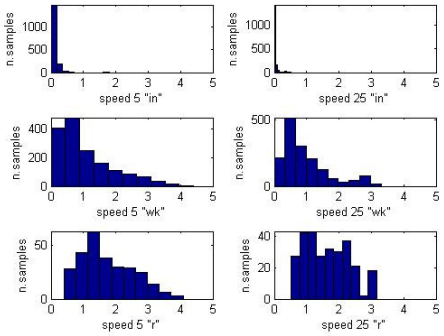


Figure 5: Histograms of features $speed_5$ (left column) and $speed_{25}$ (right column) for states "in" (inactive), "wk" (walking) and "r" (running).

The experiments consisted of learning the HMM parameters for both models, and their subsequent use for activity classification, i.e. Viterbi state decoding. The prior probability was fixed to {state 1 = 1; state 2 = 0; state 3 = 0} as we assume that the state of an individual first appearing in the scene is always "inactive". This assignment results very useful when decoding the sequence, as we do not know a priori the correspondence between the states in the Viterbi output and those in the ground truth. By fixing this probability we assure that the first code of the Viterbi output will match the inactive state.

The initialization of the variables for the EM algorithm was carried out as follows:

- Means (only for the GMs case) : μ_1 = [0.2 0.8 1.5] and μ_2 = [0.1 2 4]
- Covariance: we started with high and low values of covariance as we do not know a priori to what values the algorithm is expected to converge. Moreover, we chose a diagonal covariance matrix and two positive semidefinite matrix and not diagonal.

$$\Sigma_1 = \begin{pmatrix} 5 & 4 \\ 4 & 5 \end{pmatrix}, \qquad \Sigma_2 = \begin{pmatrix} 0.01 & 0 \\ 0 & 0.01 \end{pmatrix} \text{ and}$$

$$\Sigma_3 = \begin{pmatrix} 9 & 8 \\ 8 & 9 \end{pmatrix}$$

- Weight for each of the Gaussian components: random in the case of GMs, and 1 for KDE.
- Number of Gaussians per state (only GMs case): M = 2.
- State transition matrix (3 x 3): A = [0.6 0.2 0.2; 0.2 0.6 0.2; 0.2 0.6 0.2]. The running activities are very few and their duration is very short. That is the reason why the $a_{3,2}$ is such a high value whereas $a_{3,3}$ is low.
- Maximum number of iterations for the EM algorithm: *max-iter=50*

The data are divided into two sets of sequences for training and testing: one of 7 sequences of 1975 data in total and another of 6 sequences and 1605 data. The first set is used for training while both are separately used for validation. Table 3 shows the total classification error on the training and validation sets for the GMs and the KDE.

Table 3. Total classification error for the training and validation sets.

Error (%)	Training	Validation
(1) GMs (μ_1, Σ_1)	23,59	17,32
(2) GMs (μ_2, Σ_1)	16,86	15.82
(3) KDE (Σ_1)	14,48	16,45
(4) GMs (μ_1, Σ_2)	18,38	15,07
(5) GMs (μ_2, Σ_2)	17,37	15,32
(6) KDE (Σ_2)	14,17	16,01
(7) GMs (μ_1, Σ_3)	23,59	17,32
(8) GMs (μ_2, Σ_3)	23,59	17,32
(9) KDE (Σ_3)	14.48	16,26

The experiments show the stable results of KDE/HMM independently of the initialisation of its covariance parameter. It appears that the parameter space is very simple to search and the learning converges to the same value of Σ irrespectively of very different initial values. Conversely, the GMs HMM obtains significantly different error rates depending on the initial values of its means and covariance parameters. This shows the limitation of the GM model as a highly parametric technique of difficult initialization. The error on the training and validation sets for the KDE model remains around 14-16% while for the GMs model varies between 15 and 24% depending on the different combinations of initial covariances and means.
To provide further detail into these results, Table 4 shows the confusion matrix for the GMs and the KDE cases for experiments 1, 3, 5 and 6 in Table 3. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class (ground truth). Table 4 shows that the better overall results of KDE also correspond to improved inter-class errors with respect to the GMs model.

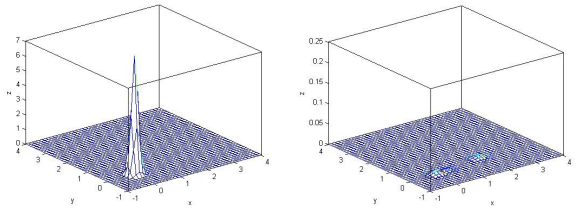
Table 4. Confusion matrix for the classification with the GMs (experiment 1) and KDE (experiments 3 and 6) models for the validation data.

GMs **(1)**	Predicted			
Actual		**in**	**Wk**	**r**
	in	560	16	0
	wk	108	719	124
	r	0	30	48

KDE **(3)**	Predicted			
Actual		**in**	**Wk**	**r**
	in	567	8	1
	wk	135	736	80
	r	0	40	38

KDE **(6)**	Predicted			
Actual		**in**	**Wk**	**r**
	in	569	6	1
	wk	143	743	65
	r	0	42	36

Finally, Figure 6 shows the pdf's of the emission probabilities for the GMs and KDE for experiments 5 and 6 for each of the states. The pdf's show the intuitively different modelling of GMs and KDE. In particular, the KDE emission probabilities are not required to be compact and spontaneously adjust to model non-clustered data and with data with uniform regions.



(a) *(d)*

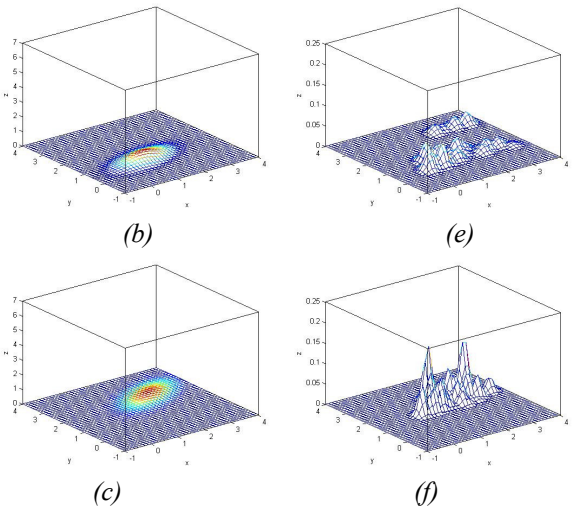


Figure 6: GMs derived from the EM algorithm, experiment (5) (in a scale 0-7) for inactive (a), walking (b) and running (c). KDE derived from the modified EM algorithm, experiment (6) (in a scale 0-0.25) for inactive (d), walking (e) and running (f).

5. Conclusions

In this paper, we have presented a modified hidden Markov model with KDE emission probabilities (HMM/KDE) and its use for activity recognition in videos. In the proposed approach, kernel density estimation of the emission probabilities occurs simultaneously with that of all the other model parameters thanks to an adapted Baum-Welch algorithm. This has allowed us to retain maximum-likelihood estimation while overcoming the known limitations of mixture of Gaussians in modelling certain data distributions such as uniform and non-clustered data. Experiments on activity recognition have been performed on the CAVIAR video surveillance database and reported in the paper. Overall, the error on the training and validation sets with kernel density estimation remains around 14-16% while for the conventional Gaussian mixture approach varies between 15 and 24%. The main advantage that we identify in the proposed KDE/HMM model is that its accuracy seems substantially independent from the choice of the initial value of its only parameter, the covariance matrix common to all its kernel components. On the contrary, the conventional GMs modelling of emission probabilities is a highly parametric technique and proves of challenging initialisation.

Obviously in a way, the increased and more stable accuracy obtained by KDE comes at higher computational costs for both model estimation and evaluation as the number of kernels in KDE is much greater than that typical of GM components. However, this does not seem to represent a significant issue in applications such as activity recognition in videos as they are however dominated by the heavy low-level processing of foreground extraction and tracking.

References

[1] O. Masound and N. Papanikolopoulos. Recognizing Human activities. In Proc. IEEE Conference on Advanced Video and Signal Based Surveillance, 157-162, 2003.

[2] M. Brand and V. Kettnaker. Discovery and segmentation of activities in video. IEEE Trans. Pattern Analysis and Machine Intell., 22(8): 844-851, 2000.

[3] N.M. Oliver, B. Rosario, and A.P. Pentland. A Bayesian computer vision system for modeling human interactions. IEEE Trans. on Pattern Anal. and Machine Intell., 22(8): 831-843, 2000.

[4] J. Ben-Arie, Z. Wang, P. Pandit, and S. Rajaram. Human Activity Recognition Using Multidimensional Indexing. IEEE Trans. on Pattern Anal. and Machine Intell., 24(8): 1091-1104, 2002.

[5] Ju Han and B. Bhanu. Human Activity Recognition in Thermal Infrared Imagery. In Proc. IEEE CS Computer Vision and Pattern Recognition, 3:17-17, 2005.

[6] L. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proc. IEEE, 77:257-286, 1989.

[7] J. Bilmes. A gentle tutorial on the EM algorithm and its application to parameter estimation for Gaussian mixture and Hidden Markov Models. Tech. Rep. ICSI-TR-97-021, University of California Berkeley, 1998.

[8] H. Bourlard and N. Morgan. Connectionist Speech Recognition. A Hybrid Approach. Kluwer Academic Publishers, 1994.

[9] E. Trentin. Nonparametric Hidden Markov Models: Principles and Applications to Speech Recognition. In Lecture Notes in Computer Science 2859:3-21, 2003.

[10] S.E. Kruger, M. Schaffoner, M. Katz, E. Andelic, and A. Wendemuth. Mixture of Support Vector Machines for HMM based Speech Recognition. In Proc. 18th Int. Conf. on Pattern Recognition 4:326- 329, 2006.

[11] B. A. Turlach. Bandwidth Selection in Kernel Density Estimation: A Review. Technical Report Université Catholique de Louvain, Belgium, 1993.

[12] CAVIAR: Context Aware Vision using Image-based Active Recognition. http://homepages.inf.ed.ac.uk/rbf/CAVIAR/ (last accessed: 30 November 2006).

[13] R. P. W. Duin. On the choice of smoothing parameters for Parzen estimators of probability density functions. IEEE Trans. on Computers, 25(11):1175-1179, 1976.

[14] Hidden Markov Model (HMM) Toolbox for Matlab: http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html