Human Motion Recognition using Gaussian Processes Classification

Hang Zhou[1], Liang Wang[2], David Suter[1]
[1]*Department of Electrical and Computer Systems Engineering*
Monash University, Vic 3800, Australia
{*hang.zhou, d.suter*}*@eng.monash.edu.au*

[2]*Department of Computer Science and Software Engineering*
The University of Melbourne, Vic 3010, Australia
lwwang@csse.unimelb.edu.au

Abstract

This paper investigates the applicability of Gaussian Processes (GP) classification for recognition of articulated and deformable human motions from image sequences. Using Tensor Subspace Analysis (TSA), space-time human silhouettes (extracted from motion videos) are transformed to low-dimensional multivariate time series, based on which structure-based statistical features are calculated to summarize the motion properties. GP classification is then used to learn and predict motion categories. Experimental results on two real-world state-of-the-art datasets show that the proposed approach is effective, and outperforms Support Vector Machine (SVM).

1. Introduction

Human motion recognition aims at classifying motions into known categories such as walking or waving. Due to the great variations in the captured human motion sequences between different instances or different persons (with various body types, motion styles and speeds), how to extract distinguishable features and get them well modeled still remains a challenge.

The 'state-space' approaches using temporal models such as Hidden Markov Models (HMMs) [4] and Conditional Random Fields (CRFs) [7] have been used to model motion patterns. However, such temporal probabilistic models are of high computational complexity since they usually require detailed statistical modeling, which involves assumptions about the probability distributions of variables of the dynamical models and the development of inference methods as well as model parameter learning algorithms. In contrast, being a kernel-based non-parametric model, Gaussian Process (GP) [6] is more computationally tractable by employing the Gaussian function properties. General properties of the GP kernel is controlled by a few hyperparameters which can be easily estimated under the Baysian framework. Furthermore, GP is used as a Bayesian prior to express beliefs about the underlying functions being modeled, which is linked to data via the likelihood. The posterior distribution is directly calculated given the training data. This makes GP being flexible and expressive in dealing with complex datasets.

So far, the solutions applying GP to human motion analysis include Wang *et al.* [9] and Raskin *et al.* [5]. Gaussian Process Dynamical Models (GPDM) are proposed in [9] for nonlinear time series analysis with application to human motion capture data. Raskin *et al.* [5] proposed to combine GPDM and annealed particle filtering for tracking and classifying human motions. Note that GPDM is essentially a 'state-space' solution which models both the distribution of the observed data and the dynamics in the latent space.

In this paper, we present a new method for human motion recognition from image sequences using GP classification. Characteristic-based descriptors [11, 10] are used to transform time-varying dynamic features of varied-length motion sequences to fixed-length feature vectors (and thus the temporal classification problem is converted to a static classification one), which enables the use of GP. Extensive experimental and comparative results on two recent motion datasets demonstrate the effectiveness of our approach.

2. Feature extraction of motions

Given a database consisting of n motion sequences $\mathcal{M} = \{\mathbf{M}_1, \mathbf{M}_2, \cdots, \mathbf{M}_n\}$, we wish to extract infor-

978-1-4244-2175-6/08/$25.00 ©2008 IEEE

mative space-time silhouettes from original videos to represent the motions performed. The process of feature extraction is shown in Figure 1. For each motion sequence $\mathbf{M}_i$ including T_i image frames in $\mathcal{M}$, *i.e.*, $\mathbf{M}_i = \{\mathbf{I}_1^i, \mathbf{I}_2, \cdots, \mathbf{I}_{T_i}^i\}, i = 1, 2, \cdots, n$, the associated sequence of human silhouettes can be obtained by foreground detection techniques. Since the size and position of the foreground region vary among frames, we center the silhouette images and normalize them to $\mathcal{S}_i = \{\mathbf{S}_1^i, \mathbf{S}_2^i, \cdots, \mathbf{S}_{T_i}^i\}$ with the same dimensions of $n_1 \times n_2$.

To represent human motions in a more compact subspace rather than in the high-dimensional image space, we adopt Tensor Subspace Analysis (TSA) [2] to perform subspace learning of the articulated motion space. TSA preserves the spatial information of silhouette images by representing an image as a second-order tensor (or a matrix). Given a set of m normalized silhouette images from the training data $\{\mathbf{S}_1, \mathbf{S}_2, \cdots, \mathbf{S}_m\}$ in $\mathcal{R}^{n_1} \otimes \mathcal{R}^{n_2}$, TSA aims to find two transformation matrices $\mathbf{U}$ of size $n_1 \times l_1$ and $\mathbf{V}$ of size $n_2 \times l_2$ that map these silhouette images to another set $\{\mathbf{Y}_1, \mathbf{Y}_2, \cdots, \mathbf{Y}_m\}$ in $\mathcal{R}^{l_1} \otimes \mathcal{R}^{l_2} (l_1 < n_1, l_2 < n_2)$, such that $\mathbf{Y}_i = \mathbf{U}^T \mathbf{S}_i \mathbf{V}$. For more details of TSA the reader may refer to [2]. In the learned tensor subspace, any silhouette sequence $\mathcal{S}_i$ can be accordingly projected into a trajectory $\mathcal{P}_i = \{\mathbf{P}_1, \mathbf{P}_2, \cdots, \mathbf{P}_{T_i}\}, \mathbf{P}_i \in \mathcal{R}^{l_1} \otimes \mathcal{R}^{l_2}$. We can regard $\mathcal{P}_i$ as a form of multivariate time series with the number of dimensions $l = l_1 \times l_2$.

Statistical features are then extracted to summarize multivariate time series which turns motion time series of different lengths into a feature vector of the same length. The nine most informative, representative and easily measurable characteristics are chosen to summarize the time series structure [11]: *trend*, *seasonality*, *serial correlation*, *non-linearity*, *skewness*, *kurtosis*, *self-similarity*, *chaotic*, and *periodicity*. Based on these characteristics, corresponding metrics are calculated to form the structure-based feature vectors [10]. For each dimension of $\mathcal{P}_i$, we obtain 13 statistical features. Thus, a multivariate time series $\mathcal{P}_i$ is summarized by a d-dimensional ($d = 13 \times l$) feature vector $\mathbf{x}$.

3. GP classification

A Gaussian process is a collection of random variables, any finite number of which has a joint Gaussian distribution [6]. A GP is fully specified by its mean function $m(\mathbf{x})$ and kernel function $k(\mathbf{x}, \mathbf{x}')$, *i.e.*,

$$f \sim \mathrm{GP}(m, k) \tag{1}$$

The GP classification models the posterior directly. The GP prior is represented by the kernel function which characterizes correlations between points in the training data (which is a sample process). The kernel function's hyperparameters can be learned from the training data. The kernel function studied in this paper is the Radial Basis Function (RBF).

Following the exposition in [6]: we have a dataset $\mathcal{D}$ with n observations $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1, 2, \cdots, n\}$, where $\mathbf{x}$ is the input vector of dimension d (*e.g.*, motion feature vector here) and y is the class labels $+1/-1$. The input $d \times n$ matrix is denoted as X. Predictions for new inputs $\mathbf{x}'$ are made out of this given training data using the GP model. As described in [6], GP binary classification is done by first calculating the distribution over the latent function f corresponding to the test case

$$p(f'|X, y, \mathbf{x}') = \int p(f'|X, \mathbf{x}', f) p(f|X, y) df \tag{2}$$

where

$$p(f|X, y) = p(y|f) p(f|X) / p(y|X) \tag{3}$$

is the latent variable posterior. $p(f'|X, \mathbf{x}', f)$ is the predictive posterior wrt possible latent functions, and the values of this could lie anywhere within the range of $(-\infty, +\infty)$. So the probabilistic prediction is made by

$$\bar{\pi}' = p(y' = +1|X, y, \mathbf{x}') = \int \sigma(f') p(f'|X, y, \mathbf{x}') df' \tag{4}$$

where σ can be any sigmoid function that 'squashes' the prediction output to guarantee the valid probabilistic value within the range of $[0, 1]$.

For multi-class classification problem, we can treat each one class as being independent from the others, and apply binary classification individually to each (one) class versus the rest classes.

4. Experiments and results

4.1. Data sets

We use two state-of-the-art datasets, *i.e.*, Dataset I [8] and Dataset II [1], to evaluate our approach. Dataset I consists of 100 video sequences from 10 different motions performed by one subject (*i.e.*, pick up object, jog in place, push, squash, wave, kick, bend to the side, throw, turn around, and talk on cell phone), and 10 different instances for each motion. Different instances of the same motion may consist of varying relative speeds, so this dataset is used to examine the effect of the temporal rate of execution on motion recognition. Some sample images are shown in Figure 2(a).

Dataset II includes 90 low-resolution videos from 9 different people, each performing 10 motions in

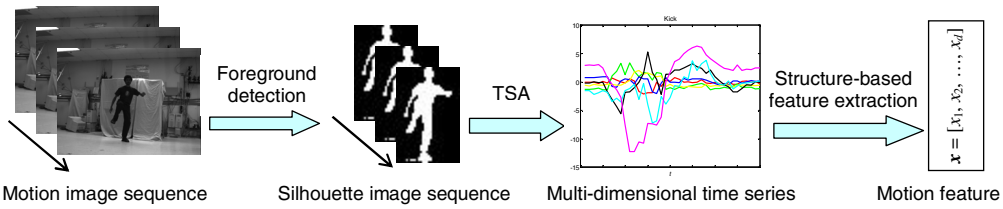

Figure 1. From motion image sequence to characteristic-based descriptor

an *repetitive* manner (*i.e.*, bend, jump jack, jump-forward-on-two-legs, jump-in-place-on-two-legs, run, skip, gallop-sideways, walk, wave-one-hand, and wave-two-hands)[1]. For this dataset, we extract a total of 198 sample sequences from the original 90 videos for testing (each of which includes a complete cycle of motions). There exist inter-person differences between the same motions due to different physical sizes and motion styles (and speeds), so this dataset provides more realistic data for the test of the method's versatility in terms of motion variations at both temporal and spatial scales. Some sample images are shown in Figure 2(b).

4.2. Data processing and classification

Datasets I and II are provided with silhouette masks. We directly center and normalize these silhouette images into 48×32 resolution (*i.e.*, $n_1 = 48$ and $n_2 = 32$). We then perform TSA [2] for dimensionality reduction. The reduced dimension of the input features is set to 4×4 (*i.e.*, $l_1 = l_2 = 4$). That is, each motion sequence is projected into a 16-dimensional time series in the learned embedding space, from which we extract 13 statistical features for each univariate time series [10]. Then these features from each univariate time series are joined as one 208-dimensional (*i.e.*, 16×13) vector.

To make the results comparable, data is processed in a similar way as in [10]. For Dataset I, we divide the 100 video sequences into 10 disjoint sets with each of the set including one sample sequence of all the 10 distinct motions. For Dataset II, the number of sample sequences for the 10 motions are $9, 23, 24, 27, 14, 22, 25, 16, 19$ and 19 respectively. This dataset is divided into 9 non-overlapping set in a way to make sequences of each motion distribute evenly among the partitioned sets and each of them includes sequences of all the 10 motions. This is a more random and less demanding division compared with the way in [10] where each of the 9 divided set includes all motions from one person.

For either Dataset I or Dataset II, classification is done in a leave-one-out (LOO) way among the divided subsets, *i.e.*, each time leave one subset out (denote as *the test set*) for testing and all the remaining subsets (denote as *the training set*) for training. Two kinds of GP classification which we call GP(a) and GP(b) are implemented. GP(a) applies GP classification to each training and testing set individually and take the mean recognition rate as the final result. GP(b) is implemented in a substantially different way from GP(a) by merging all the LOO training sets into one single set which allocates the LOO training sets on the input space in such a way: all LOO training sets input features are normalized to $[0, 1]$ and put a interval of 5 between 2 neighboring training set to make them being independent of each other while modeled by GP. For example for Dataset I, the merged training set comprises 10 LOO training sets with input ranges being $[0\ 1], [5\ 6], \cdots, [45\ 46]$, so that the merged training set input ranges from 0 to 46. Testing datasets are merged accordingly in the same way. In this way, all the LOO training-testing set pairs share the same GP model and any of the LOO training set has little effect on the testing sets other than its corresponding one. The advantage of GP(b) is that the GP model is optimized on all the LOO training-test set pairs at one time rather than in GP(a) where the GP model is applied to each training-test set pair separately which makes the GP hyperparameters estimation as well as the classification results suboptimal.

4.3. Results and analysis

(a) Dataset I

(b) Dataset II

Figure 2. Example images of motion data

We run Lawrence's program [3][2] for GP classifica-

[1] http://www.wisdom.weizmann.ac.il/∼vision/SpaceTimeActions.html

[2] http://www.cs.man.ac.uk/ neill/ivm/downloadFiles/

tion. The results using GP(a), GP(b), as well as comparison with the results using NN (Nearest-Neighbor) and SVM in [10] on the two datasets are summarized in Table 1 and Table 2. Binary GP classification is used here for any of the motions by labeling the target motion being +1 and all the rest motions being -1 in the training data and for classifying the test data. It can be seen from Table 1 and Table 2 that the GP performs better than either NN or SVM on both datasets. In addition, results of GP(b) further outperforms GP(a) as GP(b) is optimized by taking all training datasets into consideration simultaneously. It should be noted that although the overall classification rate of GP(a) is higher than that of SVM, part of GP(a) results are slightly worse than SVM. We believe this is because GP model is optimized on each LOO training set individually in GP(a) which causes suboptimal results.

Although both GP and SVM are kernel based models, GP produces an output with a clearer probabilistic interpretation. It takes into account the predictive variance of the underlying function. For SVM, uncertainties are not taken into account. There have no predictive variances, or learning of hyperparameters by Maximum Likelihood. Therefore, GP is more likely to achieve better classification results.

Table 1. Recognition rates for Dataset I consisting of 100 sequences (%)

Motions	NN	SVM	GP(a)	GP(b)
Pick	80.0	90.0	98.0	100.0
Jog	100.0	100.0	100.0	100.0
Push	90.0	100.0	100.0	100.0
Squash	90.0	100.0	100.0	100.0
Wave	100.0	100.0	100.0	100.0
Kick	30.0	70.0	98.0	100.0
Bend-side	100.0	100.0	100.0	100.0
Throw	70.0	100.0	99.0	100.0
Turn	100.0	100.0	100.0	100.0
Phone	80.0	100.0	95.0	100.0
Average	**84.0**	**96.0**	**99.0**	**100.0**

5. Conclusion

An effective new method has been proposed in this paper to improve human motion recognition by applying GP classification which is particularly effective in modeling high-dimensional data based on small training dataset. The temporal motion sequences are transformed to a vector-based pattern representation using characteristic-based descriptors. In this way the temporal sequence classification is elegantly converted to a static classification problem on which the binary GP classification is used to recognize unseen motion sequences to one of the known motion categories. Experimental evaluation on the two recent motion datasets has shown that our approach greatly enhances the accuracy.

Table 2. Recognition rates for Dataset II consisting of 198 sequences (%)

Motions	NN	SVM	GP(a)	GP(b)
Bend	88.9	100.0	100.0	100.0
Jump	87.0	95.7	94.1	100.0
Pjump	79.2	87.5	86.2	100.0
Run	59.3	92.6	90.0	100.0
Side	64.3	86.7	98.4	100.0
Kick	95.5	86.4	99.0	100.0
Skip	60.0	84.0	91.5	100.0
Walk	100.0	100.0	98.2	100.0
Wave1	89.5	100.0	99.5	100.0
Wave2	68.4	89.4	93.4	100.0
Average	**78.3**	**91.4**	**95.0**	**100.0**

References

[1] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Action as space-time shapes. In *ICCV*, 2005.

[2] X. He, D. Cai, and P. Niyogi. Tensor subspace analysis. In *NIPS*, 2005.

[3] N. D. Lawrence, J. C. Platt, and M. I. Jordan. Extensions of the informative vector machine. *Deterministic and Statistical Methods in Machine Learning*, 2004.

[4] N. Nguyen, D. Phung, S. Venkatech, and H. Bui. Learning and detecting activities from movement trajectories using the hierarchical hidden Markov models. In *CVPR*, 2005.

[5] L. Raskin, M. Rudzsky, and E. Rivlin. Tracking and classifying of human motions with gaussian process annealed particle filter. In *ACCV*, volume I, pages 442–451, 2007.

[6] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[7] C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas. Conditional models for contextual human motion recognition. In *ICCV*, 2005.

[8] A. Veerarghavan, R. Chellappa, and A. K. Roy-Chowdhury. The function space of an activity. In *CVPR*, 2006.

[9] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *TPAMI*, 30(2):283–298, 2008.

[10] L. Wang, X. Wang, C. Leckie, and K. Ramamohanarao. Characteristic-based descriptors for motion sequence recognition. In *PAKDD*, 2008.

[11] X. Wang, K. Smith, and R. Hyndman. Characteristic-based clustering for time series data. *Data Mining and Knowledge Discovery*, 13(3):335–364, 2006.