Support Vector Machines with Time Series Distance Kernels for Action Classification

Mohammad Ali Bagheri
Dalhousie University
University of Lar
bagheri@cs.dal.ca

Qigang Gao
Dalhousie University
qggao@cs.dal.ca

Sergio Escalera
Universitat de Barcelona
sergio@maia.ub.es

Abstract

Despite the outperformance of Support Vector Machine (SVM) on many practical classification problems, the algorithm is not directly applicable to multi-dimensional trajectories having different lengths. In this paper, a new class of SVM that is applicable to trajectory classification, such as action recognition, is developed by incorporating two efficient time-series distances measures into the kernel function. Dynamic Time Warping and Longest Common Subsequence distance measures along with their derivatives are employed as the SVM kernel. In addition, the pairwise proximity learning strategy is utilized in order to make use of non-positive semi-definite kernels in the SVM formulation. The proposed method is employed for a challenging classification problem: action recognition by depth cameras using only skeleton data; and evaluated on three benchmark action datasets. Experimental results demonstrate the outperformance of our methodology compared to the state-of-the-art on the considered datasets.

1. Introduction

The fast and reliable recognition of human actions from captured videos has been a goal of Computer Vision for decades. Robust action recognition has diverse applications, including gaming, sign language interpretation, human-computer interaction (HCI), surveillance, and health care. Understanding gestures/actions from a real-time visual stream is a challenging task for current computer vision algorithms. Over the last decade, spatial-temporal (ST) volume based holistic approaches and local ST feature representations have been reportedly achieved good performance on some action datasets, but they are still far from being able to express the effective visual information for efficient high-level interpretation.

Development of low-cost depth sensors with acceptable accuracy has greatly simplified the task of action recognition [20]. Most importantly, the recent release of the Microsoft Kinect camera and its evolving skeleton joints detection technique in late 2011 led to a substantial revolutionary effect in the field of Computer Vision and created a wide range of opportunities for demanding applications. Shotton et al. [20] proposed one of the greatest advances in the extraction of the human body pose from depth data, which is provided as a part of the Kinect platform. Their work enables us to recover 3D positions of skeleton joints in real time with reasonable accuracy [5, 20].

In this paper, we address the problem of human action classification by employing spatio-temporal information of skeleton joint points, i.e. the real positions of body joints over the time. More specifically, we use the 3D trajectories of dominant body joints, obtained by the Kinect camera. These trajectories encode significant discriminative information and is sufficient for human beings to recognize different actions [9]. In addition, according to an influential computational model of human visual attention theory [22], visual attention leads to visual salient entities, which provide selective visual information to make human visual perception efficient and effective. Trajectories of skeleton joints are visual salient points of human body, and their movements in 4D space reflect motion semantics.

From the classification point of view, these trajectories may be considered as multi-dimensional time series. The traditional recognition technique in the literature is based on time series dis(similarity) measures (such as Dynamic Time Warping). For these general dis(similarity) measures, k-nearest neighbor algorithms are a natural choice. In general, the k-NN classification algorithm work reasonably well; but are known to be sensitive to noise and outliers. Since SVMs often outperform k-NNs on many practical classification problems where a natural choice of positive semidefinite (PSD) kernels exists, it is desirable to extend the applicability of kernel SVMs [8].

In our action classification problem, however, time series distances measures are generally non-PSD kernels and basic SVM formulations are not directly applicable. To in-

clude non-PSD kernels in SVM, several ad-hoc strategies have been proposed. The straightforward strategy is to simply overlook the fact that the kernel should be non-PSD. In this case, the existence of a Reproducing Kernel Hilbert Space is not guaranteed [19] and it is no longer clear what is going to be optimized.

Another strategy, which has been applied in our work, is based on *pairwise proximity function* SVM (ppfSVM) [7]. This strategy involves the construction of a set of inputs such that each sample is represented with its dis(similarity) to all other samples in the dataset [8]. The ppfSVM is related to the arbitrary kernel SVM, a special case of the generalized Support Vector Machines [12]. The name is due to the fact that no restrictions such as positive semi-definiteness, differentiability or continuity are put on the kernel function [8].

In this paper, we investigate the effectiveness of this strategy for human action classification when the pairwise similarities are based on time-series distances measures. More specifically, we demonstrate the effectiveness of two trajectory-based distances measures - including Longest Common Subsequence (LCSS) and Dynamic Time Warping (DTW) as well as their derivatives- as SVM kernel functions. The experimental results on two benchmark datasets prove the outperformance of the proposed method compared to the state-of-the-art techniques.

The rest of the paper is organized as follows: Section 2 reviews the related work on action recognition, and briefly introduces LCSS and DTW. Section 3 presents our methodology for action recognition. Section 4 evaluates the proposed method and Section 5 concludes the paper.

2. Related Work

2.1. Action Recognition using Skeleton Data

Since 2011, there has been an outburst of research articles addressing action analysis using depth information. These studies may be categorized into those that employe the original depth maps and those that have only used the skeleton data. Due to limited space, we will briefly review some of the representative studies in the second category.

In [29], visual features for activity recognition are computed based on the spatial and temporal differences among detected joints. This feature set contains information about static posture, motion, and offset. Then, Naive Bayes Nearest Neighbor method was applied for the classification task.

Alternatively, a histogram of 3-D joint locations (HOJ3-D) for body posture representation is proposed in [28]. In this representation, the 3D space is partitioned into bins using a spherical coordinate system, and the HOJ3-D histogram is constructed by casting joints into certain bins. After applying linear discriminant analysis (LDA) for dimensionality reduction, HOJ3-D vectors are clustered into

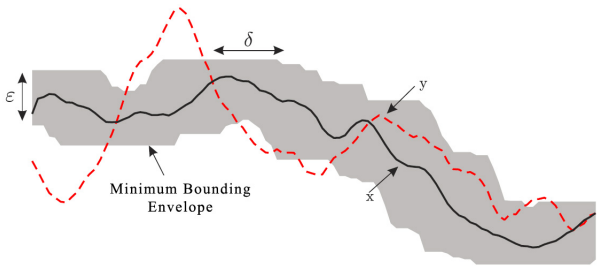

Figure 1. Matching within δ in time and ϵ in space. Everything outside the bounding envelope can never be matched (Reprinted from [6]).

k posture visual words. The temporal behaviour of these visual words is coded by discrete HMMs. Reyes et al. [18] used 15 joints from Primesense API to represent a human model. Dynamic Time Warping (DTW) with weighted joints is used to achieve a real-time action recognition system. The work in [14], combined a Gaussian-Binary restricted Boltzmann machine (GB-RBM) with a hidden Markov model (HMM) and presented a method to use RBM as a generative model for multi-class classification. In [2], the authors proposed an ensemble of five action learning techniques, each performing the recognition task from a different perspective and combined the outputs of these classifiers based on the Dempster-Shafer combination theory.

2.2. Longest Common Subsequence

The longest common subsequence dissimilarity measure is a variation of the edit dissimilarity measure, initially used in speech recognition. The underlying idea is to match two sequences by allowing them to stretch, without rearranging the sequence of the elements but allowing some elements to be unmatched or left out (e.g., outliers). Roughly speaking, LCSS counts the number of pairs of points from two sequences that match. The LCSS measure has two parameters, δ and ϵ, as shown in Fig. 1. The constant δ controls how far in time we can go in order to match a given point from one trajectory to a point in another trajectory. This parameter is a warping threshold and controls the window size for matching a given point from one trajectory to a point in another one, which is usually set to a percentage of the sequence length. The constant $0 < \epsilon < 1$ is the matching threshold: two points from two sequences are considered to match if their distance is less than ϵ.

Longest common subsequences of the time series x and y of length n and m is recursively defined as follows:

$$L(i,j) = \begin{cases} 0 & \text{for } i = 0 \\ 0 & \text{for } j = 0 \\ 1 + L(i-1, j-1) & \text{for } |x_i - y_j| < \epsilon \\ & \text{and } |i-j| \leq \delta \\ max(L(i-1,j), L(i,j-1)) & \text{in other cases} \end{cases}$$

$L(n,m)$ is the similarity between x and y, because it corresponds to the length of the longest common subsequence of elements between time series. The dissimilarity between x and y has been defined as follows:

$$LCSS(x,y) = \frac{(n+m-2L(n,m))}{(n+m)} \quad (1)$$

2.3. Dynamic Time Warping

Dynamic Time Warping (DTW) is a well-known algorithm which aims to compare and align two temporal sequences, taking into account that sequences may vary in length (time) [18]. DTW employs the dynamic programming technique to find the minimal distance between two time series, where sequences are warped by stretching or shrinking the time dimension. Although it was originally developed for speech recognition, it has also been employed in many other areas like handwriting recognition, econometrics, and action recognition.

An alignment between two time series can be represented by a warping path which minimizes the cumulative distance. The DTW distance between time series x and y of length n and m will be recursively defined as:

$$DTW(i,j) = d(i,j) + min \begin{cases} DTW(i,j-1) \\ DTW(i-1,j) \\ DTW(i-1,j-1) \end{cases}$$

Here, $d(i,j)$ is the square Euclidean distance of x_i and y_j.

3. Time Series based Kernel SVM

The proposed algorithm works as follows:

1. **Feature extraction:** Given a depth image, 20 joints of the human body can be tracked by the skeleton tracker (Fig. 2. a & b). At frame t, the position of each joint k is uniquely defined by three coordinates $P_k(t) = [x_i(t), y_i(t), z_i(t)]$. Instead of using the positions of joints, we employ the relative position of each joint to the torso at each frame, as more discriminative and intuitive 3D joint features (Fig. 2. c).
2. **Compute non-PSD kernels:** As described in the next subsection, we compute the non-PSD kernels based on pairwise distance of each normalized 3D trajectory to other trajectories, using LCSS and DTW (Fig. 2. d).
3. **Classification:** As described in subsection 3.2, we train four ppfSVMs using the computed kernels (Fig. 2.e) and simply fuse these classifiers (Fig. 2.f).

3.1. Kernel from Pairwise Data

Given labeled training data of the form $\{(x_i, y_i)\}_{i=1}^m$, with $y_i \in \{-1, +1\}$ [1], the standard form of SVM finds a hyperplane which best separates the data by minimizing a constrained optimization problem:

$$\begin{aligned} \tau(w,\xi) =& \frac{1}{2}||w||^2 + C\sum_{i=1}^{m}\xi_i \quad (2) \\ \text{subject to: } & y_i((w.x_i)+b) + \xi_i \geq 1 \\ & \xi_i \geq 0 \end{aligned}$$

where ξ_i are slack variables and $C > 0$ is the tradeoff between a large margin and a small error penalty.

The cornerstone of SVM is that non-linear decision boundaries can be learnt using the so called 'kernel trick'. A *Kernel* is a function $\mathcal{K} : \mathcal{X} \times \mathcal{X} \mapsto \mathcal{R}$, such that for all x_i, $i \in \{1, \ldots, m\}$ yields to a symmetric positive semi-definite (PSD) matrix K, where $K_{ij} = \kappa(x_i, x_j)$. Indeed, the kernel function implicitly maps their inputs into high-dimensional *feature spaces*, $x \mapsto \Phi(x)$. Two common kernel functions are the Gaussian Kernel and the Linear kernel.

In the dual formulation, the SVM algorithm maximizes:

$$\begin{aligned} W(a) =& \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{ij}\alpha_i\alpha_j y_i y_j \kappa(x_i,x_j) \quad (3) \\ \text{subject to: } & 0 \leq \alpha_i \leq C \text{ and } \sum_{\alpha_i y_i} = 0 \end{aligned}$$

The decision function is given by:

$$f(x) = sign\left(\sum_{i=1}^{m} y_i\alpha_i\kappa(x,x_i) + b\right) \quad (4)$$

where the threshold b is defined as:

$$b = y_i - \sum_{i=1}^{m} y_i\alpha_i\kappa(x_i,x_j) \quad (5)$$

In our action classification problem, however, time series distances measures are generally non-PSD kernels and basic SVM formulations are not directly applicable. To deal with this problem, we follow the strategy proposed in [7], which can be applied to general pairwise similarity measures. This strategy involves the construction of a set of inputs such that each sample is represented with its dis(similarity) to all other samples in the dataset. The basic SVM is then applied to the transformed data in the usual way. As a consequence, sparsity of the solution may be lost.

According to [7], it is assumed that instead of a standard kernel function, all that is available is a proximity function, $P : \mathcal{X} \times \mathcal{X} \mapsto R$. No restrictions are placed on the function P, not symmetry nor even continuity. The mapping $\Phi(x)$ is defined by:

$$\Phi(x) : x \mapsto (P(x,x_1), P(x,x_2), \ldots, P(x,x_m)^T \quad (6)$$

[1] In our formulation, the input samples, x_i, are not restricted to be a subset of R^n and can be any set, e.g. set of images or videos.

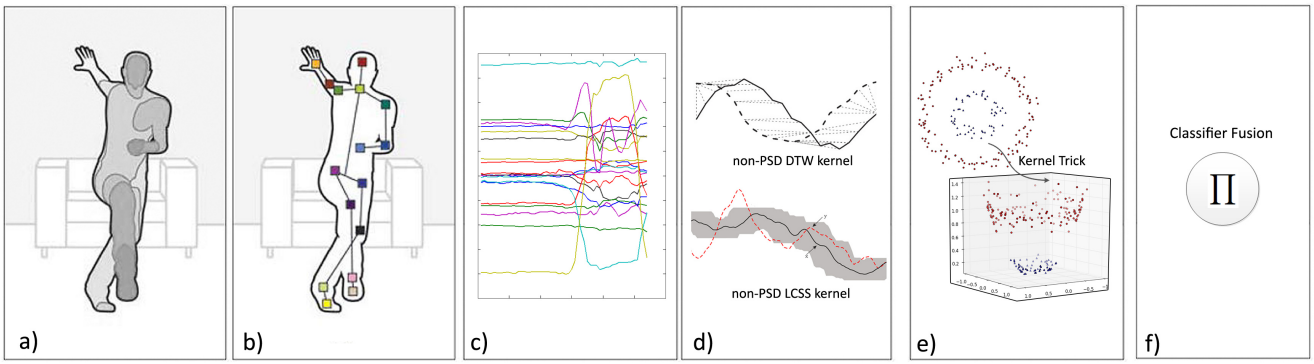

Figure 2. The framework of the proposed *Time Series based Kernel SVM for action classification*; a) an initial depth map; b) positions of 20 joints obtained by Kinect [20]; c) extract features: relative trajectories of joints over the time; d) compute non-PSD kernels using DTW and LCSS; e) Train ppfSVMs; f) classifier fusion.

where $x_i, i = 1, \ldots, m$ are the examples in dataset. Here, we represent each sample x_i by $x_i = \Phi_m(x_i)$ i.e. an m-dimensional vector containing proximities to all other samples in the dataset. Let P denote the $m \times m$ matrix with entries $P(x_i, x_j), i, j \in \{1, \ldots, m\}$. Using the linear kernel on this data representation, the resulting kernel matrix becomes $K = PP^T$. In this case the decision rule (3) simplifies to

$$f(x) = sign\left(\sum_{i=1}^{m} y_i \alpha_i P \Phi_m(x) + b\right) \tag{7}$$

All elements of $\Phi_m(x_i)$ must be computed when classifying a point x.

In this study, kernels from pairwise data is obtained by pairwise time-series distance measures, including DTW and LCSS measures. In addition, as described in the next subsections, we also calculate the pairwise distances using the derivatives of these two time-series measures.

3.1.1 Derivatives of Time Series Distance Measures

Despite the success of time series dis(similarity) measures, i.e. DTW and LCSS, they may fail in some situations. For example, since the DTW algorithm aims to explain variability in the Y-axis by warping the X-axis, it may results in unintuitive alignments where a single point on one sequence maps onto a large subsection of the other sequence; which is referred to as *"singularity"* in the related literature [10]. Also, they may fail to find obvious, natural alignments of two time series simply because a feature (i.e peak, valley, inflection point, plateau etc.) in one series is slightly higher or lower than its corresponding feature in the other time series.

To address such problems, the derivatives versions of DTW and LCSS are also employed in this work in order to enhance the level of feature representation. These modified version are called *Derivative DTW (DDTW)* and *Derivative LCSS (DLCSS)*. More formally,

$$DDTW \triangleq DTW(\nabla x, \nabla y) \tag{8}$$

$$DLCSS \triangleq LCSS(\nabla x, \nabla y) \tag{9}$$

where ∇x and ∇y are estimated derivatives of two time series x and y, respectively.

3.2. Classifier Fusion

In order to utilize the information encoded in the function values of time series and values of their first derivatives, we employed a simple ensemble classification framework [1]. In this framework, four SVMs are trained with four different types of kernels, i.e. DTW, DDTW, LCSS, DLCSS. In testing phase, the class of each sample, $\boldsymbol{x}$, is determined by:

$$c(\boldsymbol{x}) = \arg\max_i \prod_{t=1}^{4} w_t \mu_{t,i}(\mathbf{x}), i = 1, \ldots, N_c \tag{10}$$

where $c(\boldsymbol{x})$ is the ensemble class prediction, N_c is the number of classes, and $\mu_{t,i}(\boldsymbol{x}) \in [0, 1]$ represents the support given by the tth classifier to the ith class. w_t represents the weight of tth classifier, which is based on the classifier's accuracy on the training data.

4. Experiments

Here, we present the experimental details of evaluation, including the datasets used, settings of the experiments, as well as the obtained results. The codes were implemented in C/C++ with an interface in Matlab and is available upon request.

4.1. Datasets

We evaluated our framework on three public benchmark datasets: MSRAction3D [11], Cornel activity dataset

Figure 3. Some example gestures in the Chaleran dataset are very easy to be confused, even from human visual perception. (a) *Che vuoi* vs. *Che due palle*. For the *Che vuoi* gesture, both hands are in front of the chest area, where for *Che due palle* gesture they are near the waist region. (b) *Vanno d'accordo* vs. *Cos hai combinato*: both hand positions are very close and with the same motion directions; (c) both gestures, *Si sono messid'accordo* and *non ce ne piu*, require hand rotations; (d) four gestures, *Furbo, seipazzo, buonissimo*, and *cosatifarei* are required with the finger pointing to the head area, which cannot be easily determined, even with human eyes.

(CAD-60) [21], and the Multi-modal Gesture Recognition Challenge 2013 (Chalearn) [4].

MSRAction3D dataset: This dataset [11] is a well-known benchmark dataset for 3D action recognition. This dataset contains 20 actions, including *high arm wave, horizontal arm wave, hammer, hand catch, forward punch, high throw, draw x, draw tick, draw circle, hand clap, two hand wave, side-boxing, bend, forward kick, side kick, jogging, tennis swing, tennis serve, golf swing, pick up & throw*. Each action was performed 2 or 3 times by each subject. Skeleton joint data of each frame is available having a variety of motions related to arms, legs, torso, and their combinations. In total, there are 567 depth map sequences with a resolution of 320×240.

CAD-60 dataset: This dataset [21] contains 12 actions performed by 4 different subjects (two male and two female, one of them being left-handed) in 5 different environments: office, kitchen, bedroom, bathroom, and living room. The 12 activities are: *rinsing mouth, brushing teeth, wearing contact lens, talking on the phone, drinking water, opening pill container, cooking (chopping), cooking (stirring), talking on couch, relaxing on couch, writing on whiteboard, and working on computer*.

Chalearn dataset: This dataset is a newly released large video database of 13,858 gestures from a lexicon of 20 Italian gesture categories recorded with a Kinect camera, including audio, skeletal model, user mask, RGB and depth images [4]. It contains image sequences capturing 27 subjects performing natural communicative gestures and speaking in fluent Italian, and is divided into development, validation and test parts. We conducted our experiments on the depth images of development and validation samples which contains 11,116 gestures across over 680 depth sequences. Each sequence lasts between 1 and 2 minutes and contains between 8 and 20 gesture samples, around 1,800 frames. Some examples of RGB images are shown in Fig. 3.

4.2. Classification Results

For Chalearn dataset, the classification performance is obtained by means of stratified 5-fold cross-validation. For MSRAction3D dataset, many studies follow the experimental setting of Li et al. [11], such that they first divide the 20 actions into three subsets, each having 8 actions. For each subset, they perform three tests. In test one and two, 1/3 and 2/3 of the samples were used as training samples and the rest as testing samples. In the third test, half of the subjects are used as training and the rest subjects as testing. The experimental results on the first two tests are generally very promising, mainly more than 90% accuracy. On the third test, however, the recognition performance dramatically decreases. It shows that many of these methods do not have good generalization ability when a different subject is performing the action, even in the same environmental settings. In order to have more reliable results, we followed the same experimental setup of [26, 16]. In this setting, actors 1,3,5,7, and 9 are used for training and the rest for testing. It is worth mentioning that is this setting, all 20 actions are classified simultaneously. For CAD-60 dataset, we followed the leave-one-subject-out settings as in [21].

The summaries of the results are reported in Table 1, Table 2, and Table 3 for Chalearn, MSRAction3D, and CAD-60 datasets. In these tables, accuracies of traditional k-NN-based techniques using DTW and LCSS distance measures along with the corresponding accuracies using combined ppfSVMs are reported. It is important to note the outperformance of the results in comparison with the traditional kNN-based classifiers. The result are quite promising, considering the facts that the skeleton tracker sometimes fails and the tracked joint positions are quite noisy.

We then compare our classification results on MSRAction3D and CAD-60 datasets with state-of-the-art methods [2]. Table 4 shows the accuracy of our method, as well as the

[2]Some papers do not follow the standard cross subject settings (e.x.

Table 1. Classification accuracy of different learning strategies on the Chalearn gesture dataset.

	DTW	DDTW	LCSS	DLCSS	Product fusion
Kernel SVM	69.30	71.85	73.05	73.40	**82.60**
kNN	61.11	63.15	67.21	69.18	–

Table 2. Classification accuracy of different learning strategies on the MSRAction3D dataset.

	DTW	DDTW	LCSS	DLCSS	Product fusion
Kernel SVM	80.47	83.84	75.76	76.77	**90.57**
kNN	75.42	77.78	72.05	65.66	–

Table 3. Classification accuracy of different learning strategies on the CAD-60 dataset.

	DTW	DDTW	LCSS	DLCSS	Product fusion
Kernel SVM	73.33	75.00	71.67	70.00	**76.67**
kNN	68.33	68.33	65.00	66.67	–

Table 4. Comparing classification accuracy of our methodology with the state-of-the-art methods on the MSRAction3D and CAD-60 datasets.

MSRAction3D	
	Accuracy
Studies employed depth data	
Action Graph [11]	74.70
HON4D [16]	85.85
Vieira et al. [24]	78.20
Random Occupancy Patterns [25]	86.50
HOPC [17]	91.64
JAS(Cosine)+MaxMin+HOG2 [15]	94.84
DMM-LBP-FF [3]	87.90
Studies employed only skeleton data	
Actionlet Ensemble [27]	88.20
Histogram of 3D Joint [28]	78.97
GB-RBM & HMM [14]	80.20
Points in a Lie Group [23]	89.48
Ensemble classification [2]	84.85
Proposed method	**90.57**
CAD-60	
	Accuracy
Studies employed depth data	
MTO-Sparse coding [13]	65.30
Studies employed only skeleton data	
Actionlet Ensemble [27]	74.70
Sung et al. (2012) [21]	51.30
Proposed method	**76.67**

rival methods on these datasets based on the cross-subject test setting. As can be seen, most studies use depth data in addition to skeleton data; and a few of them have better performance than ours, such as [15] and [17]. However, processing sequences of depth maps is much more computationally intensive. Even though the accuracy of the proposed framework is slightly less than those methods, the advantage of our method is its fast implementation and also do not need fine-tuning of many parameters, which makes it feasible for real-time applications. The training phase of MSRAction3D dataset (including Kernel computation) takes less than a second with a Corei7 CPU and 8 GB of RAM. Most importantly, since kernel computation is based on pairwise distances between samples, it can be easily conducted in parallel. This way, the training phase can be fast on large datasets as well.

The results provided in Table 1 to Table 4 demonstrate the superiority of the proposed methodology. By only considering the skeleton data, the obtained results outperform the best accuracies on MSRAction3D and CAD-60 datasets. Considering the fact that we have only employed the skeleton data, not depth sequences, the results are promising.

they divide the 20 actions into three subsets, each having 8 actions. Therefore, we do not compare our results with those papers

5. Conclusion

In this paper, we tackled the problem of human action classification using the 3D trajectories of body joint positions over the time. To do that, we utilized two time series distance measures, including Dynamic Time Warping and Longest Common subsequences, as well as their derivatives. However, instead of employing these general measures as a distance measure for k-NN, we transformed these measures using the pairwise proximity function in order to be used for powerful SVM classification algorithm. Comparing the recognition results of the proposed methods with state-of-the-art techniques on two action recognition datasets showed significant performance improvements. Remarkably, we obtained 90.57% accuracy on the well-known MSRAction3D dataset using only 3D trajectories of body joints obtained by Kinect.

References

[1] M. A. Bagheri, Q. Gao, and S. Escalera. A framework towards the unification of ensemble classification methods. In *12th International Conference on Machine Learning and Applications (ICMLA)*, volume 2, pages 351–355, 2013.

[2] M. A. Bagheri, G. Hu, Q. Gao, and S. Escalera. A framework of multi-classifier fusion for human action recognition. In *22nd International Conference on Pattern Recognition (ICPR)*, pages 1260–1265, 2014.

[3] C. Chen, R. Jafari, and N. Kehtarnavaz. Action recognition from depth sequences using depth motion maps-based local binary patterns. In *WACV*, pages 1092–1099, 2015.

[4] S. Escalera, J. Gonzlez, X. Bar, M. Reyes, O. Lopes, I. Guyon, V. Athistos, and H. Escalante. Multi-modal gesture recognition challenge 2013. In *ICMI*, 2013.

[5] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon. Efficient regression of general-activity human poses from depth images. In *ICCV*, pages 415–422, 2011.

[6] T. Górecki. Using derivatives in a longest common subsequence dissimilarity measure for time series classification. *Pattern Recognition Letters*, 45:99–105, 2014.

[7] T. Graepel, R. Herbrich, P. Bollmann-Sdorra, and K. Obermayer. Classification on pairwise proximity data. *NIPS*, pages 438–444, 1999.

[8] S. Gudmundsson, T. P. Runarsson, and S. Sigurdsson. Support vector machines and dynamic time warping for time series. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 2772–2776, 2008.

[9] G. Johansson. Visual perception of biological motion and a model for its analysis. *Perception & psychophysics*, 14(2):201–211, 1973.

[10] E. J. Keogh and M. J. Pazzani. Derivative dynamic time warping. In *SDM*, volume 1, pages 5–7, 2001.

[11] W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *CVPRW*, pages 9–14. IEEE, 2010.

[12] O. L. Mangasarian. Generalized support vector machines. *NIPS*, pages 135–146, 1999.

[13] B. Ni, P. Moulin, and S. Yan. Order-preserving sparse coding for sequence classification. In *ECCV*, pages 173–187. Springer, 2012.

[14] S. Nie and Q. Ji. Capturing global and local dynamics for human action recognition. In *ICPR*, pages 1946–1951, 2014.

[15] E. Ohn-Bar and M. M. Trivedi. Joint angles similarities and hog2 for action recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pages 465–470. IEEE, 2013.

[16] O. Oreifej, Z. Liu, and W. Redmond. Hon4d: Histogram of oriented 4D normals for activity recognition from depth sequences. In *CVPR*, 2013.

[17] H. Rahmani, A. Mahmood, D. Q. Huynh, and A. Mian. Hopc: Histogram of oriented principal components of 3d pointclouds for action recognition. In *ECCV*, pages 742–757. 2014.

[18] M. Reyes, G. Dominguez, and S. Escalera. Feature weighting in dynamic timewarping for gesture recognition in depth data. In *CVPRW*, pages 1182–1188. IEEE, 2011.

[19] B. Schölkopf and A. J. Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[20] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR*, 2011.

[21] J. Sung, C. Ponce, B. Selman, and A. Saxena. Unstructured human activity detection from RGBD images. In *ICRA*, pages 842–849, 2012.

[22] A. Treisman and H. Schmidt. Illusory conjunctions in the perception of objects. *Cognitive psychology*, 14(1):107–141, 1982.

[23] R. Vemulapalli, F. Arrate, and R. Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *CVPR*, pages 588–595, 2014.

[24] A. W. Vieira, E. R. Nascimento, G. L. Oliveira, Z. Liu, and M. F. Campos. Stop: Space-time occupancy patterns for 3d action recognition from depth map sequences. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 252–259. Springer, 2012.

[25] J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu. Robust 3D action recognition with random occupancy patterns. In *ECCV*, pages 872–885. Springer, 2012.

[26] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *CVPR*, pages 1290–1297, 2012.

[27] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Learning actionlet ensemble for 3d human action recognition. *PAMI*, 36(5):914–927, 2014.

[28] L. Xia, C.-C. Chen, and J. Aggarwal. View invariant human action recognition using histograms of 3d joints. In *CVPRW*, pages 20–27, 2012.

[29] X. Yang and Y. Tian. Eigenjoints-based action recognition using naive-bayes-nearest-neighbor. In *CVPRW*, pages 14–19.