Human Action Recognition using Recursive Self Organizing Map and Longest Common Subsequence Matching

Wei Huang, Jonathan Wu
Department of Electrical and Computer Engineering, University of Windsor
Windsor, Ontario, Canada
{huang1m, jwu}@uwindsor.ca

Abstract

A little attention has been given to the use of Recursive Self Organizing map (SOM) for human action recognition in the past years. This paper introduces an action recognition framework using the Recursive SOM, a temporal extension of SOM that learns adapted representations of temporal context associated with a time series. We demonstrate the effectiveness of Recursive SOM for data clustering, dimensionality reduction and context learning in human action recognition. The atomic poses in motion sequences and their contextual information are extracted and encoded by the trained Recursive SOM. A human action sequence is represented as a trajectory of map units. To classify a new action, a longest common subsequence algorithm using dynamic programming is employed to robustly match action trajectories on the map. To the best of our knowledge, we are the first to try Recursive SOM approach for human action recognition. We test the approach on a well known benchmark action dataset and achieve promising results.

1. Introduction

Human action recognition in video sequences is one of the most challenging recognition problems in computer vision. There are many important applications that motivate the research in this area, such as video surveillance, video content retrieval, human-computer interaction, analysis of sports events, etc.

For recognizing human action, a set of features need to be extracted from videos and used for training classifiers. Therefore, an important consideration is what feature representations are robust to model the action sufficiently. Approaches can be roughly grouped as interest point based [1, 2, 3], space-time shape based [4, 5], optical flow based [6] and human silhouette based [7]. [1] used local space-time features to represent motion patterns and combine such descriptors with SVM classification schemes for the recognition task. 2D Harris corner detector was extended to a 3D Harris detector to detect local regions where the image intensities have high variations in both space and time. However, it usually gives insufficient spatio-temporal corners in certain motions [2]. [2] developed an alternative detector which combined Harris corner detector with Gabor filters to extract more spatio-temporal interest points. One problem with the above detectors which use local information only is that false detections may be generated due to noises instead of the true motion. [3] used global information from each video input for detecting interest points in regions containing the relevant motion. Unlike interest point based approaches, [4] extended a 2D shape representation method to 3D space-time volumes by adding time domain to a Poisson equation. Space-time shape features were extracted and used to represent an action. [5] generated a spatiotemporal volume (STV) from a sequence of 2D contours and described actions by analyzing the differential geometric properties of STV. [6] presented a motion descriptor based on explicit computation of optical flow to recognize human actions at low resolutions. Another line of work is to use human silhouettes as features for human action description [7]. Silhouettes provide rich and strong body shape information and are also easy to obtain in many scenarios, especially in the case of stationary cameras.

While features described in a high dimensional space are very discriminative, the computation for data analysis is expensive. Methods such as data clustering and dimensionality reduction are therefore desirable. An important requirement is that these methods should be able to maintain the power of higher dimensional descriptions as much as possible. Furthermore, the context in both spatial and temporal domain which is inherent in human action should not be lost after transformation. [8] used a “bag of words” model to cluster local features into spatio-temporal codewords. While the “bag of words” model has demonstrated the effectiveness in action classification, both geometric information among different body parts (spatial context) and temporal patterns across local features (temporal context) are discarded. [9] showed an example where a similar distribution of video words was shared by two different action classes. To enhance the classification capability, [9] introduced spatio-temporal correlograms to model the temporal correlation patterns among local codewords in different action classes. [10] proposed a constellation-of-bags-of-features model to exploit both the

978-1-4244-5498-3/09/$25.00 ©2009 IEEE

geometric power of the constellation model as well as the richness of the “bag of words” model. The constellation model described the mutual geometric relationship among different body parts. A “bag of words” model was attached to each part of the constellation model. [7] considered human action video sequences as data points on nonlinear dynamic shape manifolds. A novel manifold embedding method, called Local Spatio-Temporal Discriminant Embedding (LSTDE) was proposed to map data points to the low-dimensional embedding space. After manifold embedding, both the local spatial and local temporal discriminant structures of the data can be discovered effectively. [11] introduced the theory of chaotic systems to model and analyze nonlinear dynamics of human actions. A delay-embedding scheme was used to reconstruct a phase space of appropriate dimension. [12] fused multiple features for action recognition using Fiedler Embedding. The Fiedler Embedding scheme embedded different features into a common low-dimensional Euclidian space.

After the human action representations have been generated, both discriminative classifiers (for example, SVM [1]) and generative classifiers (for example, HMM [13][14] and pLSA [8] [9]) can be employed in recognition of human actions. While HMM and its variants have been widely used to express the temporal relationships inherent in human actions, HMM models very likely suffer from over-fitting due to the small training data.

In this paper, we introduce a neural network framework for human action recognition. Specifically, the Recursive Self-Organizing Map, as a temporal extension of SOM is employed to learn adapted representations of temporal context associated with human action sequences. Sequences of human silhouettes are selected as basic feature representations and act as the inputs to the Recursive SOM. The atomic poses in motion sequences and their contextual information are extracted and encoded by the trained Recursive SOM. A human action sequence is therefore represented as a trajectory of map units. To classify a new action, a longest common subsequence algorithm using dynamic programming is employed to robustly match action trajectories on the map. To the best of our knowledge, we are the first to try Recursive SOM approach for human action recognition.

2. Proposed algorithm

In the following sections we describe our algorithm in details. Section 2.1 describes the feature representation method by using sequences of human silhouettes. Section 2.2 and Section 2.3 detail Self-Organizing Map and Recursive Self-Organizing Map respectively. Section 2.4 presents the action recognition scheme based on longest common subsequence algorithm. Experimental results are given in Section 3. Section 4 concludes the proposed approach.

2.1. Feature representation by sequences of human silhouettes

This step aims to extract sequences of human silhouettes to describe human action in raw video. A human action sequence can be regarded as a temporal series in which human silhouettes continuously change over time. The human body silhouette represents the pose of the human body at any instant in time. Temporal variations of silhouettes implicitly characterize motion kinematics. There are several advantages of using silhouettes: 1) silhouettes are intuitive while containing rich shape information; 2) a series of body silhouette images can be used to recognize human actions correctly regardless of the speed of movement; 3) binary silhouettes are robust to variations in clothing and lighting condition; and 4) silhouettes are easier to extract compared with tracking body parts.

We do not intend to address the foreground detection issue in this work, so the silhouette masks provided in [4] are directly used in our experiments. To cope with size and position variation, silhouette images are centered and normalized by keeping the aspect ratio of the silhouettes so that the resulting images do not distort the moving shape and have same dimensions. Accordingly, each 2D silhouette image can be converted to a 1D vector with the same dimension in a row-scan manner. The silhouette sequences are then represented as: $X_k = \{x_1, x_2, ..., x_{N_k}\}$, where $x_i (i = 1,2,..., N_k)$ is 1D vectors and N_k is the length of the sequence. Some normalized example silhouettes of different actions performed by the subject “daria” in the Weizmann dataset are shown in Figure 1.

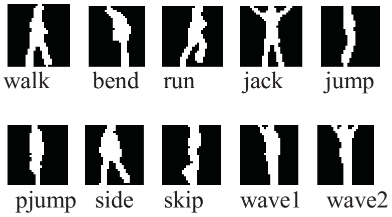

Figure 1. Normalized example silhouettes of different actions performed by the subject “daria”.

It is formidable to learn the action characteristics by analyzing data in the high dimensional image space. Therefore, the goal is to transform the high dimensional data to low dimensional data via schemes such as data clustering and dimensionality reduction. Furthermore, the temporal context which is inherent in human action should be maintained after transformation. In this paper, we introduce a Recursive Self-Organizing Map which is a temporal extension of SOM to cluster data, to reduce dimensionality, and to maintain temporal context in the action sequences. The dimension reducing property of the SOM allows replacing the original high dimensional data points by 2-dimensional points. At the same time the neighborhood preservation property of the SOM allows to utilize the mapped data points in a subsequent trajectory comparing stage. The standard SOM is not able to process temporal sequence data. To solve this problem, we employ a temporal extension of SOM, the Recursive SOM, to maintain temporal context and discover temporal patterns in temporal sequences.

2.2. Self-organizing map

The Self-Organizing Map (SOM) was introduced by T. Kohonen [15]. The SOM is one of the most popular unsupervised learning neural networks and has been successfully applied as a data mining tool with various applications in pattern recognition and image analysis. The advantages of the SOMs are that they can project high dimensional data to lower dimensional data, preserve the topology of the data space, find similarities in the data, and well approximate the distribution density of the learning data.

The SOM quantizes the data space formed by the training data and simultaneously performs a topology-preserving projection of the data onto a regular low-dimensional lattice. The lattice can be used efficiently in visualization. The map is generated by establishing a correspondence between the input data and neurons located on the lattice. The correspondence is obtained by a competitive learning algorithm. The weight vectors of the neurons are modified iteratively in the training steps. When a new input arrives every neuron competes to represent it. The best matching unit (BMU) is the neuron that wins the competition. The BMU with its neighbors on the lattice are allowed to learn the input. Neighboring neurons will gradually specialize to represent similar inputs, and the representations will become ordered on the map lattice. The best matching unit *b(n)* is the weight vector $\omega_j(n)$ that is nearest to the input *x(n)* and is obtained by the following metric:

$$b(n) = \arg\min \left\| x(n) - \omega_j(n) \right\| \tag{1}$$

In general, the Euclidean distance is used. The winning unit and its neighbors adapt to represent the input by modifying their weight vectors towards the current input:

$$\Delta\omega_j = \gamma h_{jk}(x(n) - \omega_j) \tag{2}$$

where γ is a learning rate which is a monotonically decreasing function with respect to time. h_{jk} is a neighborhood kernel function of the distance between units *j* and *k* on the map. In practice the neighborhood kernel is chosen to be wide in the beginning of the learning process to guarantee global ordering of the map, and its width decreases slowly during learning.

The Self-Organizing Map method makes the mapping statically. Then the order of the pattern vectors does not matter in the representation. In many real world applications, e.g., speech recognition and motion detection/analysis, the order of patterns is important. In some cases the order may even be crucial for the good performance of the system. It would be preferable if the order could be included into the representation.

2.3. Recursive self-organizing map

As discussed above, the original SOM method is designed only to represent the topology nature of the data. It cannot model the sequential aspects of the input stream. The basic SOM is indifferent to the ordering of the input patterns. Real data, however, is often sequential in nature thus temporal context of a pattern may significantly influence its correct interpretation.

Since the SOM is quite popular in data mining applications, where it is primarily used for visualization and clustering, a number of approaches based on SOM have been proposed to effectively account for temporal or other context of patterns. In order to store the temporal context in the time series, [16] proposed a modification to the original SOM. This modified SOM, called Recursive Self-Organizing Map, can not only function like a normal SOM but is also able to describe context between vectors appearing in sequences.

In the Recursive SOM, each unit *j* has a receptive field defined by two weight vectors, $\omega_j^x \; and \; \omega_j^y$, that are compared to the input vector, *x(n)*, and to the vector of activities in the map at the previous time, *y(n-1)*, respectively. The activity, $y_j(n)$ of unit j at step *n* is:

$$y_j(n) = \exp(-\alpha \left\| x(n) - \omega_j^x \right\|^2 - \beta \left\| y(n-1) - \omega_j^y \right\|^2) \tag{3}$$

where α and β are constant coefficients, and the norm is Euclidean. The best matching unit is the unit that has the highest activity. If *k* is the index of the unit maximizing $y_j(n)$, the learning rules used for the weights are:

$$\Delta\omega_j^x = \gamma h_{jk}(x(n) - \omega_j^x) \tag{4}$$

$$\Delta\omega_j^y = \delta h_{jk}(y(n-1) - \omega_j^y) \tag{5}$$

Where γ and δ are learning rates, and the neighborhood function h_{jk} is a Gaussian function of the Euclidean distance between units j and k on the map:

$$h_{jk} = \exp(-d(j,k)^2 / \sigma^2) \qquad (6)$$

where σ controls the width of the Gaussian.

Note that this algorithm applies the original SOM algorithm to both vectors ω_j^x *and* ω_j^y simultaneously. During training, each unit learns to associate an input vector *x(n)* with a pattern of activity *y(n-1)*, representing its temporal context. Figure 2 shows the architecture of the Recursive SOM.

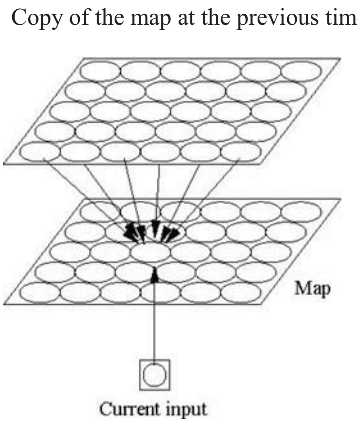

Figure 2. The architecture of the recursive SOM.

The Recursive SOM constitutes a robust temporal extension of SOM, which successfully generalizes the properties of the original SOM to time series, ie topology preservation and vector quantization. The chains of the best matching units (BMUs) for silhouette sequences produce time varying trajectories of activity on the Recursive SOM. These trajectories are therefore employed for action classification. Figure 3 illustrates the transformation of a silhouette sequence into a trajectory on the map.

2.4. Action recognition using longest common subsequence scheme

After trajectories on the Recursive Self-Organizing Map are obtained, we are ready to implement the recognition task. An action sequence has been mapped to a trajectory on the map. When action sequences of same class are input, the trajectories they produce are similar. Several schemes such as DTW, longest common subsequence (LCS), Hausdorff distance have been compared for learning trajectories [17]. A desirable recognition approach should maintain the temporal order of the action sequences as well as allow for some degree of tolerance in terms of partial matches. Such considerations motivate us to propose the action recognition algorithm based on the LCS method. LCS is used to compare two sequences by finding a longest subsequence of the two sequences: the larger the length of the common subsequence, the more similar the two sequences are. An important aspect of LCS is that the subsequence does not need to be contiguous, only the order of the subsequences is maintained. This property enables the LCS strike a good balance between the rigid temporal order constraint based methods and bag-of-words based methods which lose temporal information.

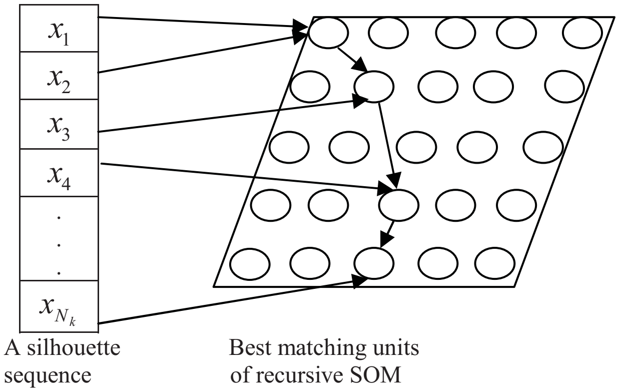

Figure 3. Transformation of a silhouette sequence into a trajectory on the map.

Given two sequences *P* and *Q,* a brute-force approach to solving the LCS problem is to enumerate all subsequences of *P* then check each of them to see if it is a subsequence of *Q*, at same time keeping the longest one. This naive implementation of the longest common subsequence algorithm will result in exponential computational complexity. As explained in [18], the LCS can be computed efficiently with dynamic programming. Let $P = \{p_1, p_2, ..., p_m\}, Q = \{q_1, q_2, ..., q_n\}$. We need to find an LCS of these two sequences. If $p_m = \mathrm{q_n}$, we need find an LCS of P_{m-1} and $\mathrm{Q_{n-1}}$. Adding the number of 1 to this LCS gives an LCS of P and *Q*. If $p_m \neq \mathrm{q_n}$, we need solve two subproblems: finding an LCS of P_{m-1} and Q and finding an LCS of P and $\mathrm{Q_{n-1}}$. Whichever of these two LCS is longer is an LCS of *P* and *Q*.

We define *c[i, j]* to be the length of an LCS of the sequences P_i and $\mathrm{Q_j}$, then the optimal substructure of the LCS problem gives the following recursive formula:

$$c[i,j] = \begin{cases} 0 & \text{if } \mathrm{i} = 0 \text{ or } \mathrm{j} = 0 \\ c[i-1, j-1] + 1 & \text{if } \mathrm{i, j} > 0 \text{ and } \mathrm{p_i} = q_j \\ \max(c[i, j-1], c[i-1, j]) & \text{if } \mathrm{i, j} > 0 \text{ and } \mathrm{p_i} \neq q_j \end{cases} \qquad (7)$$

The values of *c[i, j]* are stored in a table *c*. Another table *b* is maintained to simplify construction of an optimal solution. The table *c* has the size of m*n and returns the length of an LCS of *P* and *Q*. Using dynamic programming, the LCS problem can be solved efficiently.

3. Experimental results

We evaluate our action recognition algorithm using the Weizmann human action dataset from [4]. This dataset contains 10 action classes performed by nine different persons. There are totally 90 video sequences in our experiments. In this dataset, actors have different physical sizes, wear different clothing and perform actions differently both in speeds and motion styles. We adopt a leave-one-out scheme for evaluation. In the leave-one-out strategy, for each type of actions, the video sequences of one person are selected as the test data and the rest as the training data. We repeat this experiment for 40 times and obtain the average recognition rate. The parameters of the Recursive SOM are listed in Table 1. The Recursive SOM is trained during 4000 iterations. The normalized silhouette images extracted in the feature representation stage have the same 31x31dimention and are transformed to trajectories on the 2D Recursive SOM. The LCS scheme implemented by dynamic programming is thus used to compare each pair of the motion trajectories of map units and to classify the corresponding human action sequence. We reach 96.5% average recognition rate for all actions, using a 10x10 Recursive SOM. The corresponding confusion table generated by our classification results is shown in Table 2.

In our experiments, we also explore the effect of different Recursive SOM sizes on the recognition performance. Figure 4 shows the performance variation with various map sizes such as 4x4, 5x5, 6x6, and so on. We compare our average recognition rates with other published results on the same dataset in Table 3. From this table, we can see that the performance of the proposed Recursive SOM based algorithm is comparable to other approaches.

4. Conclusion

In this paper we introduced an action recognition framework using Recursive SOM, the temporal extension of SOM that learns adapted representations of temporal context associated with time sequences. We demonstrated the effectiveness of Recursive SOM for data clustering, dimensionality reduction and context learning in human action recognition. The atomic poses in motion sequences and their contextual information were described by the trained Recursive SOM. The human action sequences were thus represented as trajectories of map units. To classify a new action, a longest common subsequence algorithm using dynamic programming was employed to robustly match these action trajectories on the map. We obtained promising experimental results by testing the approach on a well known standard action dataset.

Table 1. The parameters of the Recursive SOM.

Parameter	Value
α	3
β	1
Initial learning rate γ	0.1
Initial learning rate δ	0.1
σ	5

Table 2. The confusion table of action recognition by the proposed algorithm using 10x10 Recursive SOM.

	Walk	Bend	Run	Jack	Jump	Pjump	Side	Skip	Wave 1	Wave 2
Walk	100									
Bend		100								
Run			100							
Jack				100						
Jump					100					
Pjump						100				
Side							100			
Skip	5		17.5		2.5			75		
Wave 1						2.5			97.5	
Wave 2									7.5	92.5

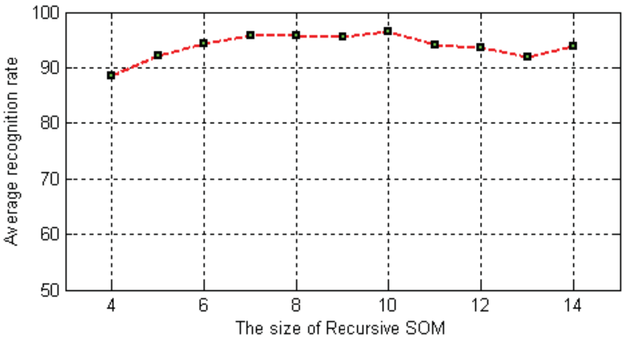

Figure 4. The effect of different Recursive SOM sizes on the recognition performance.

Table 3. Comparison between our approach and others on the Weizmann dataset.

	Blank et al. [4]	Jia et al. [7]	Niebles et al. [10]	Ali et al. [11]	Liu et al. [12]	Our method
Average Recognition Rate (%)	99.61	90.91	72.8	92.6	90.4	96.5

References

[1] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local SVM approach. In ICPR, pages III: 32–36, 2004.

[2] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In ICCV Workshop: VSPETS, 2005.

[3] Shu-Fai Wong; Cipolla, R., "Extracting Spatiotemporal Interest Points using Global Information," ICCV 2007.

[4] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In ICCV, pages 1395–1402.

[5] A. Yilmaz and M. Shah. Action sketch: A novel action representation. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, volume 1, pages 984 –989,2005.

[6] A. A. Efros, A. C. Berg, G.Mori, and J.Malik. Recognizing action at a distance. In Proceedings of the Ninth IEEE International Conference on Computer Vision, volume 2, pages 726–733, 2003.

[7] Kui Jia and Dit-Yan Yeung. Human action recognition using Local Spatio-Temporal Discriminant Embedding. In CVPR 2008.

[8] J. C. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatio-temporal words. In Proc BMVC, 2006.

[9] J.C. Niebles S. Savarese, A. Del Pozo and L. Fei-Fei. Spatial-temporal correlations for unsupervised action classification. In IEEE Workshop on Motion and Video Computing. Copper Mountain, 2008.

[10] J. Niebles and L. Fei-Fei, A hierarchical model of shape and appearance for human action classification. In CVPR, 2007.

[11] S. Ali, A. Basharat, and M. Shah. Chaotic invariants for human action recognition. In Proc ICCV, 2007.

[12] J. Liu, S. Ali, and M. Shah. Recognizing human actions using multiple features. In CVPR, 2008.

[13] Yamato, J., Ohya, J., Ishii, K.: Recognizing human action in time-sequential images using hidden Markov model. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, pp. 379–385 (1992).

[14] Mori, T., Segawa, Y., Shimosaka, M., Sato, T.: Hierarchical recognition of daily human actions based on continuous hidden markov models. Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition (2004) 779 – 784.

[15] T. Kohonen. Self-Organizing Maps. Springer-Verlag, 1995.

[16] T. Voegtlin. Recursive self-organizing maps. Neural Networks, 15(8-9):979–991, 2002.

[17] Z.Zhang, K.Huang and T.Tan, Comparison of Similarity Measures for Trajectory Clustering in Outdoor Surveillance, International Conference on Pattern Recognition, Vol. 3, pp. 1135-1138, 2006.

[18] Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. MIT Press and McGraw-Hill (2001).