Gesture and Action Recognition by Evolved Dynamic Subgestures

Víctor Ponce-López [1,2,3]
http://victorponce.org

Hugo Jair Escalante [4]
http://hugojair.org/

Sergio Escalera [2,3]
http://www.maia.ub.es/~sergio/

Xavier Baró [1,2,3]
https://xbaro.wordpress.com/

[1] IN3, Open University of Catalonia
Rambla Poblenou, 156
08018 Barcelona, Spain

[2] Dept. MAiA, University of Barcelona
Gran Via de Les Corts Catalanes, 585
08007, Barcelona, Spain

[3] Computer Vision Center
Building O, Campus UAB
08193, Bellaterra, Spain

[4] Computational Sciences Dept. INAOE
Luis Enrique Erro, 1
72840, Tonantzintla, Puebla, Mexico

Abstract

This paper introduces a framework for gesture and action recognition based on the evolution of *temporal* gesture primitives, or *subgestures*. Our work is inspired on the principle of producing *genetic* variations within a population of gesture subsequences, with the goal of obtaining a set of gesture units that enhance the generalization capability of standard gesture recognition approaches. In our context, gesture primitives are evolved over time using *dynamic programming* and *generative models* in order to recognize complex actions. In few generations, the proposed subgesture-based representation of actions and gestures outperforms the state of the art results on the MSRDaily3D and MSRAction3D datasets.

1 Introduction

Gesture and human action recognition are two widely studied topics in computer vision. Great advances have been reported in the last few years [1], mainly boosted by the release of the Kinect sensor [27]. Most of existing recognition methods learn gesture/action models that attempt to capture and recognize whole gestures (*i.e.*, an holistic approach). Classical approaches under this scheme are those based on dynamic time warping (DTW) [3] and hidden Markov models (HMM) [28, 34].

Although the previous methods have obtained high performance in several domains, recent research is moving towards approaches that model the problem in terms of gesture primitives (subgestures) [14, 19, 20, 23, 33]. The underlying assumption of this type of methods is that whole gestures are composed by primitives (that can be shared or not among gestures from different categories), and the hypothesis is that learning with primitives leads

to better recognition performance. Whereas the subgesture-based techniques have proved to be successful, it remains open the question on how to define/learn subgestures and, more importantly, how to perform inference using subgesture models.

This paper describes a novel approach for human action and gesture recognition based on subgesture modeling. Unlike other primitive modeling approaches, our proposal learns subgestures by searching for temporal patterns that improve recognition performance when used to represent and classify complex gestures and actions. An evolutionary algorithm is implemented for this purpose, with adhoc variation operators suitable for learning primitive recognizers of actions/gestures. This algorithm takes as reference two standard methods for learning from sequential data: DTW and HMMs. Besides learning the primitives from scratch, it determines the inference procedures for DTW and HMM when using subgestures. The proposed framework is evaluated in MSRDaily3D and MSRAction3D datasets, outperforming state of the art results.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 describes the evolutionary framework for learning subgestures. Next, Section 4 reports experimental results obtained with the evolved subgesture models. Finally, Section 5 summarizes our findings and outlines future work directions.

2 Related Work

Traditional gesture recognition methods were based on templates (*e.g.*, MHIs [2]), sequence alignment (*e.g.*, DTW [3]) or statistical sequential-modeling (*e.g.*, HMMs [28, 34]). Because of its effectiveness, DTW and HMM based methods are still among the most used techniques nowadays [13, 18, 21]. DTW-based methods align, via dynamic programming, sequences of different length to reference gesture models. The goal is to find the alignment that minimizes a cost given by a distance measure between elements of the sequences. HMMs on the other hand, are generative models, typically applied to sequential decision problems. Observations sequences are assumed to be generated by a *hidden* stochastic process.

Despite its effectiveness, traditional gesture recognition methodologies approach the problem in a holistic way, where gestures are processed as a whole. Results in related fields with part-based techniques (*e.g.*, in object detection [8] and action recognition [26]) have inspired researchers to build solutions based on subgesture models. For instance, in [19] HMMs based on subgestures were proposed. However, subgestures were manually provided by the users. In [20] a HMM was used to learn subgestures, althought the model was only applied to the problem of hand gesture recognition. In [29] it was proposed a method for segmenting gestures into subgesture units at the frame level. In [23] the authors proposed to use DTW for subgesture modeling, but no results were reported. In [4] subgesture units (defined as cuboids) were learned together with their relationships (using Allen's relations) under a graph-learning framework. Recently, in [33] a relational model for action recognition using dynamic-keyposes was proposed.

Some methods base on key pose/frame extraction [17, 25, 36] in order to learn a subset of key frames that are highly representative and discriminative for an action class. In [36] an information-theory criterion is adopted for selecting keyframes, whereas in [17] it is used a boosted-based criterion. In [25] a max-margin formulation of the problem is proposed. Very recently, evolutionary algorithms have been also developed for keyframe extraction [5, 6]. In these works, a bag-of-key-poses representation was adopted and an evolutionary algorithm was used to select the number of key-poses for the vocabulary (using $k-$means for cluster-

ing), the training set, features and parameters of the model (using DTW for recognition). All of these methods look for a subset of frames, whereas in subgesture modeling we aim at learning spatio-temporal units (subgestures). On the other hand, the above works assume and demonstrate that class-specific key poses/subgestures give a good performance. Nevertheless, we include the fact that some classes may contain or share similar subgestures [23]. Under this additional assumption, our method also reaches the state of the art performance and provides considerable improvements in gesture and action recognition domains.

In this work, a genetic algorithm is used to evolve gesture primitives integrated into an action recognition framework coupled with either DTW or HMMs. Different from most of the work reviewed in this section, our approach obtains dynamic subgestures (*i.e.*, sequences of frames of different lengths) and simultaneously learns the parameters of the recognition model (either DTW or HMM).

3 Training Dynamic Subgestures

This section describes the methodology to automatically learn gesture primitives (hereinafter referred to as subgestures). Consider a training data set $X^T = \{x_1^T, x_2^T, ..., x_n^T\}$, where each $x^T \in X^T$ is a *sequence* example of a gesture. Similarly, consider a validation data set $X^V = \{x_1^V, x_2^V, ..., x_m^V\}$ of gesture sequence examples. Both X^T and X^V are subsets of a data set, whose sequence examples belong to different classes $C = \{c_1, c_2, ..., c_g\}$. Our goal is to find a subgesture set $S = \{s_1, s_2, ..., s_k\}$ from X^T, being s_i a sequence representation of the subgesture i, that maximizes the recognition performance of gestures in X^V, given a particular gesture recognition method (see Algorithm[1] 1).

Data: Population P; Training data X^T; Validation data X^V
Result: Models of k subgestures S for each individual and its score
Current generation:
foreach *new unique valid I in the population P* **do**
  $k, segments \leftarrow decode(I)$;
  $X^T_{seg} \leftarrow getDataPartitions(X^T, segments)$;
  $S \leftarrow k\text{-}meansDTW(X^T_{seg}, k)$; `// Section 3.2`
  if *use dynamic programming* **then**
    $R \leftarrow getResizedClassModels(X^T, S)$; $\mathbf{D} \leftarrow getDissimilarities(S)$;[1]
    $M \leftarrow getUpdatedCosts(R, S)$; $D^V \leftarrow getUpdatedCosts(X^V, S)$;[1] `// Figure 2`
    $\omega \leftarrow addParamsToStruct(M, \mathbf{D})$;
  else if *use generative model* **then**
    $D^T \leftarrow getUpdatedCosts(X^T, S)$; $D^V \leftarrow getUpdatedCosts(X^V, S)$;[1] `// Figure 2`
    $M \leftarrow learnGM(D^T, \omega)$;
    $\omega \leftarrow addParamsToStruct(M)$;
  end
  $s, \omega^* \leftarrow g(D^V, \omega)$; `// Section 3.3`
end

Algorithm 1: Pseudocode for learning Subgesture Models at each generation

3.1 Evolutionary Optimization

Let $P = \{I_1, I_2, ..., I_l\}$ be a population of l individuals, each one composed of $1 + 2N$ genes. The first *gene* refers to the number k of subgestures and the remainder $2N$ genes refer to pairs of start-length segments from X^T. Initially, there is a probability P_s of generating

[1] The lines having more than one instruction in Algorithm 1 can be computed in parallel.

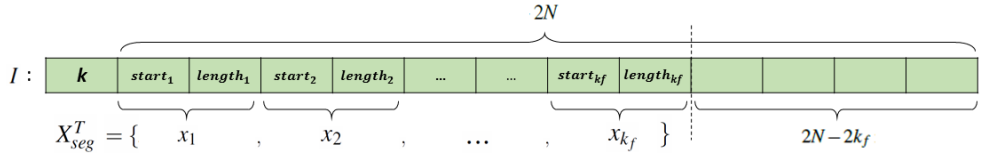

Figure 1: Representation of an individual I formed by $1+2N$ genes: k for the number of subgestures, X^T_{seg} the set of k_f pair-wise generated segments, and $2N-2k_f$ empty genes.

each pair-wise *segment*. Those candidate segments are generated via a *random selection* over the whole continuous sequence X^T (*i.e.* the concatenation of all training sequences), ensuring that the length of each possible segment is within $[n_{min}, n_{max}]$ frames. Thus, each individual I has $k_f \leq N$ pair-wise generated segments. Finally, the value of the first gene is randomly chosen between the range $[k_0, k_f]$, so that $k_0 \leq k \leq k_f$. It means that the number of k allowed *clusters is set depending on the generated segments.* The training procedure ignores the remaining $2N-2k_f$ empty segments in the fitness function. Figure 1 shows the representation of an individual.

3.1.1 Fitness function

The goal of the proposed genetic algorithm is to maximize the *score* given by the evaluation function, described in Section 3.3. It consists of obtaining a measure of performance for the learned *models*, expressed in terms of subgestures, over validation sequences in the classification task. Section 3.2 provides details of the *aligned* temporal clustering method developed to obtain subgestures. Once subgestures are computed, we provide either dynamic programming or generative model approaches to learn and evaluate the subgesture models.

Dynamic Programming: As presented in Algorithm 1, we obtain a model for each class represented in subgestures. Each subgesture within the set $S = \{s_1, s_2, ..., s_k\}$ is the centroid sequence obtained from the $k-meansDTW$ algorithm. Therefore, we design each class model $m_c \in M$, where M is the set of class models, by 1) computing $r_c \in R$ as the mean of all resized training samples of each class, where R is the set of all resized training samples, and 2) representing r_c in subgestures. This procedure is done by means of a backward loop over the DTW warping paths (see Figure 2). On the other hand, we compute the dissimiliarity matrix as:

$$\mathbf{D} = \mathbf{W} + \mathbf{W}^{\intercal}, \quad \text{s.t.} \quad \mathbf{W} = \frac{1}{\gamma} \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1k} \\ w_{21} & w_{22} & \dots & w_{2k} \\ \cdot & \cdot & \dots & \cdot \\ w_{k1} & w_{k2} & \dots & w_{kk} \end{bmatrix}, \tag{1}$$

where $\mathbf{W}$ is a squared matrix obtained from aligning all subgestures among them. This is, to compute each element as the DTW cost by:

$$w_{ij} = DTW(s_i, s_j) = min_{\Omega}\{\sum_{p=1}^{P} d_p, \Omega = < v_1, v_2, ..., v_P >\}, \tag{2}$$

where d_p is the euclidean distance between feature vectors $s_i^{\dot{x}}$ and $s_j^{\dot{y}}$ given the coordinates of the warping path $v_p = (\dot{x}, \dot{y})$ in Ω. Then, each element of the matrix $\mathbf{W}$ is normalized *w.r.t.* the maximum cost value γ of all elements $w \in \mathbf{W}$. To express both each class representative

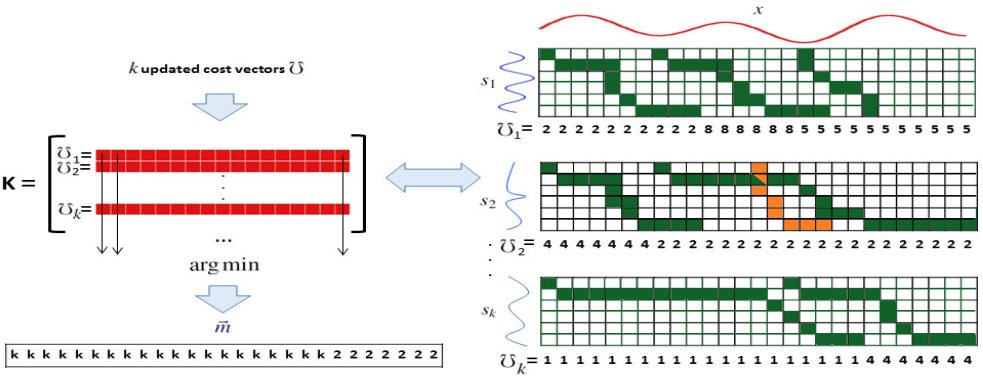

Figure 2: Graphical example of the computation of KM and KT from the backward loop over the DTW warping paths Ω (best seen in color). Input sequence x is aligned to the subgesture sequences from S so as to obtain the k updated cost vectors $\mho$ that construct the matrix K. In the different modules of our approach we use the common DTW alignment that initializes the first row and column of the DTW matrix as infinity, which indicates that the warping path goes from the first position until the last position of the DTW matrix. In this current step, however, to compute each $\mho$, we initialize first DTW row to zeros to compute multiples warping paths and perform backward search, starting from the last position of last row and stopping when the path reaches the first row. While computing the backward path, we prioritize the left-steps when the neighbor positions have same cost values, in order to maximize the length of the paths. Finally, we assign to each position of $\mho$ the minimum cost values of the paths found that involves that position. We refer to matrix K as KM and KT when input sequences are from the training and validation (or test) set, respectively. The vector $\vec{m}$ is the final input sequence represented in subgestures, *i.e.* from the arguments obtained in Eq. 3. Figure 4 shows two real examples of $\vec{m}$, identifying subgestures in real skeleton-based gesture sequences.

sequence r_c and each validation sequence x^V in terms of subgestures, we assign to each frame f the subgesture identifiers that give the minimum costs, respectively, as:

$$i = \arg\min(\vec{km}^f) \quad , \quad j = \arg\min(\vec{kt}^f); \tag{3}$$

where $\vec{km}^f$ and $\vec{kt}^f$ are vectors of length k subgestures corresponding to the *columns* of the cost matrices KM and KT for the current training and validation sequences, respectively (see description in Figure 2). Therefore, the set of arguments $i^* = \arg\min(\vec{km})$ and $j^* = \arg\min(\vec{kt})$ are the subgesture identifiers that construct the class models $m_c \in M$ and the validation sequences $d^V \in D^V$ in terms of subgestures. Then, final evaluation is obtained as:

$$DTW(m_c, d^V) = min_\Omega\{\sum_{\rho=1}^{\wp} \mathbf{D}(i_\rho, j_\rho), \Omega = < v_1, v_2, ..., v_\wp >\}. \tag{4}$$

Note that the expression of Eq. 4 takes the same form as Eq. 2, but instead of using the euclidean distance, each distance $\mathbf{D}(i_\rho, j_\rho)$ in the DTW warping path considers the similarities among subgestures.

Generative models: Still looking at algorithm 1, our generative model deals with $1D$ discrete sequences. The first step is thus to obtain discrete representations of training and validation sequences. Similarly to the DTW approach and the Figure 2, we represent each training and validation sequence in terms of subgestures using Eq. 3 so as to construct the

discrete sequences D^T and D^V. This is, therefore, how we represent the observations of the HMM from the discrete sequences in D^T and D^V, given the original sequences in X^T and X^V, respectively. Then, considering D^T as the set of training sequences, we train every HMM for each class so as to learn our generative models.

3.1.2 Genetic operators

We consider standard *selection*, *crossover* and *mutation* operators from [9]. Specifically, we apply these operators to all genes of each individual (*i.e.* k clusters and N segments). Before applying the mutation operator, however, each of the N segments has again a probability P_s either to *add* if it is empty, or to *delete* if it already exists. The *offspring* also requires to meet several constraints that might be violated once we apply these standard genetic operators. To ensure they are met, we apply a *repair* algorithm to fix the new incorrect segments immediately after applying the crossover and mutation operators. Basically, it consists of a brute force criteria that fixes those incorrect segments either by moving them so as to stay within the length of X^T (even though keeping the segment length proportions when they are correct), or by generating new segments within the range $[n_{min}, n_{max}]$ when they are out of bounds. Moreover, we use Eq. 5 either to increase k_f and hence generate new segments, or to decrease k, the number of clusters:

$$p(k) = \frac{k-k_0}{k_f-k_0} \Rightarrow \begin{cases} \text{if } p(k) \leq 1 & \text{increase } k_f \text{ segments} \\ \text{Otherwise} & \text{decrease } k \text{ clusters.} \end{cases} \tag{5}$$

This procedure ensures, not only that the offspring that pass throughout the next generations are evaluable, but also that we respect the new trends of the genes caused by these genetic operators. The repair function accelerates the convergence of the genetic algorithm.

3.2 Aligned Temporal Clustering

Let X^T_{seg} be the set of k_f sequence segments decoded from an individual I and the whole continuous training sequence X^T. Similarly to the classical $k-means$ algorithm, our method groups the X^T_{seg} examples into k clusters. In our setting, however, each example $x_s \in X^T_{seg}$ is a sequence, so that it is a point in the space and time. Therefore, it is convenient to consider an appropriate measure as DTW so as to treat temporal deformations. Thus, in the expectation step we obtain the costs of aligning each sequence to all the *centroids* (initially k random sequences of X^T_{seg}). Then, we assign each sample to the cluster having the minimum cost of the DTW warping path. In the maximization step, first we update the centroids by means of resizing all sequences that belong to the same cluster *w.r.t.* the median length sequence of that cluster. Then, we calculate the new centroid as the mean of all resized sequences for each cluster. The algorithm converges either when the costs of aligning the current centroids to the ones from the previous iteration are 0, or when it reaches the maximum number of iterations t. Once the algorithm converges, we assign the set of S subgesture sequences as the final centroids.

3.3 Evaluation

The evaluation function computes the mean score of classifying each sequence given the learned model parameters. In training, moreover, we learn the thresholds that provide the

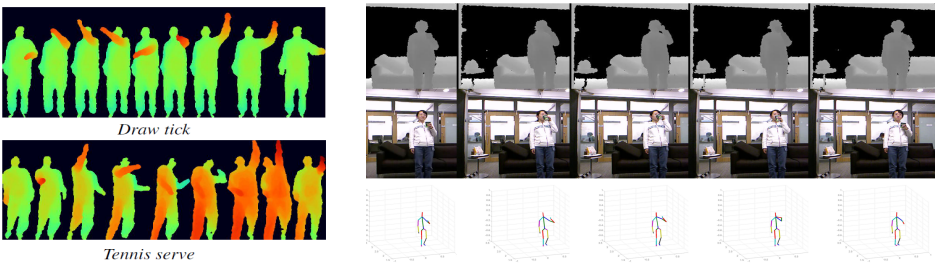

Figure 3: Sample images from the MSRAction3D (left,depth information is available, image taken from [15]) and MSRDaily3D [31] (right, skeleton, RGB-D video available) datasets.

maximum score of classifying each class. Then, we use these thresholds in test time as part of the learned model parameters. We learn and test thresholds as follows:

Dynamic Programming. Once we compute the costs of aligning all validation sequences in X^V to the class-models M, we learn a set of class-thresholds $a = \{\theta^{c_1}, \theta^{c_2}, ..., \theta^{c_g}\}$ as those DTW costs that maximize the score per class, being part of the global set of learned model parameters ω^*. These thresholds are used to compute classification rate of test samples represented in subgestures.

Generative models. Once we learn a HMM per class, we compute the probabilities of generating each discrete sequence in D^V, $P(d^v \in D^V | m_c)$, and learn the class thresholds Θ, included in ω^*. These thresholds are used to compute classification rate of test samples represented in subgestures.

4 Experiments

4.1 Datasets

For the evaluation of the proposed framework we considered two widely used data sets for human action recognition: MSRDaily3D and MSRAction data sets (see Figure 3). We evaluate the performance of our methods and compare its results with state of the art techniques that have used the same data sets.

The MSRDaily3D data set comprises 16 actions associated to daily activities, where there are objects in the background and most actions involve human-object interaction. For comparison with previous work we used this data set under two settings: cross-validation and half-subject split. The former setting allows us to compare the results of our methods with recent work that has used the same descriptor [10, 11, 12, 35]. Under this setting we considered 12 out of the 16 actions and performed 5-fold cross validation (as in [10, 11, 12, 35]). For the other setting we considered the 16 categories and used the sample half-training / half-testing subject split (*e.g.*, as in [15, 31, 32]). In either configuration, video sequences were represented with depth cuboid similarity features (DCSF), the same parameters for the descriptor as in [11, 12, 35] were used.

The MSRAction3D data set comprises 20 actions, recorded by 10 subjects, where subjects are isolated and no objects in the background are present. Together with the MSRDaily data set, this is one of the most used data sets for the assessment of human action recognition

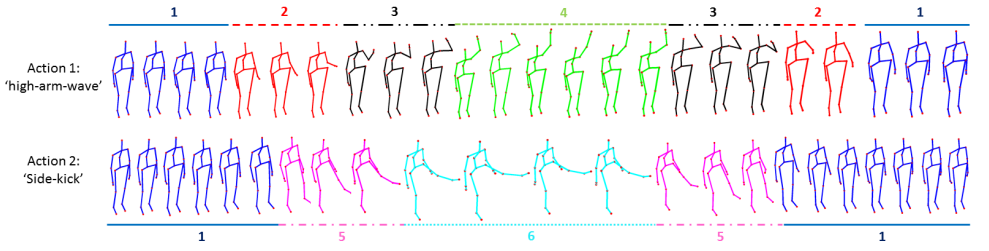

Figure 4: Visual scheme of frame-skeletons grouped into (temporal) subgesture-clusters for the MSRAction3D dataset (best seen in color).

techniques when using the depth/skeleton information. As before, video sequences were represented with a bag of DCSF descriptors. For this data set, the standard half-training (subjects 1,3,5,7,9) / half-testing (rest of subjects) split was adopted (see [22] for a complete analysis of results on this data set).

4.2 Setting and metrics

All of the methods were implemented in MATLAB/C++[2], integrating functionalities from the GA optimtool [9] and PMTK3 libraries. The parameters of our method were fixed as follows: $P_s = 0.2$, $n_{min} = 5$ and $n_{max} = 25$ (as in [37]). We set our population length to $l = 20$, with 2 elitist members that pass throughout the next generations.

The $k-meansDTW$ described in section 3.2 requires both to resize the segments samples of each cluster in X^T_{seg} and to align them *w.r.t.* the k centroids so as to obtain the new clusters. The computational cost of this step is about $\mathcal{O}(t \times k \times n^2)$. Hence, we defined $N = 500$ in our experiments to generate the pairs of start-length segments, providing a trade-off between number of segment and computation requirements. Moreover, we defined $k_0 = 3$ to consider a low value for the minimum number of clusters, so that we allow to set k between a large enough range $[k_0, k_f]$ for the $k-meansDTW$ algorithm. Finally, in the evaluation we use $T = 20$ for the range of thresholds to learn Θ, and compute mean accuracy among all test sequences.

4.3 Results

The DTW baseline consists of using the classical DTW with euclidean distance to classify the test sequences. Thus, instead of learning subgesture models, our baseline models are formed by means of direct resizing all sequence samples of each class *w.r.t.* the median length sequence from that class. On the other hand, in the HMM baseline we split each gesture sequence in 3 parts having the same length to construct the set of sequence segments for learning the subgesture models. The number of clusters k is the half of the total number of resulting segments. To reduce the computational complexity of the HMM baseline we get a reduced number of samples as input to the $k-meansDTW$, so that for each class we choose 10 random gestures rather than considering all the training gesture sequences.

[2] **Library publicly available at https://github.com/vponcelo/Subgesture**

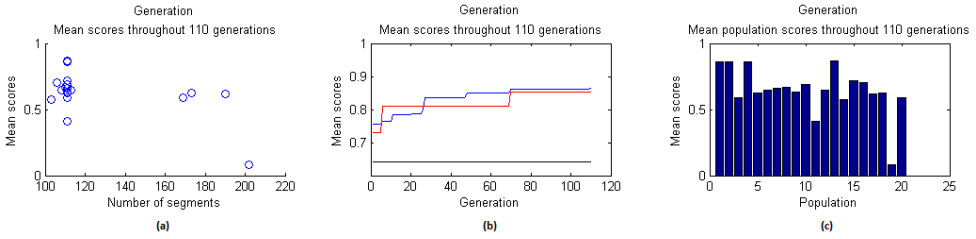

Figure 5: Example of the evolution of the genetic algorithm for the first fold of the MSR-Daily3D dataset. The plot (a) shows the number of segments that belong to each individual and their scores on the last generation. The plot (b) shows the score (accuracy) of the best individuals at each generation so as to see the evolution of the different settings: baseline in validation (black) where there is no evolution, and the evaluation of the models from the best individual of the generation in validation (blue) and test (red). The barplot (c) shows the distribution of scores of each individual of the population on the last generation.

Figure 4 shows an example of representing two sequences of different actions into subgestures on the MSRAction3D dataset, applying the procedure described in Figure 2. The two sequence actions are 'high-arm-wave' and 'side-kick', and the subgestures are those from the last generation that gave the best performance in the evolved DTW version. At the frame level, one can observe that the skeletons that fall into the same cluster are quite similar, though there are some skeletons that are visually similar to those belonging to a different cluster (*e.g.* frame-skeleton 5 in comparison to the frame-skeletons that belong to the cluster 1). At the temporal level, we can observe that the cluster 1, formed by similar segments of different length, is shared among the two different action sequences. The same phenomena happens for the clusters 2, 3 and 5, though these are shared clusters along the same action sequence. This shows the qualitative performance of the $k-means DTW$ algorithm, which provide effective clusters by computing temporal deformations over the input segments of different lengths.

In Table 1, we report the mean results of running our genetic algorithm 5 times both to the half-subject split of the MSRAction3D and MSRDaily3D datasets, and to the 5-fold cross-validation of the MSRDaily3D dataset. In all cases, the evolution of the subgesture models learned with the HMM outperforms the state of the art in these datasets, achieving results above the 91% from the *initial generations*. Specially for the MSRAction3D data set, the improvement of evolving subgesture models with the HMM is the greatest *w.r.t.* the HMM baseline, achieving the best result of the 95%. The evolved DTW version also provides a considerable improvement *w.r.t.* the DTW baselines, outperforming the state of the art on the MSRDaily3D dataset, and achieving comparable performances on the MSRAction3D.

To illustrate the evolution of the genetic algorithm. In the left plot (a) of figure 5 one can see a clear trend of the individuals to go towards the number of segments that give the best performance (111). The middle plot (b) shows that from the starting generations the performance both in validation and test are above the baseline. Their performance improve along the generations and keep very similar on the last generations. From the distribution of scores on the right barplot (c), one can observe that all individuals have positive scores and some of them achieve similar values, showing that the repair algorithm using Eq. 5 forces the individuals to become valid, speeding up convergence.

MSRAction3D-HS		MSRDaily3D-CV		MSRDaily3D-HS	
Method	Accuracy	**Method**	Accuracy	**Method**	Accuracy
[32] (LOP+J.)	88.2%	[10] (SOSVM)	68.3%	[31] (LOP)	42.5%
[35] (DCSF)	89.3%	[12] (SMMED)	73.20%	[21] (DTW)	54%
[24] (HOPC)	91.64%	[35] (DCSF)	83.60%	[32] (MKL)	80.0%
[7] (PBR)	92.3%	[35] (DCSF+Skl.)	88.2	[16] (GP)	85.6%
[30] (MMTW)	92.7%	-	-	[32] (LOP+J.)	85.75%
Dynamic Time Warping					
Baseline	85.76%	Baseline	77.36%	Baseline	70.20%
Evolved	90.89%	Evolved	**89.51%**	Evolved	**88.16%**
Hidden Markov Model					
Baseline	70.85%	Baseline	74.62%	Baseline	69.29%
Evolved	**95%**	Evolved	**91.39%**	Evolved	**92.30%**

Table 1: Recognition results in the MSRAction3D and MSRDaily3D datasets for half-split (HS) and cross-validation (CV), for the latter setting we report the 4 results available in published literature.

5 Conclusion

We introduced a novel approach for learning dynamic gesture primitives for gesture and action recognition. An evolutionary computing framework was presented incorporating two most notable gesture recognition methodologies, namely DTW and HMMs. Experimental results show the competitiveness of our methods, outperforming state of the art results in benchmark data sets after few generations. Our results suggest that the proposed subgesture learning methodology enhances the recognition performance of traditional techniques. Future work includes extending the framework for related tasks (*e.g.* gesture spotting, event detection) and an extensive evaluation under different parameter settings.

Acknowledgments

This work is supported by the projects TIN2013-43478-P and TIN2012-38187-C03-02 from the Spanish Ministry of Economy and Competitiveness. The work of Víctor is supported by the 2013FI-B01037 fellowship. H. J. Escalante was supported by CONACYT under project grant No. CB-2014-241306.

References

[1] J. K. Aggarwal and M. S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys*, 43(3), 2011.

[2] A.F. Bobick and J.W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3): 257–267, 2001.

[3] A.F. Bobick and A.D. Wilson. A state-based approach to the representation and recognition of gestures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(12):1325–1337, 1997.

[4] W. Brendel and S. Todorovic. Learning spatiotemporal graphs of human activities. In *International Conference on Computer Vision*, 2011.

[5] A.A. Chaaraoui and F. Florez-Revuelta. Adaptive human action recognition with an evolving bag of key poses. *IEEE Transactions on Autonomous Mental Development*, 6 (2):139–152, 2014.

[6] A.A. Chaaraoui, J. R. Padilla, P. Climent-Perez, and F. Florez-Revuelta. Evolutionary joint selection to improve human action recognition with rgb devices. *Expert systems with applications*, 41:786–794, 2014.

[7] A. Eweiwi, M. S. Cheema, C. Bauckhage, and J. Gall. Efficient pose-based action recognition. In *Proceedings of Assian Conference on Computer Vision, LNCS*, volume 9007, pages 428–443, 2015.

[8] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[9] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989.

[10] M. Hoai and F. De la Torre. Max-margin early event detectors. In *Computer Vision and Pattern Recognition*, 2012.

[11] M. Hoai, Z. Lan, , and F. De la Torre. Joint segmentation and classification of human actions in video. In *Computer Vision and Pattern Recognition*, 2011.

[12] D. Huang, S. Yao, Y. Wang, and F. De La Torre. Sequential max-margin event detectors. In *European Conference on Computer Vision*, 2014.

[13] D. Kim, J. Song, and D. Kim. Simultaneous gesture segmentation and recognition based on forward spotting accumulative hmms. *Pattern Recognition*, 40(11):3012–3026, 2007.

[14] K. Li, J. Hu, and Y. Fu. Modeling complex temporal composition of actionlets for activity prediction. In *European Conference on Computer Vision*, volume 7572, pages 286–299, 2012.

[15] W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3D points. In *Computer Vision and Pattern Recognition Workshops*, pages 9–14, 2010.

[16] L. Liu and L. Shao. Learning discriminative representations from rgb-d data. In *International Joint Conference on Artificial Intelligence*, 2013.

[17] L. Liu, L. Shao, and P. Rockett. Boosted key-frame selection and correlated pyramidal motion-feature representation for human action recognition. *Pattern recognition*, 46 (7):1810–1818, 2013.

[18] F. Lv and R. Nevatia. Recognition and segmentation of 3-d human action using hmm and multi-class adaboost. In *European Conference on Computer Vision*, pages 359–372, 2006.

[19] M. R. Malgireddy, J. J. Corso, S. Setlur, V. Govindaraju, and D. Mandalapu. A framework for hand gesture recognition and spotting using sub-gesture modeling. In *International Conference on Pattern Recognition*, 2010.

[20] M. R. Malgireddy, I. Nwogu, S. Ghosh, and V. Govindaraju. A shared parameter model for gesture and sub-gesture analysis. In *Combinatorial Image Analysis*, volume 6636, pages 483–493, 2011.

[21] M. Muller and T. Roder. Motion templates for automatic classification and retrieval of motion capture data. In *ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 137–146, 2006.

[22] J. R. Padilla-López, A. A. Chaaraoui, and F. Flórez-Revuelta. A discussion on the validation tests employed to compare human action recognition methods using the MSR action3d dataset. *CoRR*, abs/1407.7390, 2014. URL http://arxiv.org/abs/1407.7390.

[23] V. Ponce, M. Gorga, X. Baro, and S. Escalera. Human behavior analysis from video data using bag-of-gestures. In *International Joint Conference on Artificial Intelligence*, 2011.

[24] H. Rahmani, A. Mahmood, D. Huynh, and A. Mian. HOPC: Histogram of oriented principal components of 3d pointclouds for action recognition. In *European Conference on Computer Vision, LNCS*, volume 8690, pages 742–757, 2014.

[25] M. Raptis and L. Sigal. Poselet key-framing: A model for human activity recognition. In *Computer Vision and Pattern Recognition*, pages 2650–2657, 2013.

[26] M. Raptis, I. Kokkinos, and S. Soatto. Discovering discriminative action parts from mid-level video representations. In *Computer Vision and Pattern Recognition*, 2012.

[27] 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 a single depth image. In *Computer Vision and Pattern Recognition*, 2011.

[28] T.E. Starner and A. Pentland. Visual recognition of american sign language using hidden markov models. In *Proc. Int. Workshop Automatic Face and Gesture Recognition*, 1995.

[29] P. K. Wagner, S. M. Peres, R. C. Barros Madeo, C. A. M. Lima, and F. A. Freitas. Gesture unit segmentation using spatial-temporal information and machine learning. In *Florida Artificial Intelligence Research Society Conference*, 2014.

[30] J. Wang and Y. Wu. Learning maximum margin temporal warping for action recognition. In *Proceedings of International Conference on Computer Vision*, pages 2688–2695, 2013.

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

[32] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Learning actionlet ensemble for 3d human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(5): 914–927, 2014.

[33] L. Wang, Y. Qiao, , and X. Tang. Video action detection with relational dynamic-poselets. In *European Conference on Computer Vision*, 2014.

[34] A.D. Wilson and A.F. Bobick. Parametric hidden markov models for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(9):884–900, 1999.

[35] L. Xia and J. K. Aggarwal. Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera. In *Computer Vision and Pattern Recognition*, pages 2834–2841, 2013.

[36] Z. Zhao and A. M. Elgammal. Information theoretic key frame selection for action recognition. In *British Machine Vision Conference*, 2008.

[37] F. Zhou, F. De la Torre Frade, and Jessica K Hodgins . Hierarchical aligned cluster analysis for temporal clustering of human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3):582–596, 2013.