Skeleton-Based Hand Gesture Recognition by Learning SPD Matrices with Neural Networks
Xuan Son Nguyen, Luc Brun, Olivier Lezoray, Sébastien Bougleux

To cite this version:
Xuan Son Nguyen, Luc Brun, Olivier Lezoray, Sébastien Bougleux. Skeleton-Based Hand Gesture Recognition by Learning SPD Matrices with Neural Networks. 14th IEEE International Conference on Automatic Face and Gesture Recognition, May 2019, Lille, France. hal-02133684

HAL Id: hal-02133684
https://hal.archives-ouvertes.fr/hal-02133684
Submitted on 19 May 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Skeleton-Based Hand Gesture Recognition by Learning SPD Matrices with Neural Networks

Xuan Son Nguyen¹, Luc Brun¹, Olivier Lezoray² and Sébastien Bougleux²
¹ Normandie Univ, ENSICAEN, UNICAEN, CNRS, GREYC, 14000 Caen, France
² Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France

Abstract—In this paper, we propose a new hand gesture recognition method based on skeletal data by learning SPD matrices with neural networks. We model the hand skeleton as a graph and introduce a neural network for SPD matrix learning, taking as input the 3D coordinates of hand joints. The proposed network is based on two newly designed layers that transform a set of SPD matrices into a SPD matrix. For gesture recognition, we train a linear SVM classifier using features extracted from our network. Experimental results on a challenging dataset (Dynamic Hand Gesture dataset from the SHREC 2017 3D Shape Retrieval Contest) show that the proposed method outperforms state-of-the-art methods.

I. INTRODUCTION

Hand gesture recognition is one of the main tasks in video understanding. This task has received increasing attention from the computer vision community, as it has many potential applications in human-robot interaction, sign language interpretation, and smart home control interface. Existing vision-based approaches for hand gesture recognition use generally three data modalities, i.e. RGB and depth (RGB-D) images, and skeletal data. Most of these approaches are mainly based on RGB-D images since they can be directly acquired from vision sensors, while skeletal data must be estimated by a tracking algorithm and is thus more difficult to obtain and less reliable than RGB-D images.

The introduction of depth cameras like Intel Realsense and Microsoft Kinect enables effective hand gesture recognition using skeletal data [27], [28]. Compared to RGB-D images, skeletal data offers several advantages. First, view-invariant gesture recognition can be achieved using skeletal data. Second, features extracted from skeletal data are generally low-dimensional and fast to compute compared to those extracted from RGB-D images. Figure 1(a) shows the hand joint positions estimated by an Intel Realsense camera. Motivated by the recent success of graph convolutional networks for action recognition [17], [34], we propose to model the hand skeleton as a graph for hand gesture recognition. Figure 1(b) shows the graph at one frame corresponding to the hand skeleton in Fig. 1(a). To obtain gesture representations, we rely on first- and second-order statistics. These statistics are combined using Gaussian distributions [18], [33], forming Symmetric Positive Definite (SPD) matrices. Since SPD matrices are known to lie on a Riemannian manifold, we develop a neural network for SPD matrix learning [5], [11].

This material is based upon work supported by the European Union and the Region Normandie under the project IGIL.

[15], [19], [33]. To this end, we propose two new types of layers that transform a set of SPD matrices into a SPD matrix. Despite the simplicity of our network, it outperforms state-of-the-art methods on the challenging Dynamic Hand Gesture (DHG) dataset [28] from the SHREC 2017 3D Shape Retrieval Contest.

II. RELATED WORKS

A. Skeleton-Based Hand Gesture and Action Recognition

Ioenescu et al. [14] extracted the hand region skeleton and a dynamic signature of the gesture was then computed as the superposition of all the skeletons of the hand region. Wang and Chan [31] segmented the hand region and used a superpixel representation to capture the hand shape and texture information. De Smedt et al. [27] proposed two kinds of features based on hand shapes and movements. The shape-based features were computed using Fisher vector, and the movement-based features were computed from histogram of hand directions and histogram of wrist rotations. Nunez et al. [21] trained a combination of a Convolutional Neural Network (CNN) [16] and a Long Short-Term Memory (LSTM) [10] using two stages. In the first stage, the CNN was connected to a fully-connected multi-layer perceptron of two hidden layers. In the second stage, the output of the pre-trained CNN was connected to the LSTM. Devineau et al. [4] separated a gesture sequence into 1D sequences and fed each of them to a multi-channel CNN consisting of three parallel branches. The first branch was a residual network, while the two other branches were designed for feature extraction.

For skeleton-based action recognition, two categories of features have also been investigated, e.g. hand-crafted features and deep-learned features. In hand-crafted feature-based approaches, the 3D joint positions of the human body were used to compute skeletal quad [8], points in a Lie group [29], Actionlet Ensemble [32], EigenJoints [35]. In deep learning-based-approaches, various architectures based on CNN, Recurrent Neural Network [6], LSTM [20], [24] were introduced to learn action representations. Very recently, geometric deep learning which extends classical operations like convolution to non-Euclidean domains, e.g. manifolds and graphs, has also been applied to skeleton-based action recognition. Manifold learning approaches operated on different manifolds, e.g. SPD manifolds [11], Lie groups [12], and Grassmann manifolds [13]. Graph learning approaches [25], [34] were based on graph convolution filters
and Deep Residual Networks [9] which have shown very promising results on large scale datasets.

B. Deep Learning on Graphs

Deep learning on graphs, as mentioned previously, has received increasing attention in recent years. While various approaches have been proposed and proven effective in different tasks [2], [26], [30], very few approaches have been developed so far for action and hand gesture recognition. Yan et al. [34] introduced a Spatial Temporal Graph CNN operating on a graph constructed from body joints and physical connections between them. Their network was designed and trained based on the mechanism of Deep Residual Networks [9]. Shi et al. [25] improved [34] by automatically learning the graph structure at each layer, while Zhang et al. [36] improved [34] by defining an equivalent convolution operator on graph edges. Similarly to [34], Li et al. [17] constructed a graph to represent the human body skeleton. A polynomial approximation of the Graph Fourier Transform was then used to design multi-scale convolutional filters, which were then combined with autoregressive moving average to build a spatio-temporal graphical convolutional model.

C. Deep Learning of SPD Matrices

Ionescu et al. [15] introduced the theory and practical techniques for matrix backpropagation in the deep learning framework. They also proposed Deep2P, a network for region classification, and an algorithm for backpropagation of logarithm of SPD matrices. Based on [15], several methods for SPD matrix learning have been proposed. Dong et al. [5] introduced a 2D fully connected layer and a symmetrically clean layer for dimensionality reduction and non-linear transformation of SPD matrices. Huang and Gool [11] mapped a high dimensional SPD matrix into a lower dimensional one via Bimap, a layer having a similar form as the 2D fully connected layer proposed in [5]. Kernel matrices were also used to design non-linear activation functions [7]. Li et al. [19] and Wang et al. [33] combined parametric probability distributions modeling into CNNs, which requires backpropagation of a square-rooted SPD matrix.

III. PROPOSED NETWORK

A. Overview

Given a hand skeleton sequence, we construct a graph for each frame as illustrated in Fig. 1(b). The features at a node provide a 3-dimensional vector formed from the x, y, and z coordinates of the corresponding hand joint. We design a simple 2D convolutional layer which is applied to the input data to take into account the correlation between neighboring nodes. A sub-sequence of a specific finger is extracted from the sequence and processed by a branch of the network. Two layers referred to as GaussAgg and SPDTempAgg are then used to learn joint features for each branch. The outputs of all branches of the network are combined using a layer referred to as SPDSpatAgg resulting in a SPD matrix. Since the obtained SPD matrix lies on a Riemannian manifold, it is mapped to Euclidean spaces for classification. The proposed network architecture is illustrated in Fig. 2.

In the next two sections, we denote by $n_k$ the number of input SPD matrices of the $k^{th}$ layer, by $d_{k-1}$ and $d_k$ the dimensions of input and output SPD matrices of the $k^{th}$ layer, respectively, by $X_{k-1}^i$, $i = 1, \ldots, n_k$, the input SPD matrices of the $k^{th}$ layer, and by $X_k$ the output SPD matrix of the $k^{th}$ layer.

B. SPD Spatial Aggregation

The SPDSpatAgg layer (step (k) of Fig. 2) computes $X_k$ by a mapping $f_{sa}$ as:

$$X_k = f_{sa}^{(k)}((X_{k-1}^1, \ldots, X_{k-1}^{n_k}); W_{1}^{k}, \ldots, W_{n_k}^{k})$$

$$= \sum_{i=1}^{n_k} W_{i}^{k} X_{k-1}^i (W_{i}^{k})^{T},$$

(1)

where $W_{i}^{k} \in \mathbb{R}^{d_k \times d_{k-1}}$ are the transformation matrices.

To guarantee that $X_k$ is SPD, one can use the same assumption as [11], i.e. each transformation matrix $W_i^k$ is a full row rank matrix. In this case, each term in the right-hand side of Eq. (1) is SPD, so their sum is also SPD. Then optimal solutions of the transformation matrices are achieved by additionally assuming that each transformation matrix $W_i^k$ resides on a compact Stiefel manifold $St(d_k, d_{k-1})$\footnote{A compact Stiefel manifold $St(d_k, d_{k-1})$ is the set of $d_k$-dimensional orthonormal matrices of $\mathbb{R}^{d_k \times d_{k-1}}$.}. Note that this assumption imposes a constraint on the dimension of $X_k$, i.e. $d_k \leq d_{k-1}$. The backpropagation procedure of this layer is based on the optimization framework of [11].

C. SPD Temporal Aggregation

The SPDTempAgg layer is decomposed into three sub-layers as illustrated at steps (g), (h) and (i) of Fig. 2. The first sub-layer computes the matrix logarithm $Y_{k-1}^i$ from $X_{k-1}^i$, for $i = 1, \ldots, n_k$, as:

$$Y_{k-1}^i = \log(X_{k-1}^i) = U \log(V) U^{T},$$

(2)

where $UVU^{T}$ is the eigen-decomposition of $X_{k-1}^i$. This layer can be implemented similarly to the LogEig layer [11].
A hand skeleton sequence

Fig. 2: (Best viewed in color) The proposed network. (a) A hand skeleton sequence. (b) The convolutional layer for the graph of hand joints. (c) The output of this layer. (d) Construction of different sub-sequences, each of them is associated with a specific finger. Two sub-sequences associated with the thumb and the index are shown here. Each sub-sequence is processed by a branch of the network (from (e) to (j)). (k) The SPD matrices at step (j) are transformed into a SPD matrix.

The second sub-layer, referred to as HalfVec layer, vectorizes the upper triangle of \( Y_i \) for \( i = 1, \ldots, n_e \) as:

\[
z^t = [ Y_{i-1}^t(1, 1), \sqrt{2} Y_{i-1}^t(1, 2), \ldots, \sqrt{2} Y_{i-1}^t(1, d_{k-1}), Y_{i-1}^t(2, 2), \sqrt{2} Y_{i-1}^t(2, 3), \ldots, Y_{i-1}^t(d_{k-1}, d_{k-1})]^T
\]

(3)

where \( Y_i(u, v) \) denotes the element at the \( u \)th row and \( v \)th column of \( Y_{i-1} \).

The third sub-layer, referred to as GaussAgg layer, aggregates the \( z^t \), for \( i = 1, \ldots, n_e \), based on their Gaussian distributions [33] to obtain the output of SPDTempAgg:

\[
X_k = f_{\text{Conv}}^k((X_1^{t_k}, \ldots, X_{n_k}^{t_k})) = [\Sigma + \mu \mu^T \mu \mu^T 1]
\]

(4)

where \( \mu = \frac{1}{n_e} \sum_{i=1}^{n_e} z^t \) and \( \Sigma = \frac{1}{n_e} \sum_{i=1}^{n_e} (z^t - \mu)(z^t - \mu)^T \).

The backpropagation procedure of this layer is based on the framework of [15].

D. The Network in Detail

Let \( n_j, n_F \) be the number of hand joints and the length of the gesture sequence, respectively. The input of our network (Fig. 2) is a sequence of hand poses denoted by \( p_{0,i}^t \in \mathbb{R}^3, i = 1, \ldots, n_j, t = 1, \ldots, n_F \), where \( p_{0,i}^t \) corresponds to the 3D coordinates of joint \( i \) at frame \( t \). Let \( d_1 \) be the number of channels of output features in the first layer. Denote by \( p_{1,i}^t \in \mathbb{R}^{d_1}, i = 3, \ldots, n_j, t = 1, \ldots, n_F \) the output of the first layer, by \( p_{1,i}^t \in \mathbb{R}^{d_1}, i = 1, \ldots, d_1 \), i.e. \( p_{1,i}^t = [p_{1,i,1}, \ldots, p_{1,i,d_1}]^T \). At the first layer, the value for the \( c \)th channel of the feature vector on node \( i \) is given by:

\[
p_{1,i,c}^t = \sum_{j \in \mathcal{N}_i} w_{ij}(j,i,c) \cdot p_{0,j}^t.
\]

(5)

where \( \mathcal{N}_i \) is the set of neighbors of node \( i \), \( w_{ij}(j,i,c) \in \mathbb{R}^3 \) is the filter weight vector for the \( c \)th channel, and \( l(j,i) \) is defined as:

\[
l(j,i) = \begin{cases} 1 & \text{if } j - i = 0 \\ 2 & \text{if } j - i = 1 \\ 3 & \text{if } j - i = -1 \end{cases}
\]

(6)

At step (d) of Fig. 2, we partition the set of hand joints into 5 subsets of joints, each of them contains all joints of a finger. For example, the 5 subsets of joints corresponding to the hand skeleton illustrated in Fig. 1 are given by: \( J_1 = \{3, 4, 5, 6\}, J_2 = \{7, 8, 9, 10\}, J_3 = \{11, 12, 13, 14\}, J_4 = \{15, 16, 17, 18\}, J_5 = \{19, 20, 21, 22\} \). Then 5 sequences associated with those 5 subsets are constructed from the output of the first layer. For each \( J_s \), with \( s = 1, \ldots, 5 \), the GaussAgg layer produces SPD matrices given by:

\[
X_s^{*, t} = \frac{\Sigma^{*, t} + \mu^{*, t}(\mu^{*, t})^T}{(\mu^{*, t})^T} \mu^{*, t}
\]

(7)

where \( \mu^{*, t} = \frac{1}{n_s} \sum_{i \in J_s} p_{1,i}^t \) and \( \Sigma^{*, t} = \frac{1}{n_s} \sum_{i \in J_s} (p_{1,i}^t - \mu^{*, t})(p_{1,i}^t - \mu^{*, t})^T \).

The next step uses the ReEig layer [11] to perform nonlinear transformations of SPD matrices given by:

\[
X_s^{*, t} = f_r^{(3)}(X_s^{*, t}) = U \max(cI, V)U^T
\]

(8)
where $X_{s,t}^{q,t}$, for $s = 1, \ldots, 5$, $t = 1, \ldots, n_F$ are the output SPD matrices, $UU^T$ is the eigen-decomposition of $X_{s,t}^{q,t}$. $\epsilon$ is a rectification threshold, $I$ is an identity matrix, and $\max(I, V)$ is a diagonal matrix defined as:

$$\max(I, V))(i, i) = \begin{cases} V(i, i) & \text{if } V(i, i) > \epsilon \\ \epsilon & \text{if } V(i, i) \leq \epsilon \end{cases}$$

Next, a multi-level representation is constructed for the sequence associated with $J_s$ for $s = 1, \ldots, 5$. The sequence is evenly subdivided into $i$ sub-sequences at the $i$th level, $i = 1, \ldots, n_T$, with $n_T$ the total number of levels. This creates $n_Q = n_T(n_T + 1)/2$ sub-sequences for each $J_s$. Then the SPDTempAgg layer (Sec. III-C) is applied for each sub-sequence to generate SPD matrices $X_{i}^{q,t}$, using Eq. (4) for $s = 1, \ldots, 5$ and $q = 1, \ldots, n_Q$:

$$X_{i}^{s,q,t} = f_{i,a}^{(4)}(X_{3}^{s,t}, \ldots, X_{3}^{s,t})$$

where $t_b$ and $t_d$ are the beginning and ending frame indices of the $q^{th}$ sub-sequence ($X_{s,t}^{q,t}$) for $t = t_b, \ldots, t_d$.

Finally, the SPD matrices $X_{i}^{s,q,t}$ are transformed into a SPD matrix by the SPDSpatAgg layer presented in Sec. III-B. The new matrix is transformed to its logarithm via a LogEig layer before being fed into a fully connected (FC) layer, followed by a softmax layer.

**E. Gesture Representation**

The final representation of a gesture sequence is generated by extracting features from the output of the SPDSpatAgg layer. The obtained matrix is then transformed to its matrix logarithm and finally vectorized. Let $B \in \mathbb{R}^{d}$ be the output matrix of the SPDSpatAgg layer, then the final representation of the gesture sequence is given as $v = [b_{1,1}, \sqrt{2}b_{1,2}, \sqrt{2}b_{1,3}, \ldots, \sqrt{2}b_{1,d+1}, b_{2,2}, \sqrt{2}b_{2,3}, \ldots, b_{d,d}]^T$, where $b_{i,j}, i = 1, \ldots, d$ are the diagonal entries of $\log(B)$ and $b_{i,j}, i < j, i, j = 1, \ldots, d$ are the off-diagonal entries at the $i^{th}$ row and $j^{th}$ column of $\log(B)$.

**IV. EXPERIMENTS**

We conduct experiments using the DHG dataset [28]. The dataset contains 14 gestures performed in two ways: using one finger, and using the whole hand. Each gesture is executed several times by 28 participants. The gestures are subdivided into categories of fine and coarse: Grab (fine), Tap (coarse), Expand (fine), Pinch (fine), Rotation Clockwise (Rot-CW, fine), Rotation Counterclockwise (Rot-CCW, fine), Swipe Right (Swipe-R, coarse), Swipe Left (Swipe-L, coarse), Swipe Up (Swipe-U, coarse), Swipe Down (Swipe-D, coarse), Swipe X (Swipe-X, coarse), Swipe V (Swipe-V, coarse), Swipe + (Swipe+, coarse), Shake (coarse). The dataset provides the 3D coordinates of 22 hand joints as illustrated in Fig. 1(a).

**Experimental settings:** The dimension of an output feature vector of the first layer is set to 9. The dimensions of the transformation matrices of the SPDSpatAgg layer are set to 56 × 56. The number of levels used for the SPDTempAgg layer is set to 3. The batch size and the learning rate are set to 30 and 0.01, respectively. The rectification threshold $\epsilon$ used for the ReEig layer is set to 0.0001 [12]. For gesture classification, we use the LIBLINEAR library [9] with L2-regularized L2-loss (dual) to train the classifier and use the default parameter settings in LIBLINEAR (C is set to 1, the tolerance of termination criterion is set to 0.1 and no bias term is added). Following [28], we split the dataset into 1960 train sequences (70% of the dataset) and 840 test sequences (30% test sequences). All sequences of the dataset are normalized to have the same length as the longest sequence, which is 171. Since a gesture is performed in two ways, the number of gesture classes can be 14 or 28 [28]. We consider both cases in our experiments. The network trained at epoch 20 is used to extract features (Sec. III-E).

**Results:** Figure 3 shows the confusion matrix of our method on the DHG dataset. The recognition errors concentrate on highly similar actions, e.g. Grab to Pinch. Our method achieves 100% accuracy in recognizing actions Swipe-, Swipe-V, and more than 90% accuracy in recognizing actions Grab, Rot-CW, Rot-CCW, Swipe-R, Swipe-L, Swipe-X, Shake. The comparison of our method and state-of-the-art methods on the DHG dataset is given in Tab. I. Our method outperforms the second best method [4] by 1.1 and 1.96 percent points for experiments with 14 and 28 gestures, respectively. It worth mentioning that our network is much simpler than [4].

**V. CONCLUSIONS**

We have presented a new hand gesture recognition method based on a neural network for SPD matrix learning. Our
method learns a discriminative SPD matrix encoding the first-order and second-order statistics of local features. We have provided the experimental evaluation on a benchmark dataset showing the effectiveness of the proposed method.

REFERENCES


