A neural network based on SPD manifold learning for skeleton-based hand gesture recognition
Xuan Son Nguyen, Luc Brun, Olivier Lezoray, Sébastien Bougleux

To cite this version:
Xuan Son Nguyen, Luc Brun, Olivier Lezoray, Sébastien Bougleux. A neural network based on SPD manifold learning for skeleton-based hand gesture recognition. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019, Long Beach, United States. hal-02456437

HAL Id: hal-02456437
https://hal-normandie-univ.archives-ouvertes.fr/hal-02456437
Submitted on 27 Jan 2020
A neural network based on SPD manifold learning for skeleton-based hand gesture recognition

Xuan Son Nguyen, Luc Brun, Olivier Lézoray, Sébastien Bougleux
Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France
(xsnguyen,luc.brun)@ensicaen.fr, {olivier.lezoray,sebastien.bougleux}@unicaen.fr

Abstract

This paper proposes a new neural network based on SPD manifold learning for skeleton-based hand gesture recognition. Given the stream of hand’s joint positions, our approach combines two aggregation processes on respectively spatial and temporal domains. The pipeline of our network architecture consists in three main stages. The first stage is based on a convolutional layer to increase the discriminative power of learned features. The second stage relies on different architectures for spatial and temporal Gaussian aggregation of joint features. The third stage learns a final SPD matrix from skeletal data. A new type of layer is proposed for the third stage, based on a variant of stochastic gradient descent on Stiefel manifolds. The proposed network is validated on two challenging datasets and shows state-of-the-art accuracies on both datasets.

1. Introduction

Hand gesture recognition is an important research topic with applications in many fields, e.g., assisted living, human-robot interaction or sign language interpretation. A large family of hand gesture recognition methods is based on low-level features extracted from images, e.g., spatio-temporal interest points. However, with the introduction of affordable depth sensing cameras, e.g., Intel Realsense or Microsoft Kinect, and the availability of highly accurate joint tracking algorithms, skeletal data can be obtained effectively with good precision. Skeletal data provides a rich and high level description of the hand. This has led to an extensive development of approaches for skeleton-based hand gesture recognition in recent years.

Early works on the recognition of hand gestures or human actions from skeletal data are based on a modeling of the skeleton’s movement as time series [49, 52]. The recognition step is thus based on the comparison of sequences of features describing skeleton’s movements using, e.g., Dynamic Time Warping [49] or Fourier Temporal Pyramid [52].

Such approaches ignore the high correlations existing between the movement of two adjacent hand joints (e.g., two joints of a same finger) within a hand gesture. Taking into account this information is a crucial step for hand gesture recognition and requires the definition and the appropriate processing of hand joints’ neighborhoods. Recently, graph convolutional networks for action recognition [27, 56] have shown excellent performance by taking into account physical connections of body joints defined by the underlying structure of body skeleton. While the use of physical connections of skeleton joints are important for capturing discriminating cues in hand gesture and action recognition, the identification of other connections induced by the performed gestures and actions are also useful and can greatly improve recognition accuracy [45].

Motivated by this observation, we model in this work the hand skeleton as a 2D grid where connections, different from the classical physical connections of hand joints, are added to better capture patterns defined by hand joints’ movements. Figure 1(a) shows the hand joint positions estimated by an Intel Realsense camera. Since the hand skeleton has an irregular geometric structure that differs from grid-shaped structures, the 2D grid is constructed from the hand skeleton by removing some hand joints and adding connections between neighboring joints. Figure 1(b) shows a 2D grid corresponding to the hand skeleton in Fig. 1(a). This 2D grid integrates adjacency relationships between hand joints that often have correlated movements. Moreover, this modeling allows us to use a classical convolutional layer instead of a graph convolutional operator on an arbitrary geometric graph [56].

Our approach relies on SPD matrices to aggregate features resulting from the convolutional layer. The SPD matrices considered in this work combine mean and covariance information which have been shown effective in various vision tasks [15, 21, 22]. Since SPD matrices are known to lie on a Riemannian manifold, specific layers for deep neural networks of SPD matrices should be designed [19, 60]. Beside good performances on action recognition tasks, these
networks do not put a focus on spatial and temporal relationships of skeleton joints. This motivates us to design a neural network model for learning a SPD matrix-based gesture representation from skeletal data with a special attention on those relationships. In our work, the encoding of spatial and temporal relationships of hand joints is performed using different network architectures. This allows to capture relevant statistics for individual hand joints as well as groups of hand joints whose movement is highly correlated with that of other joints in the group. The experimental evaluation shows that our method significantly improves the state-of-the-art methods on two standard datasets.

2. Related Works

This section presents representative works for skeleton-based hand gesture recognition (Sec. 2.1) and deep neural networks for SPD manifold learning (Sec. 2.2).

2.1. Skeleton-Based Gesture Recognition

Most of approaches can be categorized as hand-crafted feature-based approaches or deep learning approaches. Hand-crafted feature-based approaches describe relationships of hand and body joints in different forms to represent gestures and actions. The simplest proposed relationships are relative positions between pairs of joints [35, 46, 57]. More complex relationships were also exploited, e.g., skeletal quad [8] or 3D geometric relationships of body parts in a Lie group [49]. Temporal relationships have also been taken into account and proven effective [50]. While all joints are involved in the performed gestures and actions, only a subset of key joints is important for the recognition task. These are called informative joints and they can be automatically identified using information theory [37]. This allows to avoid considering non-informative joints that often bring noise and degrade performance.

Motivated by the success of deep neural networks in various vision tasks [13, 17, 26], deep learning approaches for action and gesture recognition have been extensively studied in recent years. To capture spatial and temporal relationships of hand and body joints, they rely mainly on Convolutional Neural Network (CNN) [4, 25, 32, 33, 36, 53], Recurrent Neural Network (RNN) [6, 51] and Long Short-Term Memory (LSTM) [31, 36, 44]. While hand-crafted feature-based approaches have used informative joints to improve recognition accuracy, deep learning approaches were based on attention mechanism to selectively focus on relevant parts of skeletal data [30, 55]. Recently, deep learning on manifolds and graphs has increasingly attracted attention. Approaches following this line of research have also been successfully applied to skeleton-based action recognition [19, 20, 23, 27, 56]. By extending classical operations like convolutions to manifolds and graphs while respecting the underlying geometric structure of data, they have demonstrated superior performance over other approaches.

2.2. Deep Learning of SPD Matrices

In recent years the deep learning community has shifted its focus towards developing approaches that deal with data in a non-Euclidean domain, e.g., Lie groups [20], SPD manifolds [19] or Grassmann manifolds [23]. Among them, those that deal with SPD manifolds have received particular attention. This comes from the popular applications of SPD matrices in many vision problems [1, 14, 16, 58].

Deep neural networks for SPD matrix learning aim at projecting a high-dimensional SPD matrix into a more discriminative low-dimensional one. Differently from classical CNNs, their layers are designed so that they preserve the geometric structure of input SPD matrices, i.e., their output are also SPD matrices. In [5], a 2D fully connected layer was proposed for the projection, while in [19] it was achieved by a Bimap layer. Inspired by ReLU layers in CNNs, different types of layers that perform non-linear transformations of SPD matrices were also introduced [5, 7, 19]. To classify the final SPD matrix, a layer is generally required to map it to an Euclidean space. Most of approaches rely on the two widely used operations in many machine learning models, i.e., singular value decomposition (SVD) and eigen value decomposition (EIG) for constructing this type of layers [19, 29, 54]. As gradients involved in SVD and EIG cannot be computed by traditional backpropagation, they exploit the chain rule established by Ionescu et al. [24] for backpropagation of matrix functions in deep learning.

3. The Proposed Approach

In this section, we present our network model referred to as Spatial-Temporal and Temporal-Spatial Hand Gesture Recognition Network (ST-TS-HGR-NET). An overview of our network is given in Section 3.1. The different components of our network are explained in Sections 3.2, 3.3, 3.4, and 3.5. In Section 3.6, we show how our network is trained.
for gesture recognition. Finally, Section 3.7 points out the relations of our approach with previous approaches.

3.1. Overview of The Proposed Network

Our network illustrated in Fig. 2 is made up of three components. The first component, referred to as CONV, is a convolutional layer applied on the 2D grid encoding the hand skeletal data (Fig. 1). Filter weights are shared over all frames of the sequence.

The second component is based on the Gaussian embedding method of [34] and is used to capture first- and second-order statistics. This component is composed of two different architectures for feature aggregation referred to as Spatial-Temporal Gaussian Aggregation Sub-Network (ST-GA-NET) and Temporal-Spatial Gaussian Aggregation Sub-Network (TS-GA-NET).

The third component, referred to as SPD Matrix Learning and Classification Sub-Network (SPDC-NET), learns a SPD matrix from a set of SPD matrices and maps the resulting SPD matrix, which lies on a Riemannian manifold, to an Euclidean space for classification.

In the following, we explain in detail each component of our network. The backpropagation procedures of our network’s layers are given in the supplementary material.

3.2. Convolutional Layer

The convolutional layer (Fig. 3) used in the first place of our network allows to combine joints with correlated variations (Section 1). Let \( N_J \) and \( N_F \) be respectively the number of hand joints and the length of the skeleton sequence. Let us denote by \( p_{0,j}^i \in \mathbb{R}^3 \), \( i = 1, \ldots, N_J \), \( t = 1, \ldots, N_F \), the 3D coordinates of hand joint \( i \) at frame \( t \) (Section 1). The grid has three channels corresponding to the x, y, and z coordinates of hand joints. Fig. 1(b) shows the 2D grid corresponding to the hand skeleton in Fig. 1(a), where each node of \( N_i \) has at most 9 neighbors including itself. Let \( d_{out}^c \) be the output dimension of the convolutional layer. Let us denote by \( p_t^i \in \mathbb{R}^{d_{out}} \), \( i = 3, \ldots, N_J \), \( t = 1, \ldots, N_F \), the output of the convolutional layer. The output feature vector at node \( i \) is computed as:

\[
p_t^i = \sum_{j \in N_i} W_{l(j,i)} p_{0,j}^t,
\]

\( W_{l(j,i)} \) is the filter weight matrix, and \( l(j,i) \) is defined as:

\[
\begin{array}{ccccccccc}
 j - 1 & 0 & 1 & -3 & -4 & -5 & -1 & 3 \\
 l(j,i) & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9
\end{array}
\]

3.3. Spatial-Temporal Gaussian Aggregation Sub-Network

To capture the temporal ordering of a skeleton sequence, a number of sub-sequences are constructed and then fed to different branches of ST-GA-NET (see Fig. 4). A branch of ST-GA-NET is designed to aggregate features for a sub-sequence of a specific finger. In this paper, we construct six sub-sequences for each skeleton sequence. The first sub-sequence is the original sequence. The next two sub-sequences are obtained by dividing the sequence into two sub-sequences of equal length. The last three sub-sequences are obtained by dividing the sequence into three sub-sequences of equal length. This results in 30 branches for ST-GA-NET (6 sub-sequences \( \times 5 \) fingers).

To aggregate features in a branch associated with sub-sequence \( s \) and finger \( f \), \( s = 1, \ldots, 6 \), \( f = 1, \ldots, 5 \), each frame of sub-sequence \( s \) is processed through 4 layers. Let \( J_f \) be the set of hand joints belonging to finger \( f \), \( t_b^f, t_e^f \) be the beginning and ending frames of sub-sequence \( s \), \( t \) be a given frame of sub-sequence \( s \), \( \{p_{0,j}^i \mid j \in J_f, i = t_b^f, \ldots, t_e^f \} \) be the subset of output feature vectors of the convolutional layer that are fed to the branch. Let us finally consider a sliding window \( \{t - t_0, \ldots, t + t_0\} \) centered on frame \( t \). Following previous works [28, 43], we assume that \( p_{0,j}, j \in J_f, i = t - t_0, \ldots, t + t_0 \) are independent and identically distributed samples from a Gaussian distribution (hereafter abbreviated as Gaussian for simplicity):
\[ Y_{s,f}^t = h_{ga}(\{p_{s,f}^t\}_{j=t-t_0}^{t}) \]  

where \( h_{ga} \) is the mapping of the GaussAgg layer, \( Y_{s,f}^t \) is the output of the GaussAgg layer.

The next layer ReEig [19] introduces non-linear transformations of SPD matrices via a mapping defined as:

\[ Y_{s,f}^t = h_r(X_{s,f}^t) = U \max(\epsilon, V)U^T, \]

where \( h_r \) is the mapping of the ReEig layer, \( X_{s,f}^t \) and \( Y_{s,f}^t \) are the input and output SPD matrices, \( X_{s,f}^t = UVU^T \) is the eigen-decomposition of \( X_{s,f}^t \), \( \epsilon \) is a rectification threshold, \( I \) is the identity matrix, \( \max(\epsilon, V) \) is a diagonal matrix whose diagonal elements are defined as:

\[ (\max(\epsilon, V))(i, i) = \begin{cases} V(i, i) & \text{if } V(i, i) > \epsilon \\ \epsilon & \text{if } V(i, i) \leq \epsilon. \end{cases} \]

After the ReEig layer, the LogEig layer [19] is used to map SPD matrices to Euclidean spaces. Formally, the mapping of this layer is defined as:

\[ Y_{s,f}^t = h_l(X_{s,f}^t) = \log(X_{s,f}^t) = U \log(V)U^T, \]

where \( h_l \) is the mapping of the LogEig layer, \( X_{s,f}^t \) and \( Y_{s,f}^t \) are the input and output SPD matrices, as before.

The next layer, referred to as VecMat, vectorizes SPD matrices by the following mapping [48]:

\[ y_{s,f}^t = h_{vm}(X_{s,f}^t) = |X_{s,f}^t(1, 1)|^2, \sqrt{2}X_{s,f}^t(1, 2), \ldots, \sqrt{2}X_{s,f}^t(d_{out}^e + 1, d_{out}^e + 1) \]

where \( h_{vm} \) is the mapping of the VecMat layer, \( X_{s,f}^t \in \mathbb{R}^{d_{out}^e + 1} \) is the input matrix, \( Y_{s,f}^t \) is the output vector, \( X_{s,f}^t(i, i), i = 1, \ldots, d_{out}^e + 1 \), are the diagonal entries of \( X_{s,f}^t \), and \( X_{s,f}^t(i, j), i < j, i, j = 1, \ldots, d_{out}^e + 1 \), are the off-diagonal entries.

We again assume that \( y_{s,f}^t, t = t_0^e, \ldots, t_e^e \), are independent and identically distributed samples from a Gaussian \( \mathcal{N}(y; \mu_{s,f}, \Sigma_{s,f}) \) whose parameters can be estimated as:

\[ \mu_{s,f} = \frac{1}{t_e^e - t_0^e + 1} \sum_{t = t_0^e}^{t_e^e} y_{s,f}^t \]

\[ \Sigma_{s,f} = \frac{1}{t_e^e - t_0^e + 1} \sum_{t = t_0^e}^{t_e^e} (y_{s,f}^t - \mu_{s,f})(y_{s,f}^t - \mu_{s,f})^T. \]

The second GaussAgg layer then performs the mapping:
where sub-sequence SPD matrix given as:

\[
\begin{bmatrix}
Y_{s,f} = h_{ga}(\{y_{s,f}^t\}_t=t_s^{k-1}, & s,f, j \in J_s)
= \begin{bmatrix}
\Sigma_{s,f} + \mu_{s,f}^T \mu_{s,f} \mu^f
\end{bmatrix},
\end{bmatrix}
\]

The resulting SPD matrix \(Y_{s,f}\) describes variations of finger \(f\) along sub-sequence \(s\).

3.4. Temporal-Spatial Gaussian Aggregation Sub-Network

Similarly to ST-GA-NET, TS-GA-NET is composed of 30 branches where each branch aggregates features for a sub-sequence of a specific finger. The sub-sequences are constructed in exactly the same way as ST-GA-NET. However, the feature aggregation procedure at the first and second GaussAgg layers are performed differently. More precisely, considering the branch associated with sub-sequence of equal length. Let \(t_{b,k}^s\) and \(t_{e,k}^s\), \(k=1, \ldots, N_s\), be the beginning and ending frames of these sub-sequences. Then for a given hand joint \(j \in J_f\) and sub-sequence \(k\), the first GaussAgg layer computes a SPD matrix given as:

\[
Y_{s,j}^k = \begin{bmatrix}
\Sigma_{s,j} + \mu_{s,j}^T\mu_{s,j}^k
\end{bmatrix},
\]

where \(\mu_{s,j}^k = \frac{1}{t_{e,k}^s - t_{b,k}^s + 1} \sum_{t=t_{b,k}^s}^{t_{e,k}^s} \mu_{s,j}^k\) and \(\Sigma_{s,j}^k = \frac{1}{t_{e,k}^s - t_{b,k}^s + 1} \sum_{t=t_{b,k}^s}^{t_{e,k}^s} (p_{s,j}^k - \mu_{s,j}^k)(p_{s,j}^k - \mu_{s,j}^k)^T\).

Note that \(Y_{s,j}^k\) encodes the first- and second-order statistics of hand joint \(j\) computed within sub-sequence \(k\). This temporal variation of individual joints is not captured by the first GaussAgg layer of ST-GA-NET. The resulting SPD matrices are processed through the ReEig, LogEig and VecMat layers. Let \(y_{s,j}^k\), \(k=1, \ldots, N_s\), \(j \in J_f\), be the output vectors of the VecMat layer of the branch. The second GaussAgg layer of TS-GA-NET then performs the following mapping:

\[
Y_{s,f} = h_{ga}(\{y_{s,j}^k\}_{j=1}^{N_j}),
= \begin{bmatrix}
\Sigma_{s,f} + \mu_{s,f}^T\mu_{s,f} \mu_{s,f}^T
\end{bmatrix}.
\]

where \(\mu_{s,f}\) and \(\Sigma_{s,f}\) can be estimated as:

\[
\mu_{s,f} = \frac{1}{N_j} \sum_{j=1}^{N_j} y_{s,j}^k,
\]

\[
\Sigma_{s,f} = \frac{1}{N_j} \sum_{j=1}^{N_j} (y_{s,j}^k - \mu_{s,f})(y_{s,j}^k - \mu_{s,f})^T.
\]

3.5. SPD Matrix Learning and Classification Sub-Network

The outputs of sub-networks ST-GA-NET and TS-GA-NET are sets of SPD matrices. The objective of the classification sub-network (see Fig. 6) is to transform those sets to a new SPD matrix, then map it to an Euclidean space for classification. The mapping \(h_{spda}\) of the SPDAgg layer is defined as:

\[
Y = h_{spda}(\{X_1, \ldots, X_N\}; W_1, \ldots, W_N)
= \sum_{i=1}^{N} W_i X_i(W_i)^T,
\]

where \(X_i \in \mathbb{R}^{d_{in} \times d_{in}}\), \(i = 1, \ldots, N\), are the input SPD matrices, \(W_i \in \mathbb{R}^{d_{out} \times d_{out}}\) are the transformation matrices, \(Y \in \mathbb{R}^{d_{out}}\) is the output matrix.

To guarantee that the output \(Y\) is SPD, we remark that the right-hand side of Eq. (19) can be rewritten as:

\[
\sum_{i=1}^{N} W_i X_i(W_i)^T = \hat{W} \text{diag}(X_1, \ldots, X_N)(\hat{W})^T,
\]
where $\hat{W} = [W_1, \ldots, W_N]$ and $\text{diag}(X_1, \ldots, X_N)$ is constructed such that its diagonal contains the diagonal entries of $X_1, \ldots, X_N$:

$$\text{diag}(X_1, \ldots, X_N) = \begin{bmatrix} X_1 & \cdots & X_2 & \cdots & \cdots & X_N \\ \vdots & \ddots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \vdots & \ddots & \ddots & \vdots \end{bmatrix}.$$ (21)

It can be easily seen that $\text{diag}(X_1, \ldots, X_N)$ is a valid SPD matrix, as for any vector $x \neq 0$, one has $x^T \text{diag}(X_1, \ldots, X_N)x = \sum_{i=1}^{N}(x_i)^TX_ix_i$, where $x = [(x_1)^T, \ldots, (x_N)^T]^T$ and the vectors $x_i, i = 1, \ldots, N$, have equal sizes. The right-hand side of the above equation is strictly positive since $(x_i)^TX_ix_i \geq 0, \forall i = 1, \ldots, N$, and there must exist $i^{'} \in \{1, \ldots, N\}$ such that $x_{i^{'}} \neq 0$ (as $x \neq 0$), which implies that $(x_{i^{'}})^TX_ix_{i^{'}} > 0$.

Inspired by [19], we assume that the combined matrix $W = [W_1, \ldots, W_N]$ is a full row rank matrix. Then optimal solutions of the transformation matrices are achieved by additionally assuming that $W$ resides on a compact Stiefel manifold $St(d_{out}^{s}, N \times d_{in}^{s})$. The transformation matrices $W_i$ are updated by optimizing $W$ and projecting the optimal $W$ on its columns. Note that the constraint on the dimension $d_{out}^{s}$ of the output $Y$ is: $d_{out}^{s} \leq Nd_{in}^{s}$.

To map the output SPD matrix of the SPDAgg layer to an Euclidean space, we use the LogEig layer, followed by a fully connected (FC) layer and a softmax layer.

### 3.6. Gesture Recognition

The SPDAgg layer outputs a matrix $B \in \mathbb{R}^{d_{out}^{s}}$ for each gesture sequence (see Fig. 6). This matrix is then transformed to its matrix logarithm and finally vectorized. The final representation of the gesture sequence is $v = [b_{1,1}, \sqrt{2}b_{1,2}, \sqrt{2}b_{1,3}, \ldots, \sqrt{2}b_{1,d_{out}^{s}}, b_{2,2}, \sqrt{2}b_{2,3}, \ldots, b_{d_{out}^{s},d_{out}^{s}}]^T$, where $b_{i,j}, i = 1, \ldots, d_{out}^{s}$, are the diagonal entries of $\log(B)$ and $b_{i,j}, i < j, i,j = 1, \ldots, d_{out}^{s}$, are the off-diagonal entries of $\log(B)$.

### 3.7. Relation with Previous Works

Our approach is closely related to [19, 54]. We point out in the following paragraphs the relations between the proposed network and those introduced in [19, 54].

- Our network considers temporal modeling for hand gesture recognition, while temporal modeling is not considered in [54] as they focus on image classification tasks. Moreover, in our work, a Gaussian is identified as a SPD matrix, while [54] identifies a Gaussian as the square root of a SPD matrix.
- Our network takes directly 3D coordinates of hand joints as input, while in [19], covariance matrices must be computed beforehand as input of their network.
- Our network relies not only on the second-order information (covariance) as [19] but also on the first-order information (mean). The first-order information has been proven to be useful in capturing the extra distribution information of low-level features [42]. Moreover, we consider the first- and second-order information for different subsets of hand joints, while [19] uses the whole set of joints to compute statistics. Our network is thus based on a finer granularity than [19].
- Our network combines two different and complementary architectures to better capture relevant statistics for the recognition task, which makes our network distinct from those of [19, 54].

### 4. Experiments

We conducted experiments using the Dynamic Hand Gesture (DHG) dataset [46, 47] and the First-Person Hand Action (FPHA) dataset [12]. In all experiments, the dimension of a output feature vector of the convolutional layer was set to 9 ($d_{out}^{s} = 9$), the dimensions of the transformation matrices of the SPDAgg layer were set to $200 \times 56$ ($d_{in}^{s} = 56$, $d_{out}^{s} = 200$). All sequences of the two datasets were normalized to have 500 frames ($N_{F} = 500$). The batch size and the learning rate were set to 30 and 0.01, respectively. The rectification threshold $\epsilon$ for the ReEig layer was set to 0.0001 [19]. The network trained at epoch 15 was used to create the final gesture representation. The classifier was learned using the LIBLINEAR library [9] with L2-regularized L2-loss (dual) where $C$ was set to 1, the tolerance of termination criterion was set to 0.1 and no bias term was added. For FPHA dataset, the non-optimized CPU implementation of our network on a 3.4GHz machine with 24GB RAM and Matlab R2015b takes about 22 minutes per epoch and 7 minutes per epoch for training and testing, respectively. In the following, we provide details on the experimental settings and results obtained for each dataset.

#### 4.1. Datasets and Experimental Settings

**DHG dataset.** The DHG dataset contains 14 gestures performed in two ways: using one finger and the whole
Table 1: Recognition accuracy (%) of our network for different settings of hand joint’s neighborhood.

<table>
<thead>
<tr>
<th>Num. of a hand joint’s neighbors</th>
<th>FPHA</th>
<th>DHG (14 gestures)</th>
<th>DHG (28 gestures)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>93.22</td>
<td>94.29</td>
<td>89.40</td>
</tr>
<tr>
<td>2</td>
<td>93.04</td>
<td>94.17</td>
<td>89.04</td>
</tr>
<tr>
<td>3</td>
<td>93.04</td>
<td>94.29</td>
<td>89.40</td>
</tr>
</tbody>
</table>

Table 2: Recognition accuracy (%) of our network for different settings of \( t_0 \).

<table>
<thead>
<tr>
<th>( t_0 )</th>
<th>FPHA</th>
<th>DHG (14 gestures)</th>
<th>DHG (28 gestures)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>93.22</td>
<td>94.29</td>
<td>89.40</td>
</tr>
<tr>
<td>2</td>
<td>93.04</td>
<td>94.17</td>
<td>89.04</td>
</tr>
<tr>
<td>3</td>
<td>93.04</td>
<td>94.29</td>
<td>89.40</td>
</tr>
</tbody>
</table>

Table 3: Recognition accuracy (%) of our network for different settings of \( N_S \).

<table>
<thead>
<tr>
<th>( N_S )</th>
<th>FPHA</th>
<th>DHG (14 gestures)</th>
<th>DHG (28 gestures)</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>93.33</td>
<td>94.29</td>
<td>89.40</td>
</tr>
<tr>
<td>20</td>
<td>92.87</td>
<td>94.05</td>
<td>88.93</td>
</tr>
<tr>
<td>25</td>
<td>92.70</td>
<td>94.29</td>
<td>89.04</td>
</tr>
</tbody>
</table>

Table 4: Recognition accuracy (%) of sub-networks ST-GA-NET and TS-GA-NET.

4.2. Ablation Study

In this section, we examine the influence of different components of our network on its accuracy. The default values of \( t_0 \) and \( N_S \) are set to 1 and 15, respectively.

**Hand modeling.** We evaluate the performance of our network when only physical connections of hand joints are used for the computations at the convolutional layer, i.e., connections between hand joints belonging to neighboring fingers are removed from the graph in Fig. 1(b). Each joint is now connected to at most three joints including itself. Results shown in Tab. 1 confirm that the use of connections other than physical connections of hand joints bring performance improvement.

**Time interval \( t_0 \).** In this experiment, we vary \( t_0 \) and keep other components of our network unchanged. To ensure that the computation of covariance matrices is numerically stable, we set \( t_0 > 0 \). Tab. 2 shows the performance of our network with three different settings of \( t_0 \), i.e. \( t_0 = 1, 2, 3 \). Results suggest that using 3 consecutive frames for the input of the first GaussAgg layer of ST-GA-NET is sufficient to obtain good performance.

**Number \( N_S \) of sub-sequences in a branch.** This experiment is performed by varying \( N_S \) while keeping other components of our network unchanged. For the same reason related to the computation of covariance matrices, \( N_S \) must be in a certain interval. We tested with \( N_S = 15, 20, 25 \). Results given in Tab. 3 indicate that our network is not sensitive to different settings of \( N_S \).

**Contribution of ST-GA-NET and TS-GA-NET.** We evaluate the performance of two networks, referred to as ST-HGR-NET and TS-HGR-NET by removing sub-networks ST-GA-NET and ST-GA-NET from our network, respectively. Results shown in Tab. 4 reveal that none of both ST-GA-NET and TS-GA-NET always provides the best performances on the datasets. This motivates the need for their combination using the component SPDC-NET and this contributes to the overall performance of our global network combining both TS-GA-NET and ST-GA-NET.

In the following, we report results obtained with default settings of \( t_0 \) and \( N_S \), i.e. \( t_0 = 1 \) and \( N_S = 15 \).

4.3. Comparison with State-of-the-Art

**DHG dataset.** The comparison of our method and state-of-the-art methods on DHG dataset is given in Tab. 5. The accuracy of the method of [19] is obtained by using the implementation provided by the authors with their default parameter settings. Our method significantly outperforms the competing ones. The network of [19] also learns a SPD matrix-based representation from skeletal data which is similar in spirit to our network. However, they concatenate the 3D coordinates of joints at each frame to create the feature vector of that frame, and their network’s input is the covariance matrix computed from feature vectors over the whole skeleton sequence. Thus, spatial and temporal relationships of joints are not effectively taken into account. By exploiting these relationships, our network improves the recognition accuracy by 19.05% and 19.76% compared to the results of [19] for experiments with 14 and 28 gestures, respectively. For more comparison of our method and existing methods, we conducted experiments using the leave-one-subject-out experimental protocol. Results on Tabs. 6 (14 gestures) and 7 (28 gestures) demonstrate that...
We observe that the method of \cite{Devanne et al.} performed by our method by more than 10 percent points. The remaining methods are outperformed by our network compared to the competing deep neural networks. For experiments with 14 and 28 gestures, respectively.

Table 7: Recognition accuracy comparison of our method and state-of-the-art methods on FPHA dataset. The best result in each column is marked in bold.

<table>
<thead>
<tr>
<th>Method</th>
<th>Year</th>
<th>Color</th>
<th>Depth</th>
<th>Pose</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>De Smedt et al. [46]</td>
<td>2016</td>
<td>✗ ✗</td>
<td>✔</td>
<td>✔</td>
<td>82.69</td>
</tr>
<tr>
<td>CNN+LSTM [36]</td>
<td>2018</td>
<td>✗ ✗</td>
<td>✔</td>
<td>✔</td>
<td>85.39</td>
</tr>
<tr>
<td>Weng et al., [55]</td>
<td>2018</td>
<td>✗ ✗</td>
<td>✔</td>
<td>✔</td>
<td>82.00</td>
</tr>
<tr>
<td>ST-TS-HGR-NET</td>
<td></td>
<td>✗ ✗</td>
<td>✔</td>
<td>✔</td>
<td>91.28</td>
</tr>
</tbody>
</table>

Table 8: Recognition accuracy comparison of our method and state-of-the-art methods on FPHA dataset. The best result in each column is marked in bold.

<table>
<thead>
<tr>
<th>Method</th>
<th>Year</th>
<th>Color</th>
<th>Depth</th>
<th>Pose</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two stream-color [10]</td>
<td>2016</td>
<td>✔</td>
<td>✗ ✗</td>
<td>✗</td>
<td>61.56</td>
</tr>
<tr>
<td>Two stream-flow [10]</td>
<td>2016</td>
<td>✔</td>
<td>✗ ✗</td>
<td>✗</td>
<td>69.91</td>
</tr>
<tr>
<td>Two stream-all [16]</td>
<td>2016</td>
<td>✔</td>
<td>✗ ✗</td>
<td>✗</td>
<td>75.30</td>
</tr>
<tr>
<td>HOG'-depth [39]</td>
<td>2013</td>
<td>✗</td>
<td>✔</td>
<td>✗</td>
<td>59.83</td>
</tr>
<tr>
<td>HOG'-depth+pose</td>
<td>2013</td>
<td>✗</td>
<td>✔</td>
<td>✗</td>
<td>66.78</td>
</tr>
<tr>
<td>HON4D [40]</td>
<td>2013</td>
<td>✗</td>
<td>✔</td>
<td>✗</td>
<td>70.61</td>
</tr>
</tbody>
</table>

1-layer LSTM [62] 2016 ✗ ✗ ✔ 78.73
2-layer LSTM [62] 2016 ✗ ✗ ✔ 80.14
Moving Pose [59] 2013 ✗ ✗ ✔ 56.34
Lie Group [49] 2014 ✗ ✔ ✗ 82.69
HBRNN [6] 2015 ✗ ✗ ✔ 77.40
Gram Matrix [61] 2016 ✗ ✗ ✔ 85.39
JOULE-color [18] 2015 ✗ ✗ ✔ 66.78
JOULE-depth [18] 2015 ✗ ✗ ✔ 60.17
JOULE-pose [18] 2015 ✗ ✗ ✔ 74.60
JOULE-all [15] 2015 ✗ ✗ ✔ 78.78
Huang et al. [19] 2017 ✗ ✗ ✔ 84.35
Huang et al. [23] 2018 ✗ ✗ ✔ 77.57
ST-TS-HGR-NET ✗ ✗ ✔ 93.22

5. Conclusion

We have presented a new neural network for hand gesture recognition that learns a discriminative SPD matrix encoding the first-order and second-order statistics. We have provided the experimental evaluation on two benchmark datasets showing that our method outperforms state-of-the-art methods.

Acknowledgments. This material is based upon work supported by the European Union and the Region Normandie under the project IGIL. We thank Guillermo Garcia-Hernando for providing access to FPHA dataset [12].

References


