Lightweight Deep Symmetric Positive Definite Manifold Network for Real-Time 3D Hand Gesture Recognition

Mostefa Ben Naceur, Luc Brun, Olivier Lézoray

Lightweight Deep Symmetric Positive Definite Manifold Network for Real-Time 3D Hand Gesture Recognition

Mostefa Ben Naceur, Luc Brun, Olivier Lézoray
UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France
mostefa.bennaceur@esiee.fr, luc.brun@ensicaen.fr, olivier.lezoray@unicaen.fr

***Abstract*— This paper proposes a new neural network based on Symmetric Positive Definite (SPD) manifold learning for real-time skeleton-based hand gesture recognition. The transformation of the input skeletal data into SPD matrices allows to encode efficiently high-order statistics such as covariances or correlations between the joints' features. These matrices are combined and transformed by our deep neural network which is thus constrained to work on the manifold of such matrices. The online recognition is performed using two sliding windows moving along the gesture's stream in order to simultaneously detect and classify the occurrence of a new gesture within the stream. The proposed network is validated on a challenging dataset and shows state-of-the-art performances both in terms of accuracy and inference time.**

I. INTRODUCTION

Deep neural networks have seen great success in recent years, where most of the traditional methods have been outperformed by deep learning architectures, especially in the area of computer vision and handcrafted features. Convolutional neural networks (CNNs) [1], [2] are composed of alternating convolution and pooling layers and finally applying a multi-layered perceptron to perform the classification step. Other techniques that have encountered great success in many image- and video-based pipelines are Symmetric Positive Definite (SPD) matrices. Such matrices have the ability to learn non-linear relationships between input features. They have been used in many tasks and areas of computer vision [40], [41], [42], [43] such as medical imaging [3] or pedestrian classification and detection [4]. In this paper, we empower deep neural network architectures with statistical representations using SPD matrices learning to solve the issue of offline and online hand gestures recognition.

Hand gesture recognition is the process of detecting and classifying human hands into several categories. This task is one of the fundamental blocks in many applications such as human-robot interaction [7], video games [6], sign language interpretation [5], security and surveillance [8]. The issue of hand gesture recognition is solved by traditional methods, usually using sophisticated and expensive RGB and depth cameras. However, with the recent development of low-cost sensing cameras such as Microsoft Kinect, Intel Realsense, it is possible to obtain skeletal data of human hands (Fig. 1) or bodies with high precision. This kind of devices give accurate information without too much processing or computing resources. Conventional non-deep learning based works for the recognition of hand gestures or human actions are mainly based on modeling the movement of skeletal data over time using Dynamic Time Warping [9], Fourier Temporal Pyramid [10], Moving pose KNN [11]. Skeletal data has been used by deep learning methods in different forms and representations. The most effective methods are based on CNN networks [34], [35], [36], [39], RNN /LSTM networks [37], [38], [39] and more recently graph convolutional networks [24], [25]. Deep learning-based methods are powerful for capturing both spatial and temporal features as well as the relationships between joints. In addition, they can be applied to frames or sequences of frames (i.e., videos). This line of methods achieved high performance for hand/action gesture recognition. However, they are limited and constrained by the input's shape with the need for deep architectures to extract highly representative and discriminating features. Despite the performance of deep learning architectures in terms of accuracy, it is hard to design a real-time system based on these architectures. In addition, these methods do not allow straightforwardly to encode high-order statistics such as correlations or variances between features. This last point is a severe drawback in an application where several joints evolve together in order to perform a gesture. Son et al. [64] have proposed a deep learning solution based on SPD matrices to handle the task of hand gesture recognition. This solution explicitly encodes high-order relationships between features but remains too slow for real-time applications. It is thus designed especially for offline hand gesture recognition. In this paper, we propose a deep architecture based on the 3D pose of the hand's joints for real-time Hand Gesture Recognition. Inspired by [64], [45], we first propose an efficient deep neural network based on SPD matrices in order to get a rich description of each sub-sequence of a gesture. These SPD matrices extract temporal and local information about the movement and trajectories of the hand's joints over a time segment and around an instantaneous 3D pose. Such local and temporal information could be seen as a generalization of the moving pose descriptor [11], and the action snippet [48]. Our proposed network is then combined with a novel architecture in order to perform online gesture recognition.

In the following, we present in Section II the main methods of the state-of-the-art. Our sub-network provides a description of each sub-sequence in Section III, and our new architecture for online recognition is described in Section IV. The resulting method is evaluated and compared to the state-of-the-art in Section V. Finally, the conclusion and the

978-1-6654-3176-7/21/$31.00 ©2021 IEEE

perspectives are described in Section VI.

II. Related work

In this section, we present the state-of-the-art methods for the task of hand/action gesture recognition.

Skeleton Features: Traditional methods of a hand gesture or action recognition are focused on handcrafted features, that are applied for specific tasks. These methods extract statistical properties from input data before applying a discriminative learning method to classify this input. These features are based on geometric low-level properties [27], [28], [29] or complex relationships between pairs of joints [30], [9], [32]. Also, these features can be classified as informative [33] or non-informative. However, despite the intensive efforts spent to design efficient and complex features, the results obtained by these methods are now outperformed by methods learning an appropriate representation from data.

Deep Learning on Skeletal Data: After the breakthrough of deep learning, many recent approaches have been proposed for hand gesture, human action, and human activity recognition using either images or videos. Most of these approaches are based on 3D convolutional neural network (3DCNN) [13], using e.g., a two-level hierarchical structure [15], a 3DCNN with recurrent neural networks [14] such as R3DCNN [12], C3D+LSTM [16], and CNN-LSTM [26]. Other methods for the task of gesture and action recognition are based on sequence-to-sequence models using GRU networks [18], [19], [20]. The results and performance achieved by DeepGRU [20] are mainly due to using : i) a GRU network to capture the long term dependencies and the variations over time of the hand/action joints between the frames, and ii) to an attention module that detects which frame is more important than other frames. Avola et al. [21] developed a deep learning method with 4 stacked layers of LSTM Networks [17] to map the internal angles features of the input into classes for Sign Language and Semaphoric Gesture Recognition with a Leap Motion device to track the joints. Then, as they proved that angle features are not sufficient to accurately model dynamic gestures in 3D space, they added other types of features: 3D displacements of the hand's palm to manage hand translation, 3D displacements of the fingertips position to manage hand rotation, intra-finger angles between two consecutive fingers to manage static gestures. Zhang et al. [22] developed a view adaptive approach based on a RNN-LSTM architecture with 4 stacked layers of LSTM Networks for human action recognition using 3D skeleton data. Zhang et al. [22] built an architecture made of two parts. The first part determines the observation viewpoint by translating (shifting) and rotating the frames from a camera coordinate system (original frames) to a new coordinate system where the body is the center. The second part performs feature extraction and action classification. This architecture regulates the viewpoint of each action sequences to be more consistent with a new viewpoint. This transformation helps to solve the limits of using different camera position to capture the same posture from different viewpoints. Zanfir et al. [23] proposed a moving pose (MP) descriptor-based on dynamic representation of the body features for human action and activity recognition. The descriptor captures information about the 3D pose of the body's joint in addition to speed and acceleration within a short time window around the current frame. They assumed that the 3D pose is a continuous and differentiable function of the body joint positions over time, therefore, by using Taylor approximation, they numerically estimate the speed and the acceleration as the first and second order derivatives, of the pose. Thus, the descriptor captures information on the action in the current frame and within the temporal neighborhood of the given frame. The MP method is based on a modified version of a KNN classifier trained in a supervised learning manner. These methods [22], [21], and [23] are limited as 1) they only consider physical connections of hand/body joints, 2) they do not model correlations between joints 3) they are specially designed for offline gesture recognition.

Deep Learning on SPD Matrices: SPD matrices have been used in many computer vision tasks [40], [41], [42], [43]. They have the ability to learn appropriate statistical representations. The property of this kind of matrices is that they deal with non-Euclidean domains [44], [45], [46], where the problem of SPD matrices is that most of the classical deep learning approaches such as CNNs, LSTM cannot handle this kind of matrices. In order to tackle this issue, we need either to map the SPD manifolds into Euclidean domains, or to design specific layers [50], [51], [45]. As applying Euclidean geometry to SPD matrices can result in undesirable effects [3], it is more interesting to design a deep learning architecture to non-linearly learn SPD matrices on Riemannian manifolds [45].

III. Deep SPD Network

Inspired by [64], [45], we present in this section our Deep SPD network (Deep SPDNet) for solving the issue of offline hand gestures recognition from 3D skeletons.

Convolution layer:

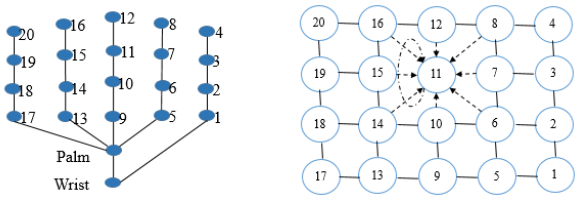

Fig. 1. (left) Illustration of hand joints, and (right) their corresponding 2D grid. This 2D grid is used as an input to our proposed pipeline.

Our Deep SPDNet is composed of 7 different layers. First, it starts with a 2D convolution layer as in CNNs networks [1], [2]. The filter weights are shared over all frames of the sequence in the dataset. The aim of this layer is to extract features and to model the relationships between hand joints. To this end, the convolution layer receives as input a 2D grid with 3 channels, i.e., x, y, and z coordinates of hand joints and outputs another grid with several channels. To build the 2D grid, we remove the palm and wrist joints from each frame. Then, the joints from 1 to 20 form a 2D grid 4×5

(i.e., 4 joints $\times$ 5 fingers) that is used as an input to the convolution layer. Each node of the 2D grid has at most 9 neighbors including itself (Fig. 1). The output feature vector at each node is computed as a spatial convolution from its neighboring nodes:

$$X_k^t = \sum_{j \in \mathcal{N}_i} W_{k,i} X_{k-1,j}^t \tag{1}$$

where $X_{k-1,j}^t \in \mathbb{R}^3$ is the input 3D coordinates of the hand joint j at frame t and for the $k-th$ layer. $W_k \in \mathbb{R}^{d_{k-1}}$ is the filter matrix. $X_k \in \mathbb{R}^{d_{out}}$ is the output feature map. $\mathcal{N}_i$ is the set of neighbors of node i (Fig. 1). The convolution layer is directly applied to the input 2D grid representing skeletal data that lie in Euclidean domain.

Gaussian Mapping layer (Gmap): After applying the convolutional layer on each skeleton grid we decompose the sequence into 6 subsequences (s) for each finger (f). The first subsequence is the original skeleton sequence. Then, we divide the sequence into two and three subsequences of equal lengths. This operation can be interpreted as the construction of a pyramid of sequences at different resolutions. This allows to capture temporal variations at different levels. Therefore, we obtain a total of 30 branches, i.e., $s = 6$ subsequences for $f = 5$ fingers. Let $\mathcal{J}_f$ denote the set of joints of a finger f. In order to characterize the temporal variations of each joint $j \in \mathcal{J}_f$ in each frame of a subsequence s, we divide s into N subsequences (sb) of equal length. Let $t_{b,sb}^s$ and $t_{e,sb}^s$ be the beginning and ending frames of a subsequence $sb \in \{1, ..., N\}$. The temporal variations of a joint $j \in \mathcal{J}_f$ at frame sb within the sequence s is defined by the Gaussian mapping layer (Gmap) that provides the SPD matrix $X_{s,j}^{sb}$ defined as follows:

$$\mathbf{X}_{s,j}^{sb} = \begin{bmatrix} \Sigma_{s,j}^{sb} + \mu_{s,j}^{sb}(\mu_{s,j}^{sb})^T & \mu_{s,j}^{sb} \\ (\mu_{s,j}^{sb})^T & 1 \end{bmatrix} \tag{2}$$

where $\mu_{s,j}^{sb}$ is the mean vector, $\Sigma_{s,j}^{sb}$ is the covariance matrix. These two parameters (i.e., mean and covariance) of the Gaussian distribution $\mathcal{N}(p; \mu, \Sigma)$ are estimated as follows:

$$\mu_{s,j}^{sb} = \frac{1}{t_{e,sb}^s - t_{b,sb}^s + 1} \sum_{t=t_{b,sb}^s}^{t_{e,sb}^s} p_{s,j}^t \tag{3}$$

where $p_{s,j}^t$ denotes the coordinates of joint $j \in J_f$ at frame t of subsequence s.

$$\Sigma_{s,j}^{sb} = \frac{1}{t_{e,sb}^s - t_{b,sb}^s + 1} \sum_{t=t_{b,sb}^s}^{t_{e,sb}^s} (p_{s,j}^t - \mu_{s,j}^{sb})(p_{s,j}^t - \mu_{s,j}^{sb})^T \tag{4}$$

Eigenvalue Rectification layer: Eigenvalue rectification (Reig) layer [45] is designed specifically to introduce non-linearity for SPD matrices. Reig layer is defined as the following mapping:

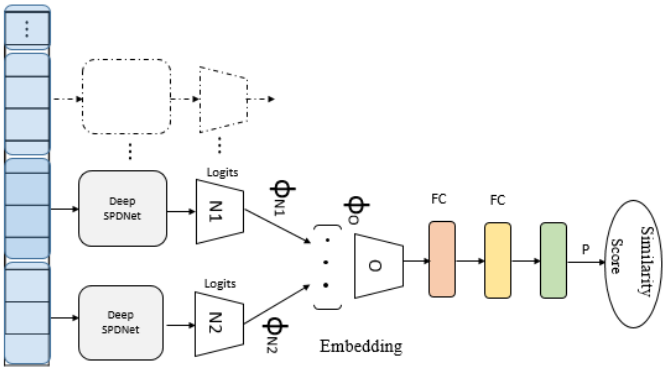

Fig. 2. Illustration of the pipeline Deep SPDNet and TDN network for online hand gesture recognition. The pipeline starts by extracting the embedding of each clip using Deep SPDNet before detecting the dissimilarity / similarly using TDN network.

$$X_{s,j,k}^{1,sb} = f_r(X_{s,j}^{sb}) = U_{k-1} max(\epsilon I, V_{k-1}) U_{k-1}^T \tag{5}$$

Where f_r is the Reig rectification function, $X_{s,j}^{1,sb}$ and $X_{s,j,k}^{sb}$ are respectively the pre- and post-activation SPD matrices, ϵ is a rectification threshold, I is an identity matrix, $X_{s,j}^{sb} = U_{k-1} V_{k-1} U_{k-1}^T$ is the eigenvalue decomposition of $X_{s,j}^{sb}$, and $M = max(\epsilon I, V_{k-1})$ is a diagonal matrix whose diagonal elements are computed as follows:

$$M(i,i) = \begin{cases} V_{k-1}(i,i), & V_{k-1}(i,i) > \epsilon, \\ \epsilon, & V_{k-1}(i,i) \leq \epsilon. \end{cases} \tag{6}$$

Reig function prevents SPD matrices to have negative or zero values in addition to adjusting their small positive eigenvalues.

Eigenvalue Logarithm layer:

The objective of this eigenvalue logarithm layer [45] (LogEig) is to map SPD matrices to Euclidean spaces. The mapping of this layer is defined as:

$$X_{s,j}^{2,sb} = f_l(X_{s,j}^{1,sb}) = log(X_{s,j}^{1,sb}) = U_{k-1} log(V_{k-1}) U_{k-1}^T \tag{7}$$

Where f_l is the LogEig function, $X_{s,j}^{1,sb} = U_{k-1} V_{k-1} U_{k-1}^T$ is an eigenvalue decomposition, $log(V_{k-1})$ is a diagonal matrix.

Matrix Vectorization layer: After applying the LogEig layer we can apply a matrix vectorization layer [52] (VecMat). VecMat vectorizes the input matrices by applying a linear transformation to convert only the upper triangular matrices[1] into vectors as the following:

$$\begin{aligned} x_{s,j}^{sb} = f_v(X_{s,j}^{2,sb}) = [X_{s,j}^{2,sb}(1,1), \sqrt{2} X_{s,j}^{2,sb}(1,2) \\ , ..., \sqrt{2} X_{s,j}^{2,sb}(1, d_{out}^c), X_{s,j}^{2,sb}(2,2), \sqrt{2} X_{s,j}^{2,sb}(2,3) \\ , ..., X_{s,j}^{2,sb}(d_{out}^c, d_{out}^c)]^T \end{aligned} \tag{8}$$

[1]This is due to the symmetric property of the matrices.

Where f_v is the VecMat function, $X^{2,sb}_{s,j}(i,i)$, is the diagonal entries, $X^{2,sb}_{s,j}(i,j), i<j$, is the upper part of $X^{2,sb}_{s,j}$.

After applying VecMat function, we obtain a set of $\{x^{sb}_{s,j}\}^{sb=1,...N}_{j\in\mathcal{J}_f}$ vectors characterizing the temporal variation of each joint on each frame of s. In order to obtain a global characterization of finger f along the subsequence s we aggregate these vectors into SPD matrices using the Gmap operator as follows:

$$X_{s,f} = f_g(\{x^{sb}_{s,j}\}^{sb=1,...N}_{j\in\mathcal{J}_f}) = \begin{bmatrix} \Sigma_{s,f} + \mu_{s,f}(\mu_{s,f})^T & \mu_{s,f} \\ (\mu_{s,f})^T & 1 \end{bmatrix} \tag{9}$$

Where the mean (μ) and covariance (Σ) are defined as follows:

$$\mu_{s,f} = \frac{1}{N|\mathcal{J}_f|}\sum_{j\in\mathcal{J}_f}\sum_{sb=1}^{N} x^{sb}_{s,j} \tag{10}$$

$$\Sigma_{s,f} = \frac{1}{N|\mathcal{J}_f|}\sum_{j\in\mathcal{J}_f}\sum_{sb=1}^{N}(x^{sb}_{s,j}-\mu_{s,f})(x^{sb}_{s,j}-\mu_{s,f})^T \tag{11}$$

Bilinear Mapping layer:

After this last Gmap layer we obtain a set of SPD matrices encoding the variations of each finger f along each sub sequence s. The aggregation of these matrices into a single SPD matrix encoding the whole gesture induces a notion of weighing in order to privilege the most relevant matrices for hand gesture recognition. In order to simplify the notations, let us denote $(X_{s,f})_{(s,f)\in\{1,...,6\}\times\{1,...,5\}}$ by $(X_i)_{i\in\{1,...,N\}}$ with $N = 30$. The Bilinear Mapping layer [45] (BiMap) transforms these SPD matrices into a single SPD matrix as follows:

$$\begin{aligned} X &= f_b(X_1,...,X_N;W_1,...,W_N) \\ &= \sum_{i=1}^{N} W_i X_i W_i^T \end{aligned} \tag{12}$$

Where $X_i \in \mathbb{R}^{d_{k-1}\times d_{k-1}}$ are the input SPD matrices, $W_i \in \mathbb{R}^{d_k\times d_{k-1}}$ are the transformation matrices[2], $X \in \mathbb{R}^{d_k\times d_k}$ is the output matrix. The BiMap layer can roughly be understood as an attention network defined on SPD matrices that insures that the resulting matrix lies on the SPD Riemannian manifold.

A vector providing a rich summary of the whole sequence is then obtained by applying LogEig and VecMat layers on the matrix X. In the context of offline gesture recognition, this vector may be combined with a softmax layer in order to obtain the final classification scores. Let us note that the softmax layer may be replaced by an SVM during the test phase while the softmax layer is used during training phase.

[2]The weight matrices W_i are assumed to be semi-orthogonal and row full-rank matrices.

IV. Online Gesture Recognition:

Online gesture recognition is a challenging issue due to the fact that (i) the input stream of hand's joint positions does not contain any indication of the start and end of each gesture, (ii) the architecture should be efficient in terms of memory consumption and computational complexity, iii) gestures may contain undesired hand motions (i.e., gestures that do not belong to any class).

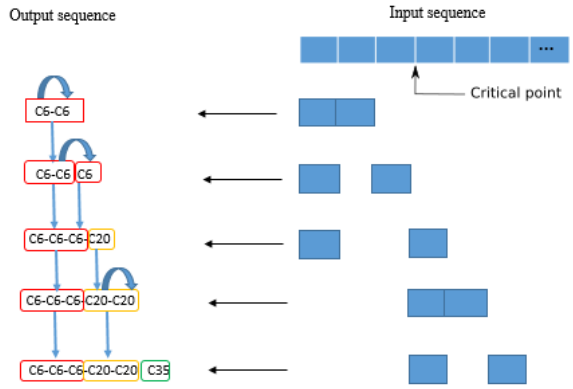

Fig. 3. The pipeline of Deep SPDNet and TDN transforms skeletons sequence into hand gestures sequences. Each blue block represents a 10-frame clip. C6 means the class 6.

In order to tackle these issues, we combine two head networks for online gesture recognition (Fig. 2). Each head network is a Deep SPD Network, as defined in Section III. The combination of these two heads networks is performed by a Multi Layer Perceptron that takes as input the vectorial encoding provided by the BiMap layer of a Deep SPD Network. It is denoted in the rest of this paper as Temporal Detection Network (TDN). The pipeline that we propose in this section receives sequentially a stream of hand's joint positions as 10-frame clips. Since the nucleus phase of the hand gesture spans multiple clips, we measure the percentage of similarity/dissimilarity between two clips. The first head network points to the first clip of a stream while the second head network points to the next clip and sequentially moves along the stream (Fig. 3). For each position of the two heads, the network Deep SPDNet defined in Section III is used according to the architecture defined in Fig. 2. This enables to obtain a vectorial encoding of each clip and to determine if they correspond to a same gesture. The point where our network detects two different gestures is called a critical point (Fig. 3). Once a critical point has been detected, the first head network moves on to this point while the second head network is shifted of 10 frames. This process is iterated until the end of the stream. Fig 3 shows our proposed pipeline, where the input of the pipeline receives a stream of 10-frame clips (blue block in Fig 3).

For the task of offline hand gestures recognition, the Deep SPDNet network is trained on sequences describing a complete gesture. In our experiments, each sequence is normalized so as to take 500 frames [64].

For the task of online hand gesture recognition, the inference time and early detection are crucial properties compared to offline hand gestures recognition. Thus, the training phase in this task is done in three steps. First, instead of taking complete sequences of 500 frames, we train the Deep SPDNet network on subsequences (clips) composed of 10 frames. Second, Deep SPDNet is used to obtain a vectorial encoding of each incoming clip of the stream. In the third step, the TDN network is trained on a different combination of positive and negative pairs. Positive pairs correspond to clips of different gestures while negative clips correspond to the opposite, i.e., the same gesture. These pairs of positive and negatives clips[3] are chosen randomly while taking into account an equivalent number of both types of clips to avoid the issues of unbalanced dataset and combinatorial explosion.

For the evaluation phase, we measure the performance of our proposed architectures for both tasks: offline and online hand gestures recognition, where the evaluation is measured in terms of many evaluation metrics, in particular in terms of accuracy and inference time, that are critical for online hand gesture recognition.

V. EXPERIMENTS AND DISCUSSIONS

In order to evaluate our proposed architectures for offline and online hand gestures recognition, we conduct a series of experiments on the First-Person Hand Action (FPHA) dataset [53]. We used the settings of [45] for the following hyper-parameters: the learning rate λ is 10^{-2}, the batch size is set to 30, the connection weights size of BiMap is set to 200×56 with random semi-orthogonal initialisation, the rectification threshold ϵ is set to 10^{-4}. The number of convolution filters is set to 9, and please note that adding more filters hurts the performance. For the SVM model, we used LIBLINEAR library [54] with L2 loss-regularization, where C is set to 1, and the tolerance of termination criterion is set to 10^{-1}. For training our architectures, we use Matlab environment with i7 PC without any GPUs.

FPHA dataset: we use in these experiments the famous First-Person Hand Action dataset, which is very challenging and widely used for hand gesture recognition. It contains 1175 videos of hand gestures divided into 600 sequences for training and 575 for testing, as proposed by [53] for the evaluation step. Each video belongs to one of the 45 classes, also, it is performed by 6 actors in three different scenarios. The dataset provides for each video the 3D (x, y, and z) coordinates of 21 hand joints, e.g., a video clip with 10 frames has 21*10 3D coordinates.

Table I summarizes the results of our proposed Deep SDPNet architecture and of the state-of-the-art methods on the offline hand gestures recognition task. As it is shown, the proposed architecture outperforms most of the state-of-the-art methods by a large margin, i.e., more than 5.5% compared to the method [55] and more than 6.6 % compared to the method[4] [45]. This reveals the importance of deep architecture property and the SPD matrices to improve the accuracy of the state-of-the-art methods on the task of hand gestures recognition. Moreover, as one can see from Table I, the 3D pose coordinates are capable to provide better features than the use of all 3 modalities (Color, depth, and pose) as the case with the method [57]. The method of [65] achieved high performance in terms of accuracy, but it does not leverage the huge parameters of the two-stream network to outperform our method. The method of [64] and our proposed Deep SPDNet network have quite close results to each other. However, the Deep SPDNet network is specifically designed to work for both tasks (i.e., offline and online) in real-time, where the method [64] is developed only for offline hand gestures recognition. Moreover, Deep SPDNet only needs 0.15 seconds as inference time to predict the class of each gesture.

TABLE I

COMPARISON OF THE RECOGNITION RESULTS OF OUR PROPOSED ARCHITECTURE AND THE STATE-OF-THE-ART METHODS ON THE FPHA DATASET. NUMBERS IN BOLD ARE THE BEST VALUES.

Method	Color	Depth	Pose	Accuracy(%)
HON4D [56]	✗	✓	✗	70.61
Lie Group [9]	✗	✗	✓	82.69
HBRNN [37]	✗	✗	✓	77.40
JOULE-all [57]	✓	✓	✓	78.78
Two stream-all [58]	✓	✗	✗	75.30
Novel View [59]	✗	✓	✗	69.21
LSTM [60]	✗	✗	✓	80.14
Gram Matrix [55]	✗	✗	✓	85.39
T Forests [61]	✗	✗	✓	80.69
M Learning [45]	✗	✗	✓	84.35
G Manifolds [62]	✗	✗	✓	77.57
ST-TS-HGR-NET [64]	✗	✗	✓	**93.22**
Two-stream NN [65]	✗	✗	✓	90.26
Deep SPDNet	✗	✗	✓	**90.96**

In order to evaluate our proposed pipeline of Deep SDPNet and TDN networks for the task of real-time online hand gesture recognition, we used the videos of hand gestures of FPHA dataset to extract features from the BiMap layer of Deep SDPNet networks. Then, these features are normalized with Z-score normalization, i.e., with zero mean and unit standard deviation to avoid the issue of activation function oscillation. After that, they are used to create 5 datasets with different sequences lengths, each dataset has sequences from 10 to 160 successive clips (100 to 1600 frames). Also, the created datasets are composed of 1000 or 2000 sequences.

The TDN network makes a prediction to detect whether the features from two clips do correspond or not to a same gesture. If they belong to a same gesture, then the TDN activates the Deep SDPNet to classify the first clip, and since the two clips are considered as similar we assume that they correspond to a same gesture. In order to validate this last assumption, we evaluate in Table II the performances of the binary classification performed by the TDN network using the construction scheme of pairs of clips defined in section IV. In this case, the set of clip pairs is split into 3 subsets: training set with 64% of data, validation set with 20% of data, test set with 16% of data. The accuracy obtained on the test set is of 98%, hence confirming our

[3]Each clip has 10 frames.

[4]The results are obtained by using the original implementation with the default settings.

initial assumption. This mapping of the two clips on a same gesture avoids the inference time required to recognize the class of the second clip. If the features of the two clips are different, then the TDN Network activates the Deep SDP Network for the two clips. So, if the number of P_c classes is less than the number of successive gestures in one sequence P_g, i.e., $P_c < P_g$, the inference time will be reduced by more than a half, since the pipeline needs only to detect different gestures (i.e., the critical points), while the same gestures will have the same class. Moreover, TDN network measures the similarity/dissimilarity of two clips only in 0.03 seconds.

TABLE II

PERFORMANCE OF TDN NETWORK IN TERMS OF DIFFERENT EVALUATION METRICS. TP: TRUE POSITIVES, FP: FALSE POSITIVES, TN: TRUE NEGATIVES, FN: FALSE NEGATIVES

Dataset	TP	FP	TN	FN	Accuracy (%)
Training	10075	188	9301	93	98.57
Validation	3121	83	2914	26	98.22
Test	2495	68	2322	30	98

In Table III, we measure the performance of the proposed pipeline to detect and recognize sequences of clips from the FPHA dataset. First, we measure the effect of the number of clips per sequence. As shown in Table III, adding more clips does not strongly hurt performances. For 10 C/S (number of clips per sequence) the performance of the pipeline is 91.28%, and when the number of gestures is multiplied by 16 (i.e., 160 C/S), the performance reached 88.35%, so it decreased only by 2.93%, which is acceptable in the considered application. Second, we measured the effects of the number of sequences on the pipeline. As shown in Table III doubling the number of sequences increases the performance by 1% to 2%.

TABLE III

MEAN ACCURACY PERFORMANCE OF THE PROPOSED PIPELINE TO DETECT AND RECOGNIZE THE SEQUENCES OF CLIPS. ± STANDARD DEVIATION, C/S: MEANS CLIPS PER SEQUENCE. EACH RESULT IN THE TABLE IS AVERAGED OVER 10 EXPERIMENTS USING 600 CLIPS.

# of sequences	10 C/S	20 C/S	40 C/S	80 C/S	160 C/S
1000	91.28(±0.77)	89.7(±1.06)	88.11(±0.83)	87.58(±1.43)	88.35(±0.96)
2000	91.71(±0.65)	90.44(±1.4)	89.78(±1.5)	89.34(±1.56)	89.09(±1.51)

In Table IV, we compare and add more test to the best results in Table III. This time, we progressively increase the number of clips (i.e., 600, 900, and 1175) that can be used to generate the sequences. From each number of clips, we generate randomly 2000 sequences. As shown in Table IV, for 10 C/S the best accuracy, obtained for 600 clips (line 1) is equal to 91.71%. Adding additional 300 clips (i.e. 900 clips, line 2) decreases the accuracy by only 2.72%. Moreover, when we use all the clips of the dataset to generate the sequences (third row), the performance decreases by only 6.44% to 85.27%. While the experiments performed in Table I and Table III do not follow the same protocol and are thus not fully comparable, we may however note that our last result of 85.27% remains competitive compared to most of the methods of the state of art.

TABLE IV

MEAN ACCURACY PERFORMANCE OF THE PROPOSED PIPELINE TO DETECT AND RECOGNIZE THE SEQUENCES OF CLIPS. THE RESULTS ARE MEANS OVER 10 RUNS, ± STANDARD DEVIATION. C/S: MEANS CLIPS PER SEQUENCE. EACH RESULT IN THE TABLE IS AVERAGED OVER 10 EXPERIMENTS USING 600, 900, AND 1175 CLIPS, RESPECTIVELY.

10 C/S	20 C/S	40 C/S	80 C/S	160 C/S
91.71(±0.65)	90.44(±1.4)	89.78(±1.5)	89.34(±1.56)	89.09(±1.51)
88.99(±0.92)	87.61(±1.57)	86.36(±2.23)	85.37(±2.59)	84.72(±2.66)
85.27(±0.88)	83.24(±2.15)	81.43(±3.14)	79.76(±4.05)	78.14(±4.90)

In order to model the different hand positions and motions during the prestroke and poststroke phases, we add randomly a percentage of noise to the generated sequences. To do so we randomly insert a given percentage of random frames within all test sequences. The joint coordinates of these inserted frames are generated from the range of the original sequences. Table V shows the performance of our proposed pipeline in the presence of noise. We set the number of clips to 1175 (i.e., all data in the dataset) and the number of sequences to 2000 sequences. The percentage of random frames is progressively increased from 10% to 20%. The results, summarized in Table V, show that for 10 C/S, adding 10% of noise (i.e., 117 sequences of randomly generated noise) to the fixed number of clips (i.e., 1175), decreases the performance by 17% compared the best results (i.e., 91.71%). Moreover, adding 20% of noise (i.e., 235 sequences of randomly generated noise) decreases the performance only by 27%.

TABLE V

MEAN ACCURACY PERFORMANCE OF THE PROPOSED PIPELINE TO DETECT AND RECOGNIZE THE SEQUENCES OF CLIPS. THE RESULTS ARE MEANS OVER 10 RUNS, ± STANDARD DEVIATION. C/S: MEANS CLIPS PER SEQUENCE. EACH RESULT IN THE TABLE IS AVERAGED OVER 10 EXPERIMENTS USING 1175 CLIPS AND 2000 SEQUENCES..

Noise (%)	10 C/S	20 C/S	40 C/S	80 C/S	160 C/S
10	74.18(±2.39)	69.84(±5.23)	63.68(±9.89)	55.8(±16.28)	48.76(±20.65)
20	64.57(±1.84)	57.59(±7.45)	49.63(±12.93)	41.39(±18.21)	34.37(±21.53)

VI. CONCLUSION

In this paper, we proposed a novel Deep SPD neural networks classifier and TDN networks to solve the issue of offline and online hand gestures recognition. Experiments show that a stream of 10-frame clips is sufficient for online hand gestures recognition, in addition, the proposed pipeline supports real-time and early detection gestures recognition. The performance of the pipeline over the state-of-the-art is demonstrated on the FPHA dataset. As future work, we intend to experiment with the proposed pipeline to other types of skeletal data in order to study applications such as human action and activity recognition. To do so, we plan to replace the initial grid convolution with a Graph

Neural Network layer in order to handle such more general skeletons.

VII. ACKNOWLEDGMENTS

The authors gratefully acknowledge the contribution of reviewers' comments. This material is based upon work supported by the European Union and the Region Normandie under the project IGIL.

REFERENCES

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, 1097-1105.

[2] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

[3] Pennec, X., Fillard, P., & Ayache, N. (2006). A Riemannian framework for tensor computing. International Journal of computer vision, 66(1), 41-66.

[4] Tuzel, O., Porikli, F., & Meer, P. (2006, May). Region covariance: A fast descriptor for detection and classification. In European conference on computer vision (pp. 589-600). Springer, Berlin, Heidelberg.

[5] Lee, B. G., & Lee, S. M. (2017). Smart wearable hand device for sign language interpretation system with sensors fusion. IEEE Sensors Journal, 18(3), 1224-1232.

[6] Kang, H., Lee, C. W., & Jung, K. (2004). Recognition-based gesture spotting in video games. Pattern Recognition Letters, 25(15), 1701-1714.

[7] Luo, R. C., & Wu, Y. C. (2012, September). Hand gesture recognition for human-robot interaction for service robot. In 2012 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI) (pp. 318-323). IEEE.

[8] Mambou, S., Krejcar, O., Maresova, P., Selamat, A., & Kuca, K. (2019). Novel hand gesture alert system. Applied Sciences, 9(16), 3419.

[9] Vemulapalli, R., Arrate, F., & Chellappa, R. (2014). Human action recognition by representing 3d skeletons as points in a lie group. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 588-595).

[10] Wang, J., Liu, Z., Wu, Y., & Yuan, J. (2012, June). Mining actionlet ensemble for action recognition with depth cameras. In 2012 IEEE Conference on Computer Vision and Pattern Recognition (pp. 1290-1297). IEEE.

[11] Zanfir, M., Leordeanu, M., & Sminchisescu, C. (2013). The moving pose: An efficient 3d kinematics descriptor for low-latency action recognition and detection. In Proceedings of the IEEE international conference on computer vision (pp. 2752-2759).

[12] Molchanov, P., Yang, X., Gupta, S., Kim, K., Tyree, S., & Kautz, J. (2016). Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4207-4215).

[13] Tran, D., Bourdev, L., Fergus, R., Torresani, L., & Paluri, M. (2015). Learning spatiotemporal features with 3d convolutional networks. In Proceedings of the IEEE international conference on computer vision (pp. 4489-4497).

[14] Graves, A., FernÃ¡ndez, S., Gomez, F., & Schmidhuber, J. (2006, June). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In Proceedings of the 23rd international conference on Machine learning (pp. 369-376).

[15] KÃ¶pÃ¼klÃ¼, O., Gunduz, A., Kose, N., & Rigoll, G. (2020). Online dynamic hand gesture recognition including efficiency analysis. IEEE Transactions on Biometrics, Behavior, and Identity Science, 2(2), 85-97.

[16] Montes, A., Salvador, A., Pascual, S., & Giro-i-Nieto, X. (2016). Temporal activity detection in untrimmed videos with recurrent neural networks. arXiv preprint arXiv:1608.08128.

[17] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[18] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. arXiv preprint arXiv:1409.3215.

[19] Cho, K., Van MerriÃ«nboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

[20] Maghoumi, M., & LaViola, J. J. (2019, October). DeepGRU: Deep gesture recognition utility. In International Symposium on Visual Computing (pp. 16-31). Springer, Cham.

[21] Avola, D., Bernardi, M., Cinque, L., Foresti, G. L., & Massaroni, C. (2018). Exploiting recurrent neural networks and leap motion controller for the recognition of sign language and semaphoric hand gestures. IEEE Transactions on Multimedia, 21(1), 234-245.

[22] Zhang, P., Lan, C., Xing, J., Zeng, W., Xue, J., & Zheng, N. (2017). View adaptive recurrent neural networks for high performance human action recognition from skeleton data. In Proceedings of the IEEE International Conference on Computer Vision (pp. 2117-2126).

[23] Zanfir, M., Leordeanu, M., & Sminchisescu, C. (2013). The moving pose: An efficient 3d kinematics descriptor for low-latency action recognition and detection. In Proceedings of the IEEE international conference on computer vision (pp. 2752-2759).

[24] Li, C., Cui, Z., Zheng, W., Xu, C., & Yang, J. Spatio-temporal graph convolution for skeleton based action recognition (2018).in AAAI, arXiv preprint arXiv:1802.09834.

[25] Yan, S., Xiong, Y., & Lin, D. (2018, April). Spatial temporal graph convolutional networks for skeleton-based action recognition. In Proceedings of the AAAI conference on artificial intelligence (Vol. 32, No. 1).

[26] Nunez, J. C., Cabido, R., Pantrigo, J. J., Montemayor, A. S., & Velez, J. F. (2018). Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition. Pattern Recognition, 76, 80-94.

[27] Luo, J., Wang, W., & Qi, H. (2013). Group sparsity and geometry constrained dictionary learning for action recognition from depth maps. In Proceedings of the IEEE international conference on computer vision (pp. 1809-1816).

[28] De Smedt, Q., Wannous, H., & Vandeborre, J. P. (2016). Skeleton-based dynamic hand gesture recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 1-9).

[29] Yang, X., & Tian, Y. L. (2012, June). Eigenjoints-based action recognition using naive-bayes-nearest-neighbor. In 2012 IEEE computer society conference on computer vision and pattern recognition workshops (pp. 14-19). IEEE.

[30] Evangelidis, G., Singh, G., & Horaud, R. (2014, August). Skeletal quads: Human action recognition using joint quadruples. In 2014 22nd International Conference on Pattern Recognition (pp. 4513-4518). IEEE.

[31] Vemulapalli, R., Arrate, F., & Chellappa, R. (2014). Human action recognition by representing 3d skeletons as points in a lie group. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 588-595).

[32] Wang, C., Wang, Y., & Yuille, A. L. (2013). An approach to pose-based action recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 915-922).

[33] Ofli, F., Chaudhry, R., Kurillo, G., Vidal, R., & Bajcsy, R. (2014). Sequence of the most informative joints (smij): A new representation for human skeletal action recognition. Journal of Visual Communication and Image Representation, 25(1), 24-38.

[34] Devineau, G., Moutarde, F., Xi, W., & Yang, J. (2018, May). Deep learning for hand gesture recognition on skeletal data. In 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018) (pp. 106-113). IEEE.

[35] Ke, Q., Bennamoun, M., An, S., Sohel, F., & Boussaid, F. (2017). A new representation of skeleton sequences for 3d action recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3288-3297).

[36] Liu, M., Liu, H., & Chen, C. (2017). Enhanced skeleton visualization for view invariant human action recognition. Pattern Recognition, 68, 346-362.

[37] Du, Y., Wang, W., & Wang, L. (2015). Hierarchical recurrent neural network for skeleton based action recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1110-1118).

[38] Liu, J., Wang, G., Hu, P., Duan, L. Y., & Kot, A. C. (2017). Global context-aware attention lstm networks for 3d action recognition. In

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1647-1656).

[39] Nunez, J. C., Cabido, R., Pantrigo, J. J., Montemayor, A. S., & Velez, J. F. (2018). Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition. Pattern Recognition, 76, 80-94.

[40] Bilinski, P., & Bremond, F. (2015, July). Video covariance matrix logarithm for human action recognition in videos. In IJCAI 2015-24th International Joint Conference on Artificial Intelligence (IJCAI).

[41] Guo, K., Ishwar, P., & Konrad, J. (2013). Action recognition from video using feature covariance matrices. IEEE Transactions on Image Processing, 22(6), 2479-2494.

[42] Sanin, A., Sanderson, C., Harandi, M. T., & Lovell, B. C. (2013, January). Spatio-temporal covariance descriptors for action and gesture recognition. In 2013 IEEE Workshop on applications of Computer Vision (WACV) (pp. 103-110). IEEE.

[43] Yuan, C., Hu, W., Li, X., Maybank, S., & Luo, G. (2009, September). Human action recognition under log-euclidean riemannian metric. In Asian conference on computer vision (pp. 343-353). Springer, Berlin, Heidelberg.

[44] Huang, Z., Wan, C., Probst, T., & Van Gool, L. (2017). Deep learning on lie groups for skeleton-based action recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 6099-6108).

[45] Huang, Z., & Van Gool, L. (2017, February). A riemannian network for spd matrix learning. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 31, No. 1).

[46] Huang, Z., Wu, J., & Van Gool, L. (2018, April). Building deep networks on grassmann manifolds. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32, No. 1).

[47] Ionescu, C., Vantzos, O., & Sminchisescu, C. (2015). Matrix backpropagation for deep networks with structured layers. In Proceedings of the IEEE International Conference on Computer Vision (pp. 2965-2973).

[48] Schindler, K., & Van Gool, L. (2008, June). Action snippets: How many frames does human action recognition require?. In 2008 IEEE Conference on Computer Vision and Pattern Recognition (pp. 1-8). IEEE.

[49] LovriÄ, M., Min-Oo, M., & Ruh, E. A. (2000). Multivariate normal distributions parametrized as a Riemannian symmetric space. Journal of Multivariate Analysis, 74(1), 36-48.

[50] Dong, Z., Jia, S., Zhang, C., Pei, M., & Wu, Y. (2017, February). Deep manifold learning of symmetric positive definite matrices with application to face recognition. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 31, No. 1).

[51] Engin, M., Wang, L., Zhou, L., & Liu, X. (2018). Deepkspd: Learning kernel-matrix-based spd representation for fine-grained image recognition. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 612-627).

[52] Tuzel, O., Porikli, F., & Meer, P. (2008). Pedestrian detection via classification on riemannian manifolds. IEEE transactions on pattern analysis and machine intelligence, 30(10), 1713-1727.

[53] Garcia-Hernando, G., Yuan, S., Baek, S., & Kim, T. K. (2018). First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 409-419).

[54] Fan, R. E., Chang, K. W., Hsieh, C. J., Wang, X. R., & Lin, C. J. (2008). LIBLINEAR: A library for large linear classification. the Journal of machine Learning research, 9, 1871-1874.

[55] Zhang, X., Wang, Y., Gou, M., Sznaier, M., & Camps, O. (2016). Efficient temporal sequence comparison and classification using gram matrix embeddings on a riemannian manifold. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4498-4507).

[56] Oreifej, O., & Liu, Z. (2013). Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 716-723).

[57] Hu, J. F., Zheng, W. S., Lai, J., & Zhang, J. (2015). Jointly learning heterogeneous features for RGB-D activity recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 5344-5352).

[58] Feichtenhofer, C., Pinz, A., & Zisserman, A. (2016). Convolutional two-stream network fusion for video action recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1933-1941).

[59] Rahmani, H., & Mian, A. (2016). 3d action recognition from novel viewpoints. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1506-1515).

[60] Zhu, W., Lan, C., Xing, J., Zeng, W., Li, Y., Shen, L., & Xie, X. (2016, March). Co-occurrence feature learning for skeleton based action recognition using regularized deep LSTM networks. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 30, No. 1).

[61] Garcia-Hernando, G., & Kim, T. K. (2017). Transition forests: Learning discriminative temporal transitions for action recognition and detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 432-440).

[62] Huang, Z., Wu, J., & Van Gool, L. (2018, April). Building deep networks on grassmann manifolds. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32, No. 1).

[63] Obinata, Y., & Yamamoto, T. (2020). Temporal Extension Module for Skeleton-Based Action Recognition. arXiv preprint arXiv:2003.08951.

[64] Nguyen, X. S., Brun, L., LÃ©zoray, O., & Bougleux, S. (2019). A neural network based on SPD manifold learning for skeleton-based hand gesture recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 12036-12045).

[65] Li, C., Li, S., Gao, Y., Zhang, X., & Li, W. (2021). A Two-stream Neural Network for Pose-based Hand Gesture Recognition. arXiv preprint arXiv:2101.08926.