applied sciences

MDPI

Article

Robust Hand Shape Features for Dynamic Hand Gesture Recognition Using Multi-Level Feature LSTM

Nhu-Tai Do, Soo-Hyung Kim *, Hyung-Jeong Yang and Guee-Sang Lee

Department of Artificial Intelligence Convergence, Chonnam National University, 77 Yongbong-ro, Gwangju 500-757, Korea; 176680@jnu.ac.kr (N.-T.D.); hjyang@jnu.ac.kr (H.-J.Y.); gslee@jnu.ac.kr (G.-S.L.)

* Correspondence: shkim@jnu.ac.kr

Received: 1 August 2020; Accepted: 7 September 2020; Published: 10 September 2020

Abstract: This study builds robust hand shape features from the two modalities of depth and skeletal data for the dynamic hand gesture recognition problem. For the hand skeleton shape approach, we use the movement, the rotations of the hand joints with respect to their neighbors, and the skeletal point-cloud to learn the 3D geometric transformation. For the hand depth shape approach, we use the feature representation from the hand component segmentation model. Finally, we propose a multi-level feature LSTM with Conv1D, the Conv2D pyramid, and the LSTM block to deal with the diversity of hand features. Therefore, we propose a novel method by exploiting robust skeletal point-cloud features from skeletal data, as well as depth shape features from the hand component segmentation model in order for the multi-level feature LSTM model to benefit from both. Our proposed method achieves the best result on the Dynamic Hand Gesture Recognition (DHG) dataset with 14 and 28 classes for both depth and skeletal data with accuracies of 96.07% and 94.40%, respectively.

Keywords: Dynamic Hand Gesture Recognition; human-computer interaction; hand shape features

1. Introduction

Besides the common language modalities, hand gestures are also often used in our daily lives to communicate with each other. For example, close friends can greet each other with a wave of their hands instead of words. Furthermore, hand gestures are the language of communication for deaf and mute people. In addition, hand gesture recognition is also one of the ways that computers can interact with humans by translating the human hand gestures into commands. Recently, hand gesture recognition research has developed rapidly, which is an essential element in the development of new technologies in the computer vision and pattern recognition fields. Especially, real-time 3D hand pose estimation combined with depth cameras has contributed to the successful launch of virtual reality and augmented reality applications such as sign language recognition [1], virtual reality [2], robotics [3], interaction systems [4], and interactive gaming [5]. Nevertheless, there exist various challenges hindering the achievement of accurate results due to the complex topology of the hand skeleton with high similarity among fingers and a small size. In addition, the cultural factors or personal habits of humans such as position, speed, and style can lead to variations in the hand gesture. Due to these special features, the hands can have various shapes describing the same pose. Feix et al. [6] found 17 different hand shapes that humans commonly use in everyday tasks to preform grasping.

Recent studies have suggested a number of solutions for challenges such as using reliable tools to capture 3D hand gestures and motion or using color gloves with attached sensors to capture real-time measurements of the hand [7,8]. However, their calibration setup process is complex and expensive.

In 2013, Shotton et al. [9] presented a concept called the "body skeleton" to accurately predict the 3D positions of 20 body joints from depth images. The author demonstrated that the position, movement, and orientation of the joints can be a great description of human action. As such, the hand skeleton can also process accurate information about the hand shape, and later, Potter et al. [10] proposed research on Australian Sign Language by using a reliable dataset of the labeled 3D hand skeleton corresponding to the 22 joints, which was provided by the Leap Motion Controller (LMC) device. Even so, the result was still inaccurate when the hand was near or perpendicular to the camera or when the person performed a quick gesture. In 2016, some 3D hand gesture datasets were proposed by [11], and Smedt et al. [12] gave a promising solution for performing gesture recognition tasks.

In addition to the dataset, the algorithm method also must meet the optimization needs for gesture recognition. Previously, traditional methods produced the feature descriptors in the spatial and temporal dimension to encode the statuses of hand motion and hand shape [13,14]. Currently, methods based on deep learning are considered solutions to recognize and classify images efficiently and reliably. Specifically, dynamic gesture recognition also applies deep learning such as [15,16]; however, they are limited in real-time execution.

As shown in Figure 1, the diversity of hand features improves the dynamic hand gesture recognition under the challenges of the complex topology of the hand and hand pose variations due to cultural factors and personal habits. The inputs of a gesture are the depth data and the hand skeleton perceived by a depth camera, as shown in the first row of Figure 1. We can focus on the global hand movement as the hand bounding box in red color and the hand posture in Row 2. We refer to the hand posture as the hand shape to highlight the local movements among hand components.

There are two kinds of hand shapes based on the input data skeleton or depth data, as in Row 3 and Row 4 in Figure 1, respectively. As the brightness constrains optical flow problems, the hand shapes are robust features because they not only focus on the local movements of the hand components, but also track the displacement of every element in the data such as the depth value in the depth data or the hand joint in the skeletal data between two consecutive times.

With our hypothesis that robust hand shape features impact the learning of local movements directly and global movements indirectly, our work explores the hand shape approach with the derivatives from the depth data and skeletal data. For the hand depth feature approach, we exploited hand component features from the hand segmentation model as shown in Row 5 in Figure 1. This can capture the shape changes of a gesture based on the hand component labels referring to the local movement in a hand gesture. For the hand skeleton features, we could exploit the point-cloud data from the depth data by 3D hand reconstruction; however, due to the time constraints in real-time hand applications, we focused on the 3D point-cloud under the hand joint coordinates in the 3D world space from the skeletal data. Our model will be able to learn 3D geometric transformation features. Simultaneously, we also use the displacement and rotation of the hand joints with respect to their neighbors for the hand skeletal shape features. Therefore, our model of dynamic hand gesture recognition addresses the diversity problem in hand features from the two modalities of the hand depth and hand skeletal data.

To benefit from the various features, we propose multi-level feature LSTM using Conv1D, the Conv2D pyramid, and the LSTM block to exploit the diversity of the hand features from handcrafted data to automatically generate data from the deep learning model for the time series data and depth data. Our proposed method achieves the best accuracy on Dynamic Hand Gesture Recognition (DHG) [12] on 14 and 28 classes, respectively, with the skeletal data. It is good for real-time applications requiring low processing costs.

Our paper's contribution consists of three parts. Firstly, we identify various hand features from two modalities with depth and skeletal data. We then propose the best features for exploiting skeletal data and depth data to achieve the best results. Secondly, we build the multi-level feature LSTM with Conv1D, the Conv2D pyramid, and the LSTM block to use the effective hand features. Finally, we experimented on DHG 14 and 28 and obtained the best results.

The rest of this paper is composed of the following sections: Related Work, Proposed Idea, Hand Posture Feature Extraction, Network Architecture, Experiment, and Discussion and Conclusion. In Related Work, we discuss the datasets and approaches of 3D hand gestures. In the next two parts, our proposed method is described in detail. We then analyze the strengths of our method and make comparisons with the state-of-the-art in the experiments and discussion. The conclusions are given in the final part.

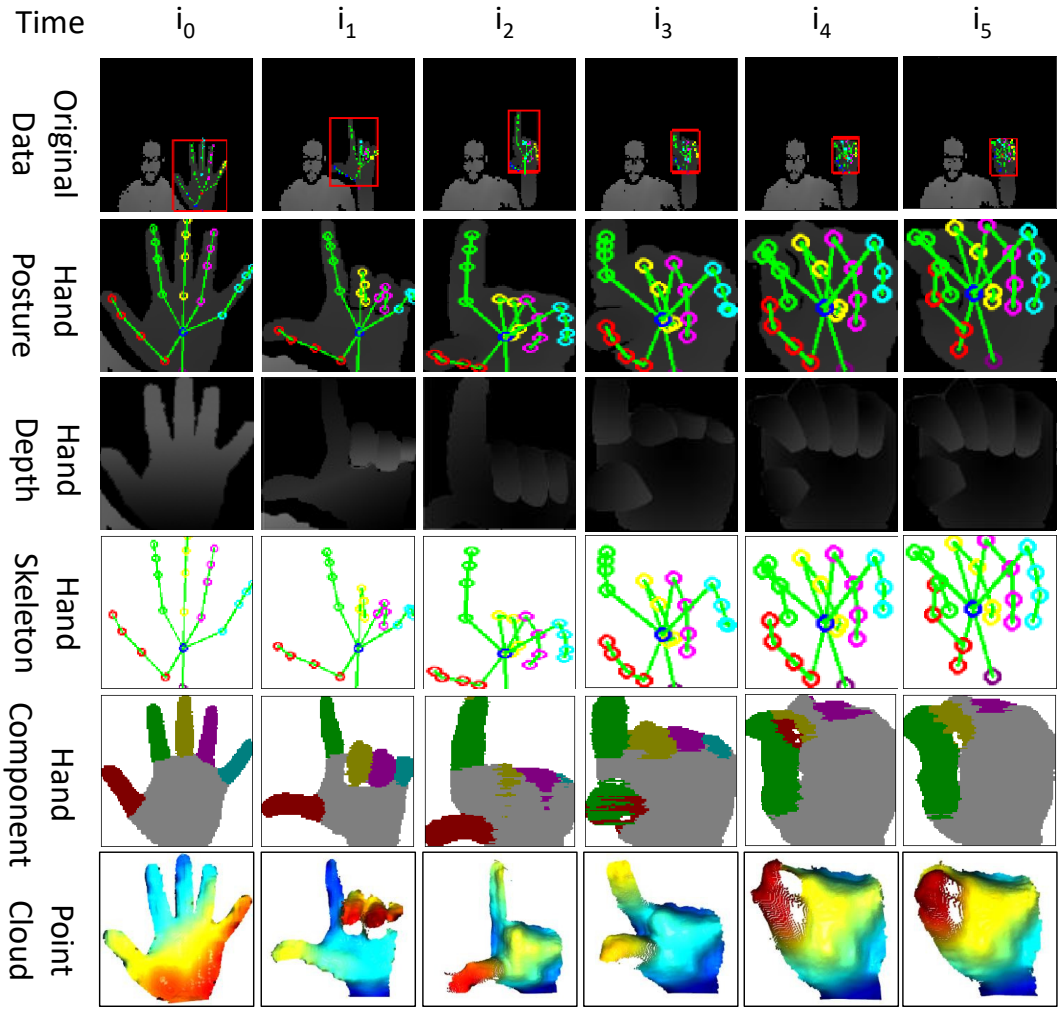


Figure 1. Overview of the features of a dynamic hand gesture. Left to right shows the time axis of the gesture, and top to bottom shows the types of hand data features, consisting of the original data, hand posture, hand depth, hand skeleton, hand component, and hand point-cloud.

2. Related Works

Hand gesture recognition research has robustly developed with a variety of approaches in recent years. The advancement in 3D depth sensors with a low cost has been one of the key elements that has increased the research into 3D hand gestures. With this technology, light variations, background clutter, and occlusions are major concerns in the detection and segmentation of hands. Furthermore, the depth sensors can capture 3D information in the scene context, which helps give faster estimation of the hand skeleton from the hand pose. Hence, there is much information to recognize hand gestures such as the hand skeleton, depth, and color images [17]. Below are the main categories of the approaches to 3D hand gesture recognition: static and dynamic hand gesture recognition or hand gesture recognition using deep images and/or hand skeletal data.

2.1. Dynamic Hand Gesture Recognition

The first approach is to identify static hand gestures. The 3D depth information and various traditional methods are utilized to detect hand shadows and hand regions used to extract features. Namely, Kuznetsova et al. [18] used the hand point cloud to compute invariant features, then they applied a multi-layered random forest for training to recognize hand signs. In the same way, Pugeault et al. [19] combined the Gabor filter and random forest for gesture classification to detect hand shape for the American Sign Language (ASL) finger-spelling system. Dong et al. [20] suggested a hierarchical mode-seeking method to localize hand joint positions under kinematic constraints and applied random forest to classify ASL signs based on joint angles. Ohn-Bar et al. [14] leveraged a modified HOG algorithm, named the spatial-temporal HOG2 descriptor, to extract useful information from spatial-temporal descriptors.

Ren et al. [21] expressed the hand shape as a time-series curve to facilitate the classification and clustering of shapes without using HOG, SIFT, and random forest. Furthermore, they proposed a new distance metric named the finger-earth mover's distance to discriminate hand gestures. Recently, Zhang obtained remarkable results using his proposed histogram of 3D facets to encode 3D hand shape information from the depth map [13]. Furthermore, Oreifej et al. [22] built a histogram of the normal orientations' distribution by integrating the time, depth, and spatial coordinates into 4D space to recognize the activity from depth sequences.

If the static approach handles the hand region and extracts hand features from a single image, the dynamic methods deem hand gesture recognition as recognizing a sequence of hand shapes by exploiting the temporal features of motion. Zhang et al. solved the gesture recognition problems by linking the histogram of 3D facets, the N-gram model, and dynamic programming on depth maps [13]. On the other hand, Monnier et al. [23] used a boosted classifier cascade to detect the gesture. They also leveraged body skeletal data and the histogram of oriented gradients to obtain the features.

Recently, the significant progress of Convolutional Neural Networks (CNNs) has led to various groundbreaking studies in the computer vision field, and hand gesture recognition in particular, such as image classification [24], object detection [25], and image segmentation [26]. Aghbolaghi et al. [27] performed a survey to demonstrate the effectiveness of deep learning approaches in action and gesture recognition. In [28], a factorized spatial-temporal convolutional network, which is a cascaded deep architecture, learned spatio-temporal features and transferred learning from the pre-trained ImageNet on a 2D CNN, while Varol et al. [29] used a neural network having long-term temporal convolutions to compute motion features for temporal information. In order to study real-time hand gesture recognition, Neverova et al. [30] proposed a method that combines both video data and articulated pose with multi-scale and multi-modal deep learning. Similarly, Molchanov et al. [16] applied multi-scale 3D-CNN models with depth, color, and stereo-IR sensor data.

2.2. Depth and 3D Skeleton Dynamic Hand Gesture Recognition

In recent works, along with the advances in hand pose estimation and the technology of depth-based cameras, skeleton-based recognition has gained more traction. In [11], they extracted the features of the distances, angles, elevations, and adjacent angles of fingertips by employing the data of direction, normal position, the central location of the palm, and fingertip position. Garcia et al. built the mo-capsystem to make hand pose annotations and gather 6D object poses from RGB-D video data for hand recognition [31]. De Smedt et al. [12] published an approach with results better than the results of depth-based methods. They calculated the shape of connected joints descriptor from the connected joints of the hand skeleton and used a Fisher vector to encode it. A temporary pyramid was used to model Fisher vectors and skeleton-based geometric features before extracting the final feature.

There are various methods that are based on deep learning, for dynamic hand gesture recognition using skeleton-like information. Chen et al. [32] performed training on a bidirectional Recurrent Neural Network (RNN) using the movement features of fingers and hand and skeleton sequences. Devineau et al. proposed parallel convolutions to handle sequences of the hand skeleton.

De Smedt [33] proposed a new way based on CNN and LSTM by fusing two streams of the hand shape and skeleton model.

Collectively, all the above methods have problems in recognizing gestures at some distance from the camera, with variable illumination.

3. Proposed Idea

3.1. Problem Definition

A dynamic hand gesture $\boldsymbol{G} = (\boldsymbol{S}, \boldsymbol{D})$ in this study can be described as a time-series stream of the 3D hand skeletal data $\boldsymbol{S}$ and hand depth data $\boldsymbol{D}$ with the length N_t frames. The goal of our method is based on $\boldsymbol{S}$ and $\boldsymbol{D}$ to classify whether a dynamic hand gesture belongs to one of the given gesture types $C = \{c_1, c_2, ..., c_{N_c}\}$. The gesture types are determined based on the specific dataset.

Let $\boldsymbol{x}_j^t \triangleq \left(x_j^t, y_j^t, z_j^t\right)$ be the world space coordinate of the 3D hand joint j at time t; the hand skeleton posture $S_t \in \mathbb{R}^{N_j \times 3}$ at time t is the set J of hand joints with N_j coordinates in 3D space defined as follows:

$$\boldsymbol{S}_t = \left\{\boldsymbol{x}_j^t\right\}_{j=1..N_j}^{t} \tag{1}$$

The 3D hand skeletal data $S \in \mathbb{R}^{N_t \times N_j \times 3}$ are expressed as the set of hand skeleton postures S_t as below:

$$\boldsymbol{S} = \{\boldsymbol{S}_t\}^{t=1..N_t} \tag{2}$$

In the case of the hand depth data, we let $D_t \in \mathbb{R}^{w \times h}$ be the depth image at time t with the width w and height h. The 3D hand depth data $D \in \mathbb{R}^{N_t \times w \times h}$ are represented by the set of hand depth postures D_t as follows:

$$\boldsymbol{D} = \left\{\boldsymbol{D}_t \in \mathbb{R}^{w \times h}\right\}^{t=1..N_t} \tag{3}$$

3.2. Problem Overview

The overview of our proposed pipeline is as shown in Figure 2. We first apply the temporal frame sub-sampling for every dynamic gesture G to the specific length N_t. Then, we split the dynamic gesture into hand skeletal data S and hand depth data D as the input to the feature extraction step.

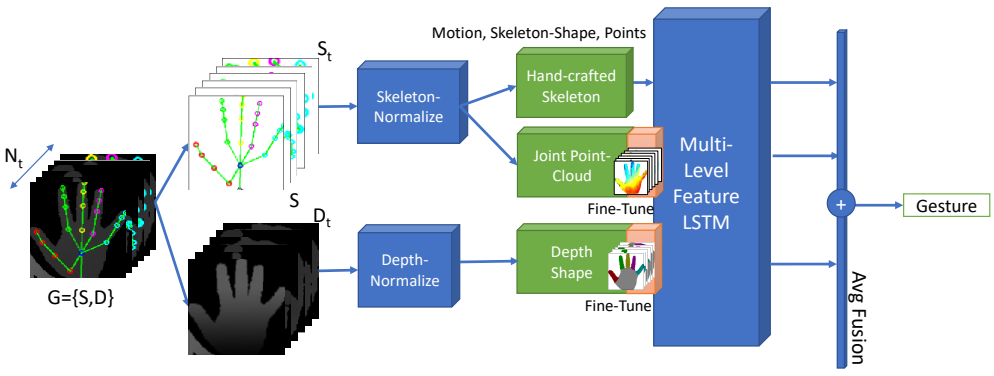


Figure 2. Overview of the proposed system for dynamic hand gesture recognition.

In the normalize phase, with hand skeletal data, to deal with the variants in the size and pose (translation and rotation) of the gestures due to the changes in the performer and the camera pose, we need to normalize the 3D hand joint coordinates of all frames by the same transformation of rotation and translation and uniformly scale from the first hand posture in the gesture to the given reference hand posture in front of the camera with the palm joint at the root coordinate. With the hand depth data, we apply image processing techniques to eliminate the isolated depth pixels in the hand region and convert the depth value to the range $[0, 1]$.

In our proposed framework, the feature extraction phase consists of five feature types to exploit the robust features for the dynamic hand gesture as follows:

Three first three feature types are the handcrafted skeleton features including the motion, skeleton shape, and normalized hand skeletal coordinates. The motion feature captures the changes of the translation and rotation of the overall hand. A new contribution of this study is using the pairwise joint distance instead of using the Shape of Connected Joints (SoCJ) descriptor as in [34] to reduce the complex feature space. Moreover, we add more than one feature with the oriented angle of the three joint tuples to represent the hand skeletal shape with the oriented values among the hand joints.

The next two feature types are the joint point-cloud feature and the hand depth shape feature automatically extracted by the deep learning models in an end-to-end fashion. With the recent success of the hand pose regression problem in using 3D reconstruction from depth data to the point-cloud and voxel [35,36], we opted to use Point-Net [37] to learn the 3D geometric features. Instead of using all 3D points in the hand depth data reconstructed from the point-cloud, we propose to only use the 3D world space hand joints as the key points for Point-Net. Regarding the hand depth shape feature, we propose to use the middle layer of the encoder-decoder hand segmentation model as the hand depth shape feature.

Finally, we propose the multi-level feature LSTM model to train on every hand gesture feature. Our architecture firstly uses the LSTM filter layer to exploit the long-term dependencies between the frames and reduce the complexity of the input feature. After the first LSTM layer, we use the self-attention mechanism and three kinds of blocks, namely, Con1D, Conv2D, and LSTM, to exploit the spatial and temporal coherency in the feature spaces. The LSTM filter layer will send the encode states to the feature LSTM layer to help the second LSTM learn better.

Note that for hand gesture feature extracting from deep learning models, they will be integrated into the hand gesture recognition model to fine-tune again during the training phase of the dynamic gesture recognition model.

Finally, we use the average pooling layer to integrate the classification probability for all separate models for every hand gesture feature. Our result will classify the type of gestures.

3.3. Hand Skeleton Normalization

The hand skeletal data are received from various camera sensors, the pose of the camera, as well as the performer. Therefore, we need to normalize the data to the reference pose and size to prevent over-fitting of the method with respect to the environmental elements.

First, we choose the reference hand $\boldsymbol{S}_{ref} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_{N_j}]^T$ in the dataset with the status of open and in front of the camera, as in Figure 3. Then, we transform the reference hand so that the palm joint is at the root coordinate and scale the hand size to fit in the unit sphere as follows:

$$\boldsymbol{S}_{norm} = \boldsymbol{S}_{ref} - \boldsymbol{x}_{palm} \tag{4}$$

$$scale = \max \|\boldsymbol{x}_i\|_2 \text{, where } \boldsymbol{x}_i \in \boldsymbol{S}_{norm} \tag{5}$$

$$\boldsymbol{S}_{norm} = \frac{\boldsymbol{S}_{norm}}{scale} \tag{6}$$

For every skeletal data sequence, we find the optimal rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$ using [38] between the hand skeletal data at the first frame $\boldsymbol{S}_{t_1}$ and the reference hand $\boldsymbol{S}_{norm}$ based on the seven hand joints corresponding to the 3D points (palm, wrist, from the thumb to pinky base joints) as the least squares error minimization problem:

$$E = \sum_{i=1}^{N_b} \left\| \boldsymbol{R}\boldsymbol{x}_i^{t_1} + \boldsymbol{t} - \boldsymbol{x}_i^{norm} \right\|_2 \longrightarrow 0 \tag{7}$$

where $\boldsymbol{x}_i^{t_1} \in$ base joints $(\boldsymbol{S}_{t_1})$, $\boldsymbol{x}_i^{norm} \in$ base joints $(\boldsymbol{S}_{norm})$, and $N_b = 7$ is the number of base joints (palm, wrist, from the thumb to pinky base joints).

To solve Equation (7), we find the center of base joints (S_{t_1}), base joints (S_{norm}), respectively:

$$x_{center}^{t_1} = \frac{1}{n_{t_1}} \sum_{i=1}^{n_{t_1}} x_i^{t_1} \tag{8}$$

$$x_{center}^{norm} = \frac{1}{n_{norm}} \sum_{i=1}^{n_{norm}} x_i^{norm} \tag{9}$$

where $n_{t_1} = |\text{base joints}\,(S_{t_1})|$, and $x_i^{t_1} \in \text{base joints}\,(S_{t_1})$; similarly, $n_{norm} = |\text{base joints}\,(S_{norm})|$, and $x_i^{norm} \in \text{base joints}\,(S_{norm})$. Then, we calculate the Singular Value Decomposition (SVD) of the co-variance matrix $\boldsymbol{H}$ to find the rotation transformation $\boldsymbol{R}$ as follows:

$$\boldsymbol{H} = \left(\text{base joints}\,(S_{t_1}) - x_{center}^{t_1}\right)\left(\text{base joints}\,(S_{norm}) - x_{center}^{norm}\right)^T \tag{10}$$

$$\boldsymbol{U}, \boldsymbol{S}, \boldsymbol{V} = SVD\,(\boldsymbol{H}) \tag{11}$$

$$\boldsymbol{R} = \boldsymbol{V}\boldsymbol{U}^T \tag{12}$$

We address the reflection case of the SVD results by checking the determinant $|R| < 0$ and fixing again as the equation below:

$$\boldsymbol{U}, \boldsymbol{S}, \boldsymbol{V} = SVD\,(\boldsymbol{R}) \tag{13}$$

$$\boldsymbol{V}\,[column3] = -1 * \boldsymbol{V}\,[column3] \tag{14}$$

$$\boldsymbol{R} = \boldsymbol{V}\boldsymbol{U}^T \tag{15}$$

Finally, the translation is calculated as below:

$$\boldsymbol{t} = x_{center}^{norm} - \boldsymbol{R} \times x_{center}^{t_1} \tag{16}$$

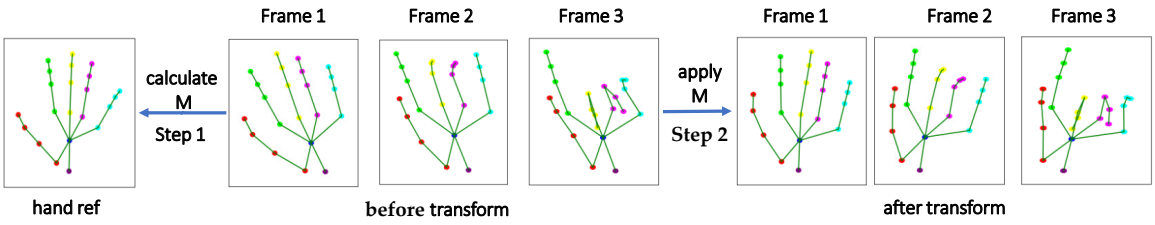


Figure 3. Hand skeleton normalization. We will calculate the rotation, translation, and uniform scale of the first hand skeleton to the reference hand skeleton. Then, the matrix transform will be applied to the remaining hand skeletons.

3.4. Hand Depth Normalization

Given that $\boldsymbol{H_t} = (x_t, y_t, w_t, h_t)$ is the bounding box of the hand region at time t, the hand skeleton data $\boldsymbol{D_t}$ are extracted from the depth data $\boldsymbol{I_t}$ by $\boldsymbol{I_t}\,(\boldsymbol{H_t})$. There are many background and noisy pixels, which are often the isolated depth pixels in the hand depth data $\boldsymbol{D_t}$. The morphology operator in image processing is used to eliminate them.

For the background pixel elimination problem, the depth values of pixels in the hand region gather around the centroid $M_{centroid}$ of the hand depth values. Therefore, the depth threshold t_{depth} is used to remove the background pixels as follows:

$$\boldsymbol{D}_t\,(x,y) = \begin{cases} \boldsymbol{D}_t\,(x,y) & |\boldsymbol{D}_t\,(x,y) - M_{centroid}| < t_{depth} \\ 0 & otherwise \end{cases} \tag{17}$$

where the centroid $M_{centroid} = Mode\,(\boldsymbol{D}_t)$. All depth values after removing isolated and background pixels are normalized to [0, 1].

4. Hand Posture Feature Extraction

A dynamic hand gesture often consists of two components: motion and shape. Motion components contain the changing information with respect to time of the overall hand for global motions and the fingertip positions for local motions. The shape components represent the hand shape at the specific time by the hand joint positions and hand components (palm, wrist, fingers at the base, middle or tip regions) in the corresponding domains (skeleton or depth).

In this study, we only calculate the global motions of the overall hand. The local motions can be exploited from the hand shape changes with respect to time. Therefore, the hand shape feature models sometimes archive better performance than the motion models due to capturing the local motions from the shape changes.

Furthermore, there are three ways to divide the types of hand posture features. The first group is based on skeleton and depth data. The second group is based on the means of feature extraction, such as the handcrafted features and deep learning features. The last group is the components of the gesture, such as motion and shape.

In this section, we mention three main groups: handcrafted skeleton features (motion, skeleton shape, and normalized points), joint point-cloud feature (input from normalized points to exploit 3D geometric characteristic), and depth shape feature (input from the depth data to determine the hand components).

4.1. Handcrafted Skeleton Features

4.1.1. Motion Feature Extraction

We represent the global motion S_{motion} of the specific hand by the changes of the palm coordinate S_{dir}, the angle between the palm and wrist joint S_{rot}, and the major axis of all hand joints S_{maj}:

$$S_{motion} = \{S_{dir}, S_{rot}, S_{maj}\} \tag{18}$$

The translations of the hand that determine S_{dir} by the direction of the two palm joints at two consecutive times t_i and t_{i+1} are calculated as below:

$$S_{dir}^t = \frac{x_{palm}^t - x_{palm}^{t-1}}{\left\| x_{palm}^t - x_{palm}^{t-1} \right\|_2} \tag{19}$$

$$S_{dir} = \{S_{dir}^t\} \tag{20}$$

The rotation of the hand represents the sign of the angles between the wrist and palm joints using the dot product operator as below:

$$S_{rot}^t = \frac{x_{wrist}^t . x_{palm}^t}{\left\| x_{wrist}^t \right\| \left\| x_{palm}^t \right\|} \tag{21}$$

$$S_{rot} = \{S_{rot}^t\} \tag{22}$$

Furthermore, we propose to use the changes in the major axis of all hand joints to more precisely express the orientation of all hand joints. The major and minor axes of the hand joints correspond to the eigenvectors of the covariance matrix of the set of hand joints:

$$Cov\left(S^t\right) = \frac{1}{N_j} \sum_{i=1}^{N_j} \left(x_i^t - \bar{x}_i^t\right) \left(x_i^t - \bar{x}_i^t\right)^T \tag{23}$$

$$U^t, S^t, V^t = SVD\left(Cov\left(S^t\right)\right) \tag{24}$$

$$U^t = \left[v_1^t, v_2^t, v_3^t\right] \tag{25}$$

where $\bar{x}_i^t$ is the center of the 3D hand joint coordinates S^t and $\{v_i\}$ is the eigenvectors forming the orthogonal basis, as well as the major axes. Hence, the major axis feature S_{maj} is expressed as:

$$S_{maj}^t = \{v_i^t\}_{i=1}^3 \tag{26}$$

$$S_{maj} = \left\{S_{maj}^t\right\} \tag{27}$$

4.1.2. Hand Skeleton Shape Extraction

The role of hand shape in the dynamic hand gesture recognition determines the movement of the joints and components of the local motion. Therefore, we represent the hand skeleton shape S_{kshape} with two components: the movement of the joints with respect to their neighbors S_{kmov} and the angle of the joints with respect to two neighboring joints S_{krot}, as shown in Figure 4:

$$S_{kshape} = \{S_{kmov}, S_{krot}\} \tag{28}$$

Regarding the shape descriptor of the movement of a hand joint with respect to its neighbors, we use all displacements at a point with respect to the remaining points with no overlap between the two points as follows:

$$S_{kmov}^t = \left\{x_j^t - x_i^t \mid i < j \text{ and } i, j \in [1, N_j]\right\} \tag{29}$$

$$S_{kmov} = \{S_{kmov}^t\} \tag{30}$$

In this study, with $N_j = 22$, there are in total $C_{N_j}^2 = 231$ elements in S_{kmov}^t. Besides the local movement features, we also suggest the angles between one joint and two distinct joints as the features to exploit the local rotation in the dynamic gesture. This is calculated as:

$$S_{krot}^t = \left\{\frac{\left(x_j^t - x_i^t\right) . \left(x_k^t - x_j^t\right)}{\left\|\left(x_j^t - x_i^t\right)\right\| \left\|\left(x_k^t - x_j^t\right)\right\|} \mid i < j < k \text{ and } i, j, k \in [1, N_j]\right\} \tag{31}$$

$$S_{krot} = \{S_{krot}^t\} \tag{32}$$

Similarly, the number of angle features at time t is $C_{N_j=22}^3 = 1540$.

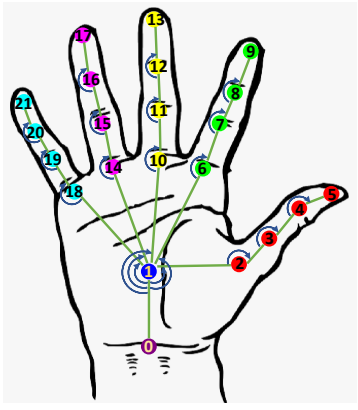


Figure 4. Hand skeleton shape calculated by the movement and rotation of joints with respect to their neighbors. There are 22 joints in the hand skeleton data numbered from 0–21: 0 (wrist), 1 (palm), 2–5 (thumb), 6–9 (index), 10–13 (middle), 14–17 (ring), and 18–21 (pinky).

4.1.3. Normalized Points

Finally, we directly use the hand joint coordinates at time t normalized S^t_{points} to the classification model for gesture recognition as below:

$$S^t_{points} = \{x^t_i\} \tag{33}$$

$$S_{points} = \left\{S^t_{points}\right\} \tag{34}$$

4.2. Joint Point-Cloud Feature Model

With the success of deep learning networks, various computer vision tasks can extract the features and automatically classify the object in an end-to-end fashion. Therefore, we aimed to build our feature models to exploit the 3D geometric transformation features from the point-cloud and the visual features from the depth data.

The joint point-cloud feature model facilitates the learning of the 3D geometric transformation features from the 3D point-cloud. In the hand gesture data, there are two ways to construct the point-cloud. Firstly, we can reconstruct the hand depth data into a set of 3D points. This approach is hindered by the unordered attributes of the point-cloud, the alignment between the points at different times, and the processing cost to convert and process. Secondly, the hand joint points of the gesture can represent the point-cloud. This has the advantages of the order of the set and the alignment of the joints by time.

Due to the low resolution of the skeletal point-cloud, we chose PointNet [37], as shown in Figure 5, to learn deep features in the 3D geometric transform in an end-to-end fashion.

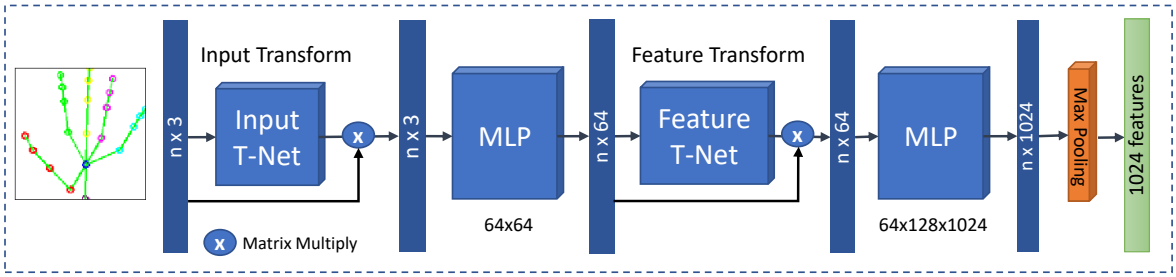


Figure 5. Point-Netarchitecture [37].

Therefore, the joint point-cloud feature model f_{point_cloud} is expressed as:

$$S^t_{point\_cloud} = f_{point\_cloud}\left(S^t_i\right) \tag{35}$$

$$S_{point\_cloud} = \left\{S^t_{point\_cloud}\right\} \tag{36}$$

where $S^t_{point\_cloud}$ is the point-cloud feature at time t from f_{point_cloud}.

Point-Net is comprised of the transform blocks, MLP blocks, and one max-pooling layer. The transform blocks can be represented as a function f to map a point set (input T-Net) or a feature point set (feature T-Net) to a feature vector with permutation invariance and geometric transformations as follow:

$$f^{trans}_{point\_cloud}\left(S^t_i\right) = \gamma\left(\max\left\{h\left(S^t_i\right)\right\}\right) \tag{37}$$

where x^t_i is the point-cloud, h is the MLP block to capture the feature of the point set, γ is the symmetric function as an appropriate rigid or affine transformation with a 3×3 matrix transform to achieve the normalization, and the operator max selects highly activated values from the point features. After every transform block, there are MLP blocks to learn and extend the feature size. Finally, the max-pooling layer will return the feature values of the point-cloud.

In this study, the joint point-cloud model was integrated as a child block of the dynamic hand gesture recognition model to learn point features from scratch while training the gesture classification.

4.3. Depth-Shape Feature Model

The depth shape feature model f_{depth_shape} plays the role of the feature extraction block to obtain the feature vector that presents the hand shape in the depth data. The depth shape feature model in our study is based on the U-Net architecture [39] to learn the hand components from segmenting hand regions from depth data. It can be expressed as:

$$\boldsymbol{D}^t_{depth\_shape} = f^e_{depth\_shape}\left(\boldsymbol{D}^t\right) \tag{38}$$

$$\boldsymbol{H}^t_{mask} = f^d_{depth\_shape}\left(\boldsymbol{D}^t_{depth\_shape}\right) \tag{39}$$

$$\boldsymbol{D}_{depth\_shape} = \left\{\boldsymbol{D}^t_{depth\_shape}\right\} \tag{40}$$

or:

$$\boldsymbol{H}^t_{mask} = f^d_{depth\_shape}\left(f^e_{depth\_shape}\left(\boldsymbol{D}^t\right)\right) \tag{41}$$

where $f^e_{depth\_shape}$ is the encoder function to encode the depth data as the depth shape feature $\boldsymbol{D}^t_{depth\_shape}$ at time t while $f^d_{depth\_shape}$ is the decoder function to map the encoder feature to the hand component masks $\boldsymbol{H}^t_{mask}$ by one-hot encoding.

U-Net, as shown in Figure 6, consists of the encoder and decoder blocks and the skip connections between them. The structure of the encoder can be based on the common visual image classification models such as VGG16 [40], Resnet50 [41], etc. The backbone of the encoder using VGG16, as shown in Figure 6, has five blocks of two convolution layers, batch-normalization, and max-pooling. The encoder block converts the depth data input into the encoded features, then the decoder blocks convert the encoded features into the semantic representation of the input data with the hand component masks in the background, palm, thumb, index, middle, ring, and finger regions. The role of the skip connections helps our model to be trained stably and achieve a better result by passing the features from the encoder to the decoder at the same levels concurrently with the features from the decoders below. Through the skip connection, the model can keep learning even when the deeper encoder and layer cannot learn by the dying ReLU problem or the vanishing problem.

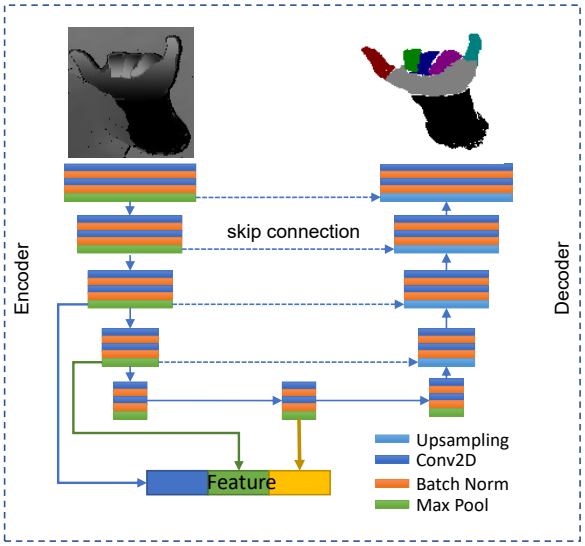


Figure 6. Hand component segmentation model to extract depth shape features.

The middle block between the encoder and decoder blocks is the feature block to return the feature vector. To enhance the feature vector, the hierarchical depth shape feature combines Blocks 3 and 4 of the encoder, as shown in Figure 6.

For the dataset for training the depth shape, it should focus on the hand components instead of only the hand pose due to the complex structure of the hand such as the small size and self-occlusions. In this study, Finger-Paint [42] is a suitable dataset with all the requirements.

For the loss function, the soft Dice loss [43] is a good choice in the unbalanced cases among the segmentation regions. Since the palm region often has a larger size compared to finger regions in the hand components, the loss function measures the overlap between the two regions the object region and the non-object region, in the binary classification. In the multi-class classification problem, the final score will be averaged with the Dice loss of each class expressed as:

$$L\left(y_{true}, y_{pred}\right) = 1 - \frac{1}{N_c}\sum_{c=1}^{N_c}\frac{2\sum_{pixels} y_{true}^{c} y_{pred}^{c} + \epsilon}{\sum_{pixels}\left(y_{true}^{c}\right)^2 + \sum_{pixels}\left(y_{pred}^{c}\right)^2 + \epsilon} \tag{42}$$

where c is the region of the hand components including the N_c regions (background, palm, thumb, index, middle, ring, pinky); y_{true}^{c} and y_{pred}^{c} are the ground-truth and the prediction of the hand component masks, respectively, in region c.

5. Our Network Architecture

From the hand feature extraction step, the system receives handcrafted skeleton features ($\boldsymbol{S}_{motion}$, $\boldsymbol{S}_{skeleton\_shape}$, and $\boldsymbol{S}_{points}$), the joint point-cloud feature $\boldsymbol{S}_{point\_cloud}$, and the depth shape feature $\boldsymbol{D}_{depth\_shape}$. In general, let $\chi\left\{\boldsymbol{G}^t\right\}$ be a feature transform of a gesture $\boldsymbol{G}^t$ at time t using one of the feature extractions mentioned. Our proposed model shown in Figure 7 uses the first LSTM layer [44] for a time series of features extracted from gesture data to exploit the long-term dependencies and encode them into a sequence the same as the length of the input gesture with the specific feature size. The encoder LSTM layer can be expressed as follows:

$$\boldsymbol{h}_1^t, \boldsymbol{c}_1^t = LSTMCell\left(\boldsymbol{G}^t, \boldsymbol{h}_1^{t-1}, \boldsymbol{c}_1^{t-1}\right) \tag{43}$$

$$\boldsymbol{h}_1 = \left\{\boldsymbol{h}_1^t\right\}, \boldsymbol{c}_1 = \left\{\boldsymbol{c}_1^t\right\} = LSTM\left(\boldsymbol{G}\right) \tag{44}$$

where $\boldsymbol{h}_1$ and $\boldsymbol{c}_1$ are the set of hidden state $\boldsymbol{h}_1^t$ and cell state $\boldsymbol{c}_1^t$ at time t. If the feature transform χ is the deep learning feature model such as $\boldsymbol{S}_{point\_cloud}$ and $\boldsymbol{D}_{depth\_shape}$, the proposed model can be straightforward to fine-tune again the feature model integrated in it.

$\boldsymbol{h}_1$ plays the role of the normalized encoding features containing long-term dependencies, and $\boldsymbol{c}_1$ is the cell state in the LSTM used to transfer to the next layer the states for the other LSTM or as an attention vector.

The encoding feature vector $\boldsymbol{h}_1$ gives the Con1D pyramid block, the Conv2D pyramid block, as well as the LSTM block to exploit the temporal-spatial coherency, as shown in Figure 8.

The Conv1D pyramid block contains the Conv1D blocks with every block consisting of the Conv1D layers and the dropout layer with different filter sizes. Global average pooling is in the last position to capture the feature vector by exploiting $\boldsymbol{h}_1$ on its feature axis.

The Conv2D pyramid block consists of the same Conv2D blocks as the VGG16 block, which contain the Conv2D layers, dropouts, and max-pooling at the end of the block. It will exploit features in the time-step and feature axis of the input, select the high values by max-pooling, and down-sample the input on the time-step axis. Finally, the global average pooling layer will compress and return the feature vector.

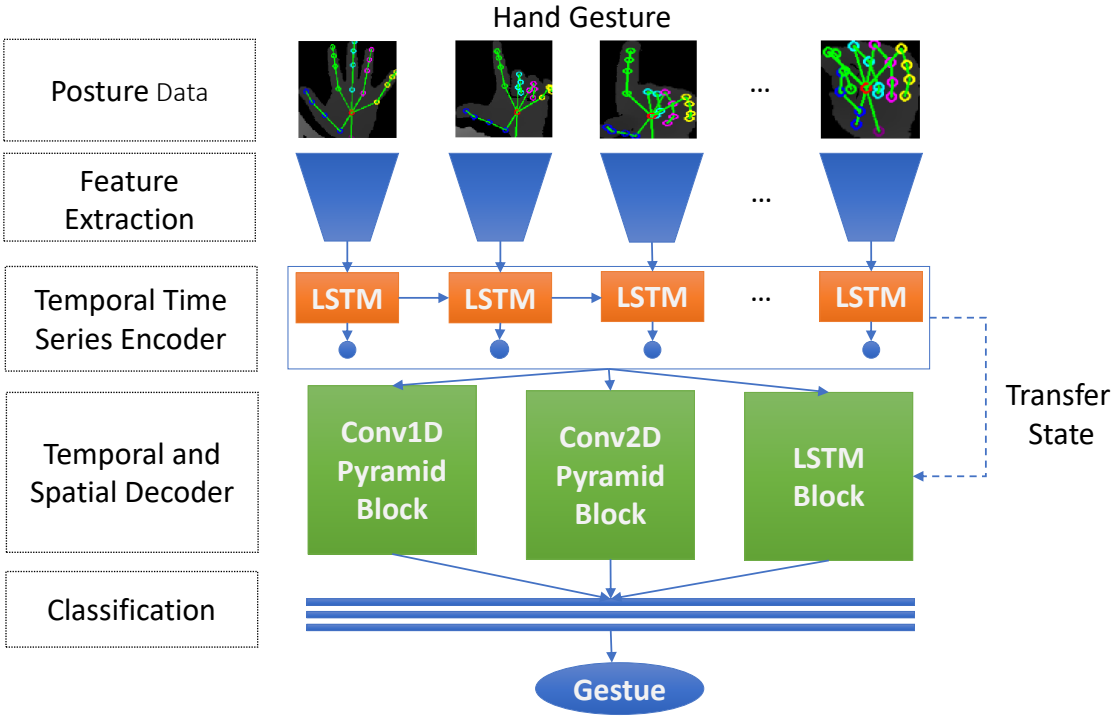


Figure 7. Multi-level feature LSTM architecture.

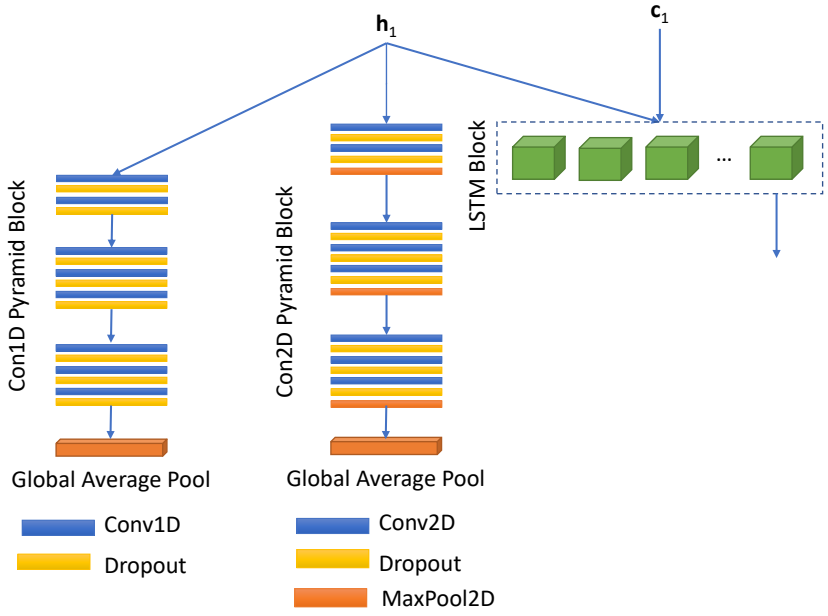


Figure 8. The structure of the LSTM block, Conv1D, and the Conv2D pyramid block.

Unlike Conv1D and the Conv2pyramid block, the input of the LSTM model from h_1 and c_1 of the previous LSTM layer c_1 will help the model learn h_1 better by inheriting the cell state from the previous LSTM layer.

Finally, all features are concatenated from the building blocks and added to the dense layers to classify the gestures.

For the loss function, we use the Category Cross-Entropy (CCE) for classification expressed as:

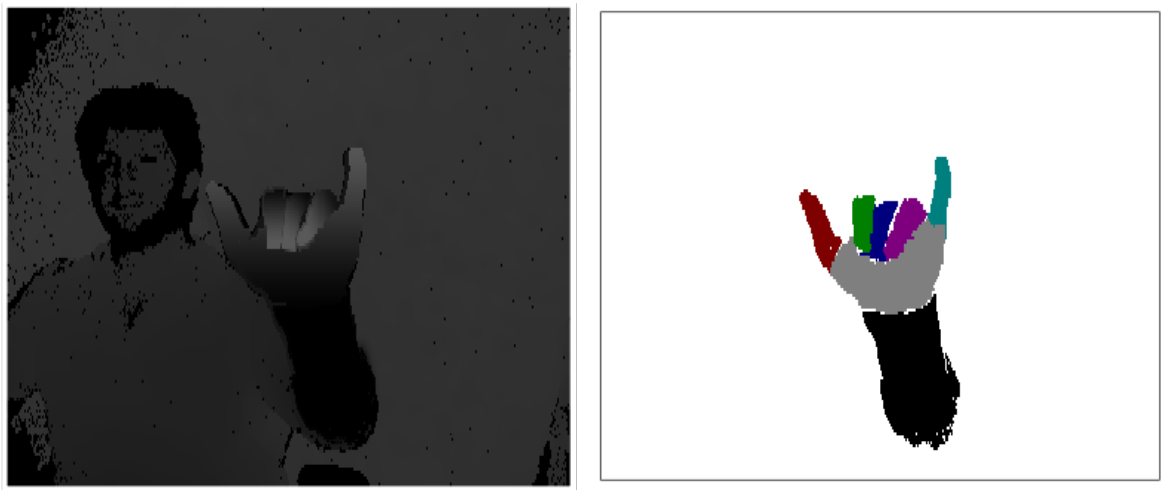
$$CCE\left(y_{true}, y_{pred}\right) = -\sum_{i=1}^{C} y_{true} \log y_{pred} \tag{45}$$

6. Experiments and Discussion

6.1. Training Datasets

6.1.1. Depth-Shape Model

In this work, we selected a suitable dataset for training the depth shape model. The depth shape model needs to focus on the hand components clearly for the recognition of the various hand poses including the hard cases with small-sized hands and self-occlusion hand pose. For this reason, we chose the FingerPaintdataset, as shown in Figure 9, by Sharp et al. [42]. There were five performers, A, B, C, D, and E, with three hand pose subjects to record in the dataset. Regarding the hand pose subjects, the "global" subject focused on the large global movements while the hand pose was almost static; the "pose" subject consisted of gestures with only moving fingers and no hand movement; finally, the "combined" subject was attributed to two subjects, "pose" and "global". There is also the special topic of "penalization", which was based on the performer.



(**a**) (**b**)

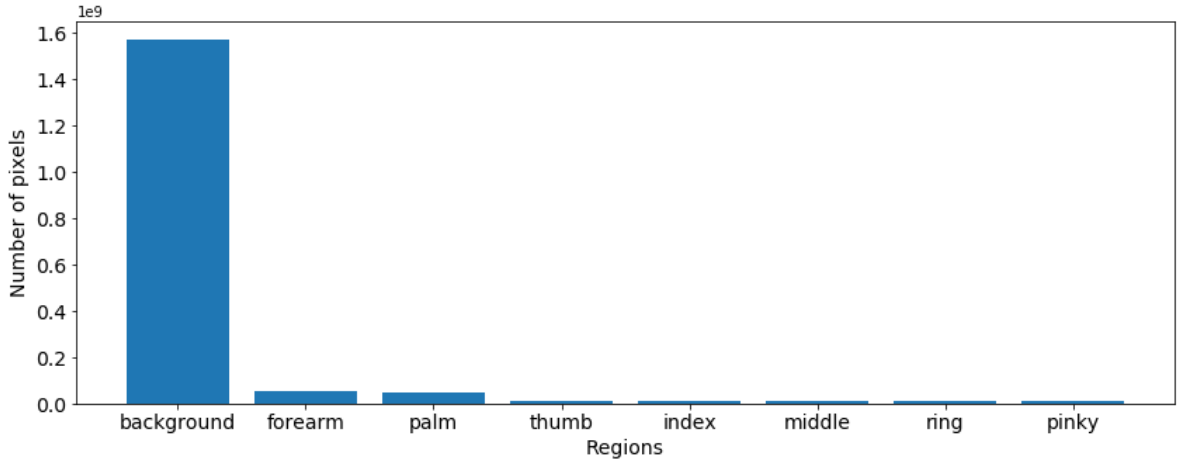
Figure 9. Hand depth data (**a**) and hand component label (**b**) in the FingerPaint dataset.

There was a total 56,500 hand poses with the statistical number of hand poses based on the performer per subject shown in Table 1.

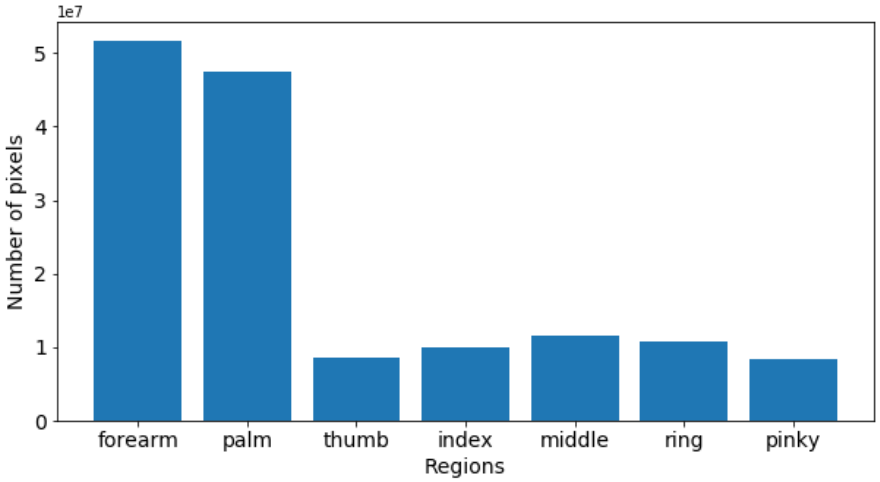
Table 1. Number of hand poses of every performer per subject.

	Subject A	**Subject B**	**Subject C**	**Subject D**	**Subject E**
Global	3500	3500	3500	3500	3500
Pose	3500	3500	3500	3500	3500
Combined	3500	3500	3500	3500	3500
Personalization	800	800	800	800	800

To enhance the segmentation performance, a survey was conducted on the number of pixels between hand components only focusing on the hand region. Figure 10a shows the statistic result of all regions and Figure 10b illustrates on the other regions except the background region.



(**a**)



(**b**)

Figure 10. Number of pixels of the regions in the FingerPaint dataset on all regions (**a**) and regions without the background (**b**).

There was an unbalance between the background and the remaining regions. Without the background, the number of pixels in the forearm and palm regions was larger than the finger regions. For training, we did a split of 70%/30% of every subject and performer for training/validation. Additionally, we used rotation, translation, and scaling by 10% for data augmentation during the training process.

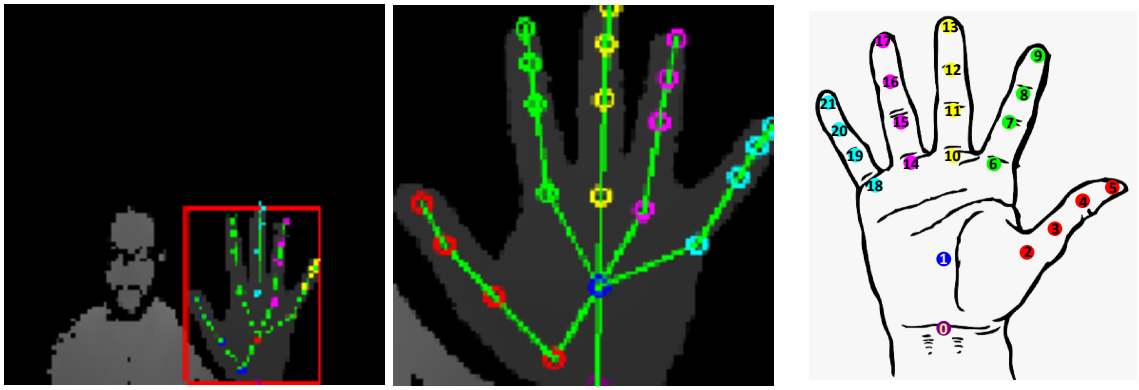
6.1.2. Dynamic Gesture Recognition

For the dynamic hand gesture recognition, we chose the Dynamic Hand Gesture (DHG) dataset containing the hand skeleton and hand depth data suitable for our method. There were 20 performers making up the dataset. Every person performed five gestures in two different ways: using one finger or the whole hand. The dataset had a total of 2800 sequences with 1960 for training and 840 for validation. Every sequence was labeled with 14 or 28 gestures depending on one finger or the overall hand for the ground-truth, as shown in Table 2.

A dynamic hand gesture had a length from 20 to 150 frames. Every frame consisted of a depth image of size 640 $\times$ 480, the skeleton information in the image coordinate, and the world coordinate of 22 hand joints captured by the Intel RealSense camera. Figure 11 shows samples of the gestures in the DHG dataset.

Table 2. List of gestures in the Dynamic Hand Gesture (DHG) dataset.

14 Classes	28 Classes	Gesture	Label
1	1, 2	Grab	Fine
2	3, 4	Expand	Fine
3	5, 6	Pinch	Fine
4	7, 8	Rotation CW	Fine
5	9, 10	Rotation CCW	Fine
6	11, 12	Tap	Coarse
7	13, 14	Swipe Right	Coarse
8	15, 16	Swipe Left	Coarse
9	17, 18	Swipe Up	Coarse
10	19, 20	Swipe Down	Coarse
11	21, 22	Swipe X	Coarse
12	23, 24	Swipe V	Coarse
13	25, 26	Swipe +	Coarse
14	27, 28	Shake	Coarse



(**a**) (**b**) (**c**)

Figure 11. A sample hand posture in the DHG dataset: (**a**) hand depth data with drawing the hand bounding box and the 22 hand joints; (**b**) hand regions in zoom mode; (**c**) 22 hand joints with 0 (wrist), 1 (palm), 2–5 (thumb), 6–9 (index), 10–13 (middle), 14–17 (ring), and 18–21 (pinky).

6.2. Setup Environments, Metrics, and Training Parameters

Environments: Our program was developed with Python 3.5 using the TensorFlow Keras framework to build the deep learning models. Our program ran on a desktop PC with Intel Corei7 8700k with 32 GB of RAM and one graphic card GeForce GTX 1080 Ti.

Metrics: For the depth shape model in the hand component segmentation, we used the metric mean IoU to evaluate our segmentation results. This is the intersection over union between the ground-truth and prediction on every hand region. We used eight hand regions: background, palm, forearm, thumb, index, middle, ring, and pinky.

$$MeanIoU = \sum_{i=1}^{C} \frac{TP_i}{(TP_i + FP_i + FN_i)} 100 \tag{46}$$

where C is the number of hand regions and $TP_i/FP_i/FN_i$ the true positive/false positive/false negative for region i.

For the dynamic hand gesture recognition, we quantified them based on the accuracy between the prediction and ground-truth and the time cost for predicting a gesture.

Parameters in training: We performed a temporal augmentation on the sequence length of a hand gesture by randomizing the position of the first frame and getting 32 frames with equal step sizes. For the spatial augmentation of the depth data, we used basic transforms, such as random rotation by 45 degrees, translation by 5%, and scaling by 10%, based on the frame size.

For training the model, we used Adam [45] with a learning rate of 0.001 and for the first time training reducing the learning rate on the plateau. For the fine-tuning step in the previous training, we used SGD [46] to train with a learning rate ranging from 0.004 to 0.0001 using the cosine annealing learning rate schedule.

Experimenting with the features and models: We conducted the experiments based on the list of features shown in Table 3.

Table 3. List of hand features.

No.	Name	Features	Input	Description
1	Motion	Motion	Skeleton	Movement, rotation, and major axes of the hand
2	Skeleton Shape	Hand Shape	Skeleton	Movement, rotation of joints with neighbors
3	Points	Raw data	Skeleton	Normalize hand joint points
4	Joint point-cloud	3D geometric transformation	Skeleton	Point-Net model
5	Depth shape	Hand components	Depth	Palm, thumb, index, etc., regions

There was a total of five hand features from the two input types, skeleton and depth. We divided the groups of features as the motion group (learning the global motion of hand gestures) with feature motion, hand shape (capturing the changes of hand components) with feature skeleton shape, joint point-cloud, and depth shape, and the others with the feature input by the normalizing points.

For the experiments on our proposed models, we divided our proposed model into with/without Con1D-2D pyramid blocks, as in Table 4.

Table 4. List of the proposed models.

No.	Model Name
1	Multi-Level Feature LSTM without Conv1+2D (MLF LSTM)
2	Multi-Level Feature LSTM with Conv1+2D (MLF LSTM Conv1-2D)

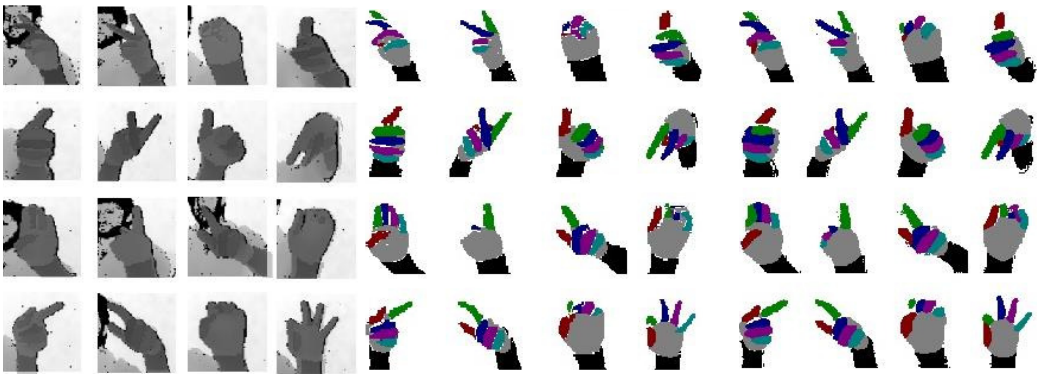
6.3. Results on Hand Component Segmentation

We trained our depth shape models with three types of backbones: VGG16 [40], MobinetV2 [47], and Seresnext [48]. Our results are shown in Table 5.

Table 5. Results of hand component segmentation.

Backbone	Mean IoU
VGG16	82.30%
MobilenetV2	84.00%
Seresnext	86.40%

We achieved the highest mean IoU with the backbone Seresnext and, therefore, chose this backbone for our depth shape model. Figure 12 shows the quality of the depth shape model with the backbone Seresnext compared to the ground-truth.



(**a**) (**b**) (**c**)

Figure 12. Results of hand component segmentation using the Seresnext backbone with the ground-truth depth (**a**), ground-truth labels (**b**), and Seresnext (**c**).

6.4. Results Using the Single Hand Features

We conducted the experiments on two models, MLF LSTM and MLF LSTM with Conv1D and the Conv2D pyramid block, to analyze the influence of single hand features on the DHG dataset with 14 and 28 classes. The results are shown in Table 6.

Table 6. Performance results of the two models using the separate hand features.

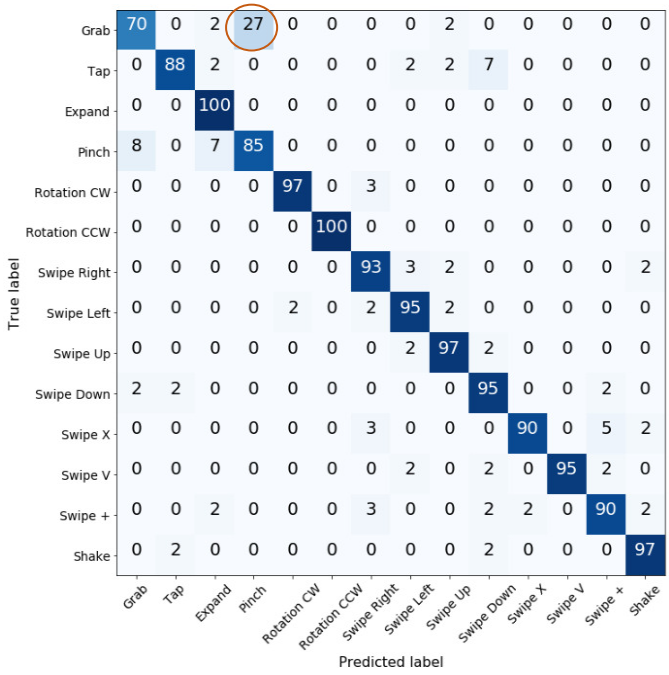
		14 Classes		28 Classes	
No.	**Features**	**MLF LSTM**	**MLF LSTM Conv1-2D**	**MLF LSTM**	**MLF LSTM Conv1-2D**
1	Motion	80.23	82.5	72.85	70.23
2	Skeleton shape	74.76	74.16	70.83	69.4
3	Joint point-cloud	68.92	85.11	56.07	70.23
4	Points	88.33	88.09	82.97	83.09
5	Depth shape	92.26	90.71	87.61	88.33

Motion features are better with the gestures focusing on global movement. However, when performing the complex gestures using from one finger (14 classes) to all fingers (28 classes), the motion features decrease significantly from 82.5% to 70.23%.

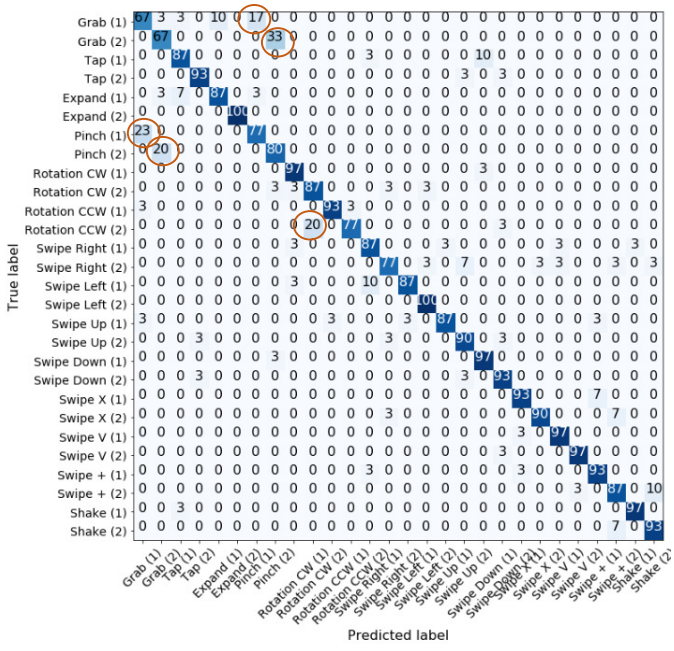
The performance of the models using the depth shape feature only was reduced slightly from 92.26% and 90.71% down to 87.61% and 88.33%. Depth shape features also give the best accuracy of all the features, because they help the model recognize the local motion and also capture the changes of the depth values between two consecutive frames, enabling the model to learn the optical flow features; therefore, the model can recognize global motion.

Upon comparison of the performance between the two models, MLF LSTM Conv1-2D gives better results when using joint point-cloud features with 85.11%/68.92% on 14 classes and 70.23%/56.07% on 28 classes. In contrast, MLF LSTM shows better results of 92.26% and 87.61% on 14 and 28 classes as compared to MLF LSTM Conv1-2D with 90.71% and 88.3%. The different between the two models is the training from scratch and the training from the pre-trained weight.

Figure 13 shows the false cases using the depth shape on DHG 14 and 28. For the 14 classes, the gestures grab and pinch are greatly confused. From 14 to 28 classes, the confusion occurred between grab and pinch in both the one finger gesture and all-finger gesture. Rotation CW(1) and (2) are nearly confused the same between the one finger gesture and the all-finger gesture.



(**a**)



(**b**)

Figure 13. Confusion matrix of the best models using the singleshape feature on DHG 14 (accuracy of 92.26%) (**a**) and DHG 28 (accuracy of 88.33%) (**b**). The red circles are the false cases causing the confusion in recognition.

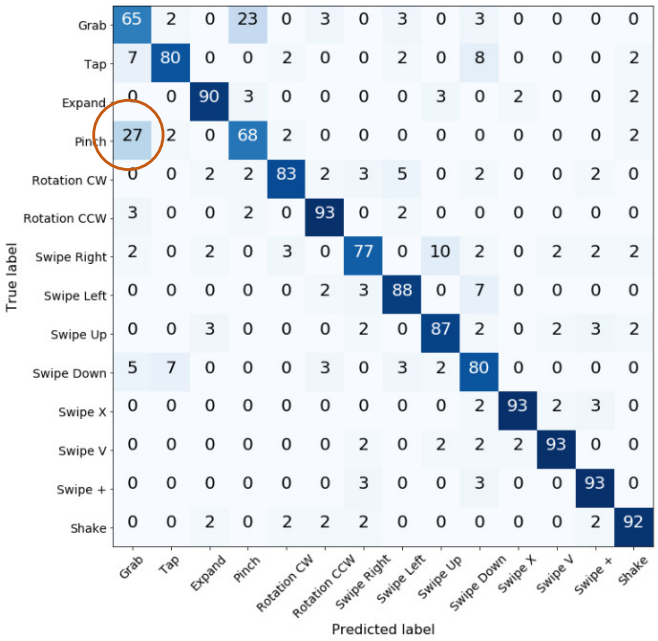
6.5. Experiment 2: Effects of Hand Features in the Skeleton Data

This experiment shows the influences among hand features from skeletons, as shown in Table 7. It has an important role in real-time applications with high requirements for the processing time.

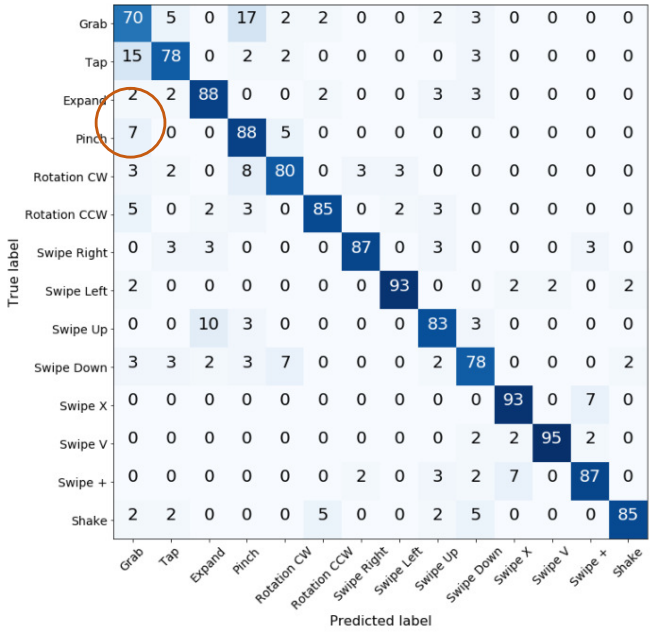
The model using motion features achieved 82.50% on the 14 classes and 72.85% on the 28 classes. When we added hand shape skeletons, our model could capture the local motion between the fingers, increasing our accuracy from 82.50% to 84.52% on the 14 classes and from 72.85% to 82.02% on the 28 classes.

Moreover, the joint point-cloud model could learn good features in 3D geometry and improve the performance of our model by 5.07% (14 classes) and 2.98% (28 classes). Finally, when we integrated normalized point features, we achieved good accuracy on the skeleton data, 93.45% (14 classed) and 90.11% (28 classes).

In Figure 14, the red circle on the left confusion matrix points out the weakness of the model using the motion + skeleton shape feature. The model confused the grab and pinch gestures with the false cases being 27%. In contrast, the joint point-cloud was better with the false cases being 13%. Therefore, the combined results of the two models increased the accuracy from 84.52% to 90.59% (6.07% increase).



(**a**)



(**b**)

Figure 14. Confusion matrix of prediction results using the MLF LSTM Conv1-2D model with the features motion + skeleton shape (accuracy of 84.52%) (**a**) and joint point-cloud (accuracy of 85.11%) (**b**) on DHG-14.

Table 7. Performance results on the skeleton data.

No.	Features	14 Classes		28 Classes	
		MLF LSTM	MLF LSTM Conv1-2D	MLF LSTM	MLF LSTM Conv1-2D
1	Motion	80.23	82.50	72.85	70.23
2	Motion + Skeleton-Shape	86.07	84.52	82.02	81.19
3	Motion + skeleton shape + joint point-cloud	89.52	90.59	85.47	86.78
4	Motion + skeleton shape + joint point-cloud + points	93.45	93.69	89.40	90.11

6.6. Experiment 3: Comparison of Input Data

Table 8 shows that our model achieved the best result on the 14 and 24 classes with 96% and 94.4% combining the skeleton and depth data. Our confusion matrices are as shown in Figures 15 and 16.

Table 8. Overall performance results.

No.	Input	14 Classes		28 Classes	
		MLF LSTM	MLF LSTM Conv1-2D	MLF LSTM	MLF LSTM Conv1-2D
1	Skeleton	93.45	93.69	89.4	90.11
2	Depth2D	92.26	90.71	87.61	88.33
3	All	96.07	94.28	94.4	92.38

Moreover, the model with skeleton input performed better than that with depth data. Because it uses much GPU resource, the model using depth data addresses the performance problem in real-time applications.

6.7. Experiment 4: Comparison with Related Works

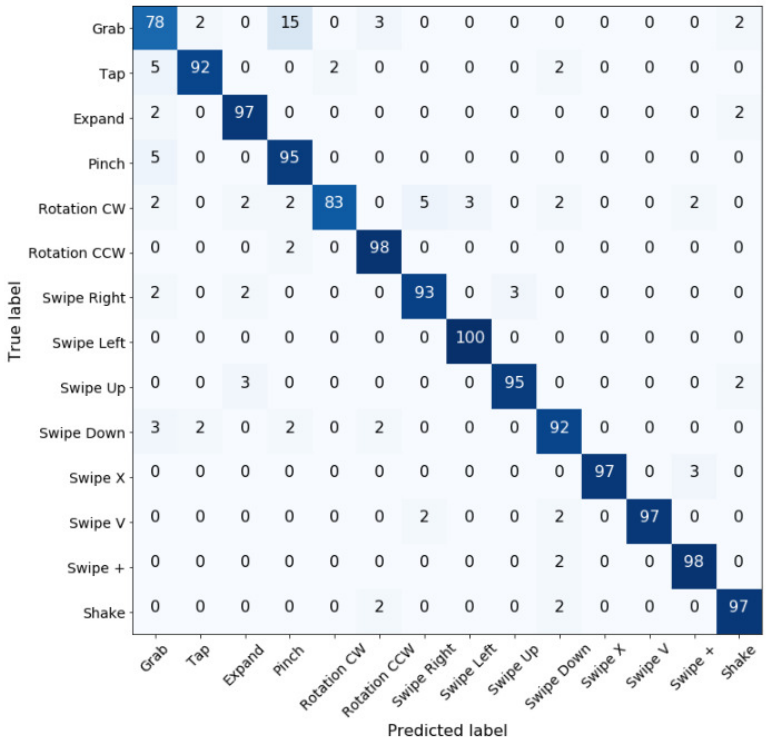
We made a comparative survey of the previous works on the dynamic hand gesture recognition, as shown in Table 9. Smedt et al. [12] built the DHG dataset and conducted their experiments based on the Fisher vector to extract features from the Shape of Connected Joints (SoCJ), built temporal pyramid features, and classified by SVM. Their works achieved state-of-the-art performances of 86.86% and 84.22% on DHG 14 and 28, respectively, with the traditional approach. Using deep learning with the dynamic graph and attention mechanism, Chen at al. successfully achieved the highest accuracy of 91.9% and 88% on DHG 14 and 28 by the deep learning approach.

Due to the multi-modal features between enhancing traditional features and integrating the joint point-cloud model for exploiting the 3D geometric transform, our method gave better results than the other two. The proposed method gave 93.69% and 90.11% on DHG 14 and 28 using skeleton data, as well as 96.07% as 90.11% using both depth and skeleton data.

Table 9. Comparison with the related works. SoCJ, Shape of Connected Joints.

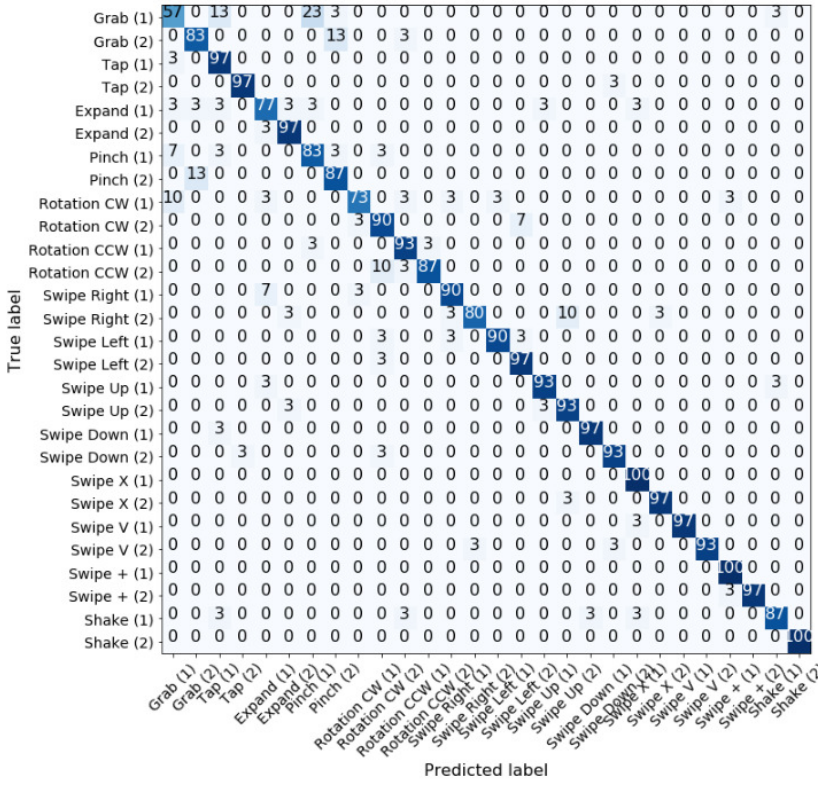
Method	Input	Year	DHG 14	DHG 28
HOG2 [14]	Depth	2013	81.85	76.53
HON4D [22]	Depth	2013	75.53	74.03
MotionManifold [49]	Skeleton	2015	76.61	62
SkeletalQuads [50]	Skeleton	2014	84.5	79.43
Fea-SVM [51]	Skeleton	2014	50.32	30.85
3D Key Frame [52]	Depth	2017	82.9	71.9
MotionFeature+RNN [32]	Skeleton	2017	84.68	80.32
CNN+LSTM [53]	Skeleton	2017	85.6	81.1
STA-Res-TCN [54]	Skeleton	2018	89.2	85
Parallel CNN [55]	Skeleton	2018	91.28	84.35
NIUKF-LSTM [56]	Skeleton	2018	84.92	80.44
ST-GCN [57]	Skeleton	2018	91.2	81.7
SoCJ+HoHD+HoWR [34]	Skeleton	2019	86.86	84.22
DG-STA [58]	Skeleton	2019	91.9	88
GREN [59]	Skeleton	2020	82.29	82.03
Our proposed method	Skeleton		93.69	90.11
Our proposed method	Depth		92.26	88.33
Our proposed method	Overall		96.07	94.4

Our confusion matrices for the proposed methods are as shown in Figures 15 and 16.



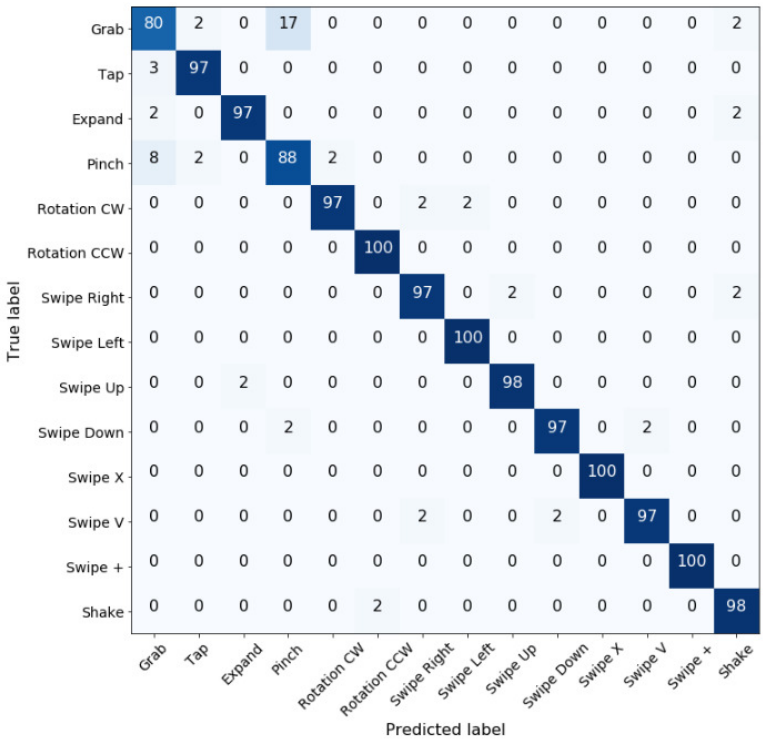
(**a**)

Figure 15. *Cont.*



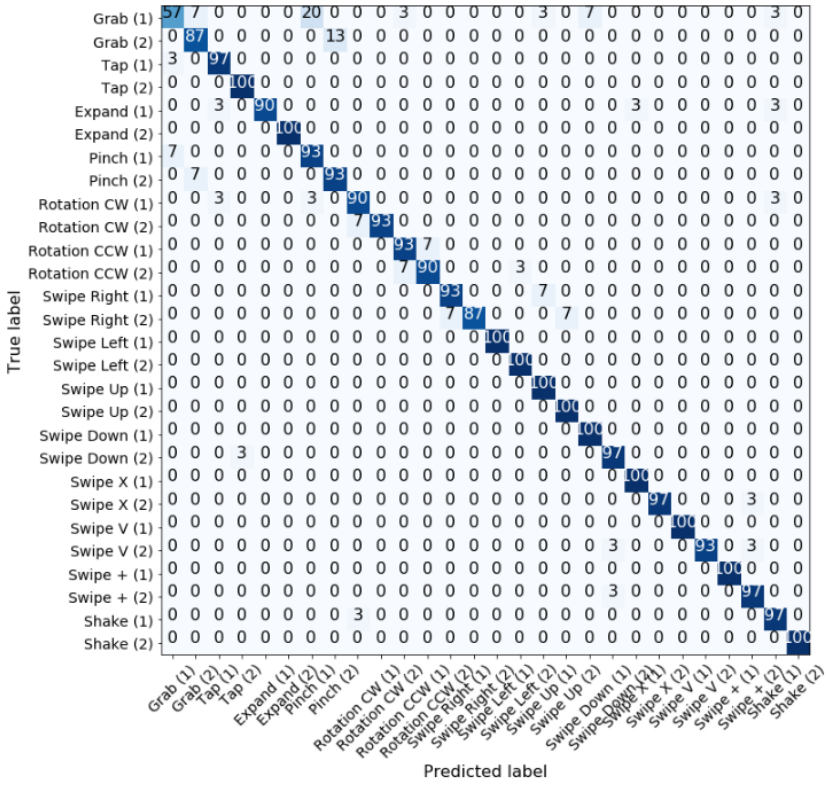
(**b**)

Figure 15. Confusion matrix of the best prediction results on the skeleton data: MLF LSTM Con1-2D with skeleton data (accuracy of 93.69%) (**a**) on DHG-14 and MLF LSTM Conv1-2D with skeleton data (accuracy of 90.11%) (**b**) on DHG-28.



(**a**)

Figure 16. *Cont.*



(**b**)

Figure 16. Confusion matrix of the best prediction results using depth + skeleton data: MLF LSTM (accuracy of 96.07%) (**a**) on DHG-14 and (accuracy 92.38%) (**b**) on DHG-28.

7. Conclusions

In this study, we build a novel method for benefiting from the MLF LSTM model from the 3D geometric transformation and displacement features in hand skeleton data, as well as the hand shape features in depth data from the hand component segmentation model. For the hand skeleton feature approach, we improve the handcrafted features in the motion features by adding the major axes and the skeleton shape through the displacement and rotation of the hand joints with respect to their neighbors. We propose using PointNet in the joint point-cloud model to exploit the 3D geometric transformation on the skeletal data. Our skeleton features improve the performance of our model over the state-of-the-art accuracy with 93.69% and 90.11% on DHG 14 and 28.

For the hand depth feature approach, we also propose using the hand component segmentation features from the depth shape model to recognize the hand shape. Our pre-trained depth shape model was based on U-Net with the Seresnext backbone. Our model using depth shape features gives improved performance with accuracies of 92.26% and 88.33% on DHG 14 and 28.

To learn from the two features, we propose MLF LSTM using Conv1D, the Conv2D pyramid block, and the LSTM block to exploit the hand features. Our model, using depth and skeleton data, gave the best performance with an accuracy of 96.07% and 94.4%. Upon comparison of our model with related works, our model achieves the best results.

In the future, we need to exploit the point-cloud features in the whole hand and enhance the LSTM model to natively integrate the diversity of features from the handcrafted features in the time-series the feature vector from the visual deep learning model and the point-cloud model.

Author Contributions: Conceptualization, N.-T.D. and S.-H.K.; Funding acquisition, S.-H.K., G.-S.L. and H.-J.Y.; Investigation, N.-T.D.; Methodology, N.-T.D.; Project administration, S.-H.K., G.-S.L. and H.-J.Y.; Supervision, S.-H.K.; Validation, N.-T.D.; Writing—original draft, N.-T.D.; Writing—review & editing, N.-T.D. and S.-H.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant NRF-2018R1D1A3A03000947 and Grant NRF-2020R1A4A1019191.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Huang, J.; Zhou, W.; Li, H.; Li, W. Sign Language Recognition using 3D convolutional neural networks. In Proceedings of the IEEE International Conference on Multimedia and Expo (ICME), Turin, Italy, 29 June–3 July 2015; pp. 1–6. [CrossRef]
2. Tan, T.D.; Guo, Z.M. Research of hand positioning and gesture recognition based on binocular vision. In Proceedings of the IEEE International Symposium on Virtual Reality Innovations (ISVRI), Singapore, 19–20 March 2011; pp. 311–315. [CrossRef]
3. Raheja, J.L.; Rajsekhar, G.A.; Chaudhary, A. Controlling a remotely located robot using hand gestures in real time: A DSP implementation. In Proceedings of the 2016 5th International Conference on Wireless Networks and Embedded Systems (WECON), Rajpura, India, 14–16 October 2016; Curran Associates, Inc.: New York, NY, USA, 2017; pp. 1–5. [CrossRef]
4. Lee, S.-H.; Sohn, M.-K.; Kim, D.-J.; Kim, B.; Kim, H. Smart TV interaction system using face and hand gesture recognition. In Proceedings of the IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 11–14 January 2013; pp. 173–174. [CrossRef]
5. Rautaray, S.S.; Agrawal, A. Interaction with virtual game through hand gesture recognition. In Proceedings of the IEEE International Conference on Multimedia, Signal Processing and Communication Technologies, Aligarh, India, 17–19 December 2011; pp. 244–247. [CrossRef]
6. Feix, T.; Pawlik, R.; Schmiedmayer, H.B.; Romero, J.; Kragi, D. A comprehensive grasp taxonomy. In Proceedings of Robotics, Science and Systems Conference: Workshop on Understanding the Human Hand for Advancing Robotic Manipulation, Seattle, WA, USA, 28 June–1 July 2009; pp. 2–3.
7. Wang, R.Y.; Popović, J. Real-time hand tracking with a color glove. *ACM Trans. Graph.* **2009**, *28*, 1–8. [CrossRef]
8. Schroder, M.; Elbrechter, C.; Maycock, J.; Haschke, R.; Botsch, M.; Ritter, H. Real-time hand tracking with a color glove for the actuation of anthropomorphic robot hands. In Proceedings of the 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids), Osaka, Japan, 29 November–1 December 2012; pp. 262–269. [CrossRef]
9. Shotton, J.; Sharp, T.; Fitzgibbon, A.; Blake, A.; Cook, M.; Kipman, A.; Finocchio, M.; Moore, R. Real-Time human pose recognition in parts from single depth images. *Commun. ACM* **2013**, *56*, 116–124. [CrossRef]
10. Potter, L.E.; Araullo, J.; Carter, L. The leap motion controller: A view on sign language. In Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration, Adelaide, Australia, 25–29 November 2013; pp. 175–178. [CrossRef]
11. Lu, W.; Tong, Z.; Chu, J. Dynamic Hand Gesture Recognition with Leap Motion Controller. *IEEE Signal Process. Lett.* **2016**, *23*, 1188–1192. [CrossRef]
12. De Smedt, Q.; Wannous, H.; Vandeborre, J.P. Skeleton-Based Dynamic Hand Gesture Recognition. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 1206–1214. [CrossRef]
13. Zhang, C.; Tian, Y. Histogram of 3D Facets: A depth descriptor for human action and hand gesture recognition. *Comput. Vis. Image Underst.* **2015**, *139*, 29–39. [CrossRef]
14. Ohn-Bar, E.; Trivedi, M.M. Joint angles similarities and HOG2 for action recognition. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Portland, OR, USA, 23–28 June 2013; pp. 465–470. [CrossRef]
15. Oyedotun, O.K.; Khashman, A. Deep learning in vision-based static hand gesture recognition. *Neural Comput. Appl.* **2017**, *28*, 3941–3951. [CrossRef]
16. Molchanov, P.; Gupta, S.; Kim, K.; Kautz, J. Hand gesture recognition with 3D convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Boston, MA, USA, 7–12 June 2015; pp. 1–7. [CrossRef]

17. Wang, J.; Chen, Y.; Hao, S.; Peng, X.; Hu, L. Deep learning for sensor-based activity recognition: A survey. *Pattern Recognit. Lett.* **2019**, *119*, 3–11. [CrossRef]
18. Kuznetsova, A.; Leal-Taixé, L.; Rosenhahn, B. Real-time sign language recognition using a consumer depth camera. In Proceedings of the IEEE International Conference on Computer Vision, Sydney, Australia, 2–8 December 2013; pp. 83–90. [CrossRef]
19. Pugeault, N.; Bowden, R. Spelling It Out: Real – Time ASL Fingerspelling Recognition University of Surrey. In Proceedings of the 2011 IEEE International Conference on THE Hand: Computer Vision Workshops (ICCV Workshops), Barcelona, Spain, 6–13 November 2011; pp. 1114–1119.
20. Dong, C.; Leu, M.C.; Yin, Z. American Sign Language alphabet recognition using Microsoft Kinect. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Boston, MA, USA, 7–12 June 2015; pp. 44–52. [CrossRef]
21. Ren, Z.; Yuan, J.; Meng, J.; Zhang, Z. Robust part-based hand gesture recognition using kinect sensor. *IEEE Trans. Multimed.* **2013**, *15*, 1110–1120. [CrossRef]
22. Oreifej, O.; Liu, Z. HON4D: Histogram of oriented 4D normals for activity recognition from depth sequences. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 716–723. [CrossRef]
23. Monnier, C.; German, S.; Ost, A. A multi-scale boosted detector for efficient and robust gesture recognition. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; Springer: Berlin, Germany, 2015; pp. 491–502. [CrossRef]
24. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
25. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef]
26. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495. [CrossRef]
27. Asadi-Aghbolaghi, M.; Clapes, A.; Bellantonio, M.; Escalante, H.J.; Ponce-Lopez, V.; Baro, X.; Guyon, I.; Kasaei, S.; Escalera, S. A Survey on Deep Learning Based Approaches for Action and Gesture Recognition in Image Sequences. In Proceedings of the 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017), Washington, DC, USA, 30 May–3 June 2017; pp. 476–483. [CrossRef]
28. Sun, L.; Jia, K.; Yeung, D.Y.; Shi, B.E. Human action recognition using factorized spatio-temporal convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015. [CrossRef]
29. Varol, G.; Laptev, I.; Schmid, C. Long-Term Temporal Convolutions for Action Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 1510–1517. [CrossRef]
30. Neverova, N.; Wolf, C.; Taylor, G.W.; Nebout, F. ModDrop: Adaptive multi-modal gesture recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *38*, 1692–1706. [CrossRef]
31. Garcia-Hernando, G.; Yuan, S.; Baek, S.; Kim, T.K. First-Person Hand Action Benchmark with RGB-D Videos and 3D Hand Pose Annotations. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 409–419. [CrossRef]
32. Chen, X.; Guo, H.; Wang, G.; Zhang, L. Motion feature augmented recurrent neural network for skeleton-based dynamic hand gesture recognition. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 2881–2885. [CrossRef]
33. De Smedt, Q. Dynamic Hand Gesture Recognition—From Traditional Handcrafted to Recent Deep Learning Approaches. Ph.D. Thesis, Université de Lille 1, Sciences et Technologies, Lille, France, 2017.
34. De Smedt, Q.; Wannous, H.; Vandeborre, J.P. Heterogeneous hand gesture recognition using 3D dynamic skeletal data. *Comput. Vis. Image Underst.* **2019**, *181*, 60–72. [CrossRef]
35. Ge, L.; Cai, Y.; Weng, J.; Yuan, J. Hand PointNet: 3D Hand Pose Estimation Using Point Sets. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June2018; pp. 8417–8426. [CrossRef]
36. Moon, G.; Chang, J.Y.; Lee, K.M. V2V-PoseNet: Voxel-to-Voxel Prediction Network for Accurate 3D Hand and Human Pose Estimation from a Single Depth Map. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; Volume 2. [CrossRef]

37. Cherabier, I.; Hane, C.; Oswald, M.R.; Pollefeys, M. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the 2016 4th International Conference on 3D Vision, Stanford, CA, USA, 25–28 October 2016; pp. 601–610. [CrossRef]
38. Arun, K.S.; Huang, T.S.; Blostein, S.D. Least-Squares Fitting of Two 3-D Point Sets. *IEEE Trans. Pattern Anal. Mach. Intell.* **1987**, 698–700. [CrossRef]
39. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI*; Springer: Cham, Switzerland, 2015; pp. 1–8. [CrossRef]
40. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. In Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015), San Diego, CA, USA, 7–9 May 2015; pp. 1–14.
41. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [CrossRef]
42. Sharp, T.; Keskin, C.; Robertson, D.; Taylor, J.; Shotton, J.; Kim, D.; Rhemann, C.; Leichter, I.; Vinnikov, A.; Wei, Y.; et al. Accurate, robust, and flexible realtime hand tracking. In Proceedings of the Conference on Human Factors in Computing Systems, Seoul, Korea, 18–23 April 2015; pp. 3633–3642. [CrossRef]
43. Sudre, C.H.; Li, W.; Vercauteren, T.; Ourselin, S.; Jorge Cardoso, M. Generalised Dice Overlap as a Deep Learning Loss Function for Highly Unbalanced Segmentations. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; Springer: Berlin, Germany, 2017; pp. 240–248. [CrossRef]
44. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
45. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015.
46. Schaul, T.; Zhang, S.; LeCun, Y. No more pesky learning rates. In Proceedings of the 30th International Conference on Machine Learning (ICML 2013), Atlanta, GA, USA, 16–21 June 2013.
47. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520. [CrossRef]
48. Hu, J.; Shen, L.; Albanie, S.; Sun, G.; Wu, E. Squeeze-and-Excitation Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**. [CrossRef]
49. Devanne, M.; Wannous, H.; Berretti, S.; Pala, P.; Daoudi, M.; Del Bimbo, A. 3-D Human Action Recognition by Shape Analysis of Motion Trajectories on Riemannian Manifold. *IEEE Trans. Cybern.* **2015**, *45*, 1340–1352. [CrossRef]
50. Evangelidis, G.; Singh, G.; Horaud, R. Skeletal Quads: Human Action Recognition Using Joint Quadruples. In Proceedings of the 2014 22nd International Conference on Pattern Recognition, Stockholm, Sweden, 24–28 August 2014; pp. 4513–4518. [CrossRef]
51. Xu, Y.; Wang, Q.; Bai, X.; Chen, Y.L.; Wu, X. A novel feature extracting method for dynamic gesture recognition based on support vector machine. In Proceedings of the 2014 IEEE International Conference on Information and Automation (ICIA), Hailar, China, 28–30 July 2014; pp. 437–441. [CrossRef]
52. De Smedt, Q.; Wannous, H.; Vandeborre, J.P.P.; Guerry, J.; Le Saux, B.; Filliat, D.; Saux, B.L.; Filliat, D. 3d hand gesture recognition using a depth and skeletal dataset: Shrec'17 track. In Proceedings of the Workshop on 3D Object Retrieval, Lyon, France, 23–24 April 2017; pp. 33–38. [CrossRef]
53. Núñez, J.C.; Cabido, R.; Pantrigo, J.J.; Montemayor, A.S.; Vélez, J.F. Convolutional Neural Networks and Long Short-Term Memory for skeleton-based human activity and hand gesture recognition. *Pattern Recognit.* **2018**, *76*, 80–94. [CrossRef]
54. Hou, J.; Wang, G.; Chen, X.; Xue, J.H.; Zhu, R.; Yang, H. Spatial-temporal attention res-TCN for skeleton-based dynamic hand gesture recognition. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; Springer: Berlin, Germany, 2019; pp. 273–286. [CrossRef]
55. Devineau, G.; Moutarde, F.; Xi, W.; Yang, J. Deep learning for hand gesture recognition on skeletal data. In Proceedings of the 13th IEEE International Conference on Automatic Face and Gesture Recognition, Xi'an, China, 15–19 May 2018; pp. 106–113. [CrossRef]

56. Ma, C.; Wang, A.; Chen, G.; Xu, C. Hand joints-based gesture recognition for noisy dataset using nested interval unscented Kalman filter with LSTM network. *Vis. Comput.* **2018**, *34*, 1053–1063. [CrossRef]
57. Li, Y.; He, Z.; Ye, X.; He, Z.; Han, K. Spatial temporal graph convolutional networks for skeleton-based dynamic hand gesture recognition. *EURASIP J. Image Video Process.* **2019**, 78. [CrossRef]
58. Chen, Y.; Zhao, L.; Peng, X.; Yuan, J.; Metaxas, D.N. Construct Dynamic Graphs for Hand Gesture Recognition via Spatial-Temporal Attention. In Proceedings of the 30th British Machine Vision Conference 2019, Cardiff, UK, 9–12 September 2019; pp. 1–13.
59. Ma, C.; Zhang, S.; Wang, A.; Qi, Y.; Chen, G. Skeleton-based dynamic hand gesture recognition using an enhanced network with one-shot learning. *Appl. Sci.* **2020**, *10*, 3680. [CrossRef]