Proc. 14th IEEE Image, Video, and Multidimensional Signal Processing Workshop (IVMSP 2022), Nafplio, Greece, June 2022.

Attribute-based Gesture Recognition: Generalization to Unseen Classes

George Retsinas, Panagiotis Paraskevas Filntisis, Nikos Kardaris, Petros Maragos
School of Electrical and Computer Engineering, National Technical University of Athens, GR-15773 Athens, Greece
Email: {gretsinas, filby}@central.ntua.gr & nkardaris@mail.ntua.gr & maragos@cs.ntua.gr

***Abstract*—In this paper, we present a hand pose based gesture representation that can effectively classify both static and dynamic gestures. In contrast to resource and data intensive deep learning models, we postulate that hand pose alone can be used to extract compact yet discriminative features that are suitable for most applications that require real-time gesture recognition with minimal computational overhead. Building on the robustness of modern hand pose estimation frameworks and the expressive power of neural networks, we extract a fine-grained gesture description by decomposing gestures in a set of defined attributes. Our approach is highly capable of generalizing to unseen data and unseen classes, as shown by our experiments in the context of the BonsAPPs Challenge use-case.**

I. Introduction

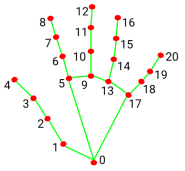
Gesture Recognition is a prominent research field with numerous real-life applications [1], typically in the context of Human-Machine Interaction (HMI), such as gaming and virtual reality [2], sign language recognition [3], touch-less driving [4] and robot perception [5], [6]. Recent advances in body and hand pose estimation have made it possible to extract the hand skeleton with high accuracy and fidelity from raw visual data in real time [7], [8], [9]. In this work, we present a novel skeleton-based gesture recognition system and explore the capabilities of generalization to both unseen data and unseen classes/actions. We argue that the hand pose can be used effectively to extract compact yet expressive and discriminative representations, taking advantage of the robustness of skeleton extraction algorithms, which are trained on large and diverse datasets. We show that this approach allows our model to generalize well to unseen gesture data.

Our case study focuses on the classes set by the BonsAPPs Gesture Recognition Challenge[10]. We were tasked with devising a gesture recognition system for automotive applications, allowing the car driver to interact with the machine interface via gestures. The complete set of classes could not be found in any superset of known datasets and therefore a typical end-to-end approach could not be followed. To this end, we devised a way to address these unseen classes by breaking them into *attributes*. Each class then corresponds to a set of attributes and is classified as such. These attributes are separated into three categories: hand structure (e.g. open hand/peace sign), action (e.g. show/rotate) and direction (e.g. up/down). Using this attribute-based representation, classification is translated to a simple cosine similarity between the predicted attribute and the reference attributes of the unseen classes. The complete methodology and its motivation are presented in Section III.

This work was partially supported by EU grant PILLAR-Robots #101070381.

BonsAPPs Classes	
fist	reach HMI with index
L sign	reach HMI with 2 fingers
hang loose	reach HMI with hand
middlefinger	rotate index CW
ok sign	rotate index CCW
open hand	rotate 2 fingers CW
peace sign	rotate 2 fingers CCW
point index	click with index
point 2 fingers	click with 2 fingers
thumb up	thumb to left
thumb down	thumb to right

(a)



(b)

Fig. 1: 1a List of Gesture Classes as imposed by the BonsAPPs Challenge and 1b the hand model of Mediapipe.

II. Related Work

The importance of gesture recognition for various human machine interaction and perception tasks has led to great advances in the field. Recent approaches on gesture recognition follow the same trend with other computer vision problems, relying on deep neural networks to recognize either static (i.e. handshapes) or dynamic gestures. Thus, many works use two stream methods (e.g. [11], [12], [13]) or 3D CNN methods (e.g. [14], [15], [16], [17]) to encode static and dynamic appearance. A few approaches model the temporal evolution of dynamic gestures using recurrent neural networks and their variants [13], [18], [19], [20]. Various methods focus on combining features from multiple complementary optical data streams, such as RBG, depth, IR, optical flow etc. ([13], [21], [17]).The main idea of most methods is to classify unprocessed streams of uni- or multi-modal visual data.

Skeleton-based gesture recognition using geometric shape features has been widely used in the past, driven by the advent of RGB-D sensors [22], [23]. Recently, there has been an increased interest in end-to-end approaches that use skeleton-based features as input to DNNs. [18] proposes a combination of CNN and LSTM to encode the spatial patterns and temporal evolution of 3D skeleton joints respectively. In [24] sequences of hand joints are processed in parallel by 3 CNN branches to model different time resolutions. The output features of each branch are concatenated and fed to a Multi Layer Perceptron for classification. [25] uses skeleton joint distances and temporal differences to construct location and viewpoint invariant embeddings that encode joint correlations, which are fed to 1D convolutional layers for action and gesture recognition. In [26] the authors propose to decompose dynamic gestures into hand posture evolution and hand movements and construct different skeleton-based representations for each one. These are processed by a 3D and 2D CNN respectively and the output features are combined for classification. [27] processes sequences of 3D hand coordinates using separate temporal and spatial branches. Our method combines elements of different

approaches, as it employs a geometrical gesture description, which is used as prior information to enhance the neural network's ability to discriminate and generalize.

III. Methodology

In this section, we describe in detail our approach to tackle the gesture recognition task of the BonsAPPs challenge. Figure 1a shows the gesture classes involved in the task. The proposed approach comprises of multiple steps, carefully devised to promote generalization properties, as a method of zero-shot learning. We are interested in generalization to *unseen data* and to *unseen classes*. The former is promoted by a well-performing hand skeleton prediction system, while the latter is addressed by the proposed attribute-based approach. In what follows, all essential building blocks are described.

A. Skeleton Extraction

We extract the hand skeleton using the off-the-shelf state-of-the-art 3D hand skeleton extraction module provided by the Google Mediapipe library [7]. The input of the module is a monocular RGB image and the extracted hand model skeleton can be seen in Fig. 1b. We have decided to use this module due to the fact that the library supports multiple platforms and has end-to-end acceleration for common hardware (not relying only on high-performance graphics card), thus enabling easier integration into real-life HMI applications. Furthermore, it is important to note that the model has been tested in people all around the globe, mitigating ethnicity and skin color biases.

B. Static Gestures

First, we focus on the set of "static" gestures, i.e., gestures that do not require any specific actions (essentially they correspond to the "show" action, where the user simply presents as specific hand-shape) and can be categorized only by the hand shape: different finger configurations correspond to different classes. The static gestures of interest are listed in the first column of Table 1a. The complete set of the required static classes could not be found in the existing datasets. To address this, we created a small dataset of static gestures (2 persons for training and 1 for testing). The collection protocol promoted simplicity and generalization: users perform a gesture for a few seconds with a moving hand, in order to capture slight variations of the same gesture - each frame is treated as a different instance of the same class.

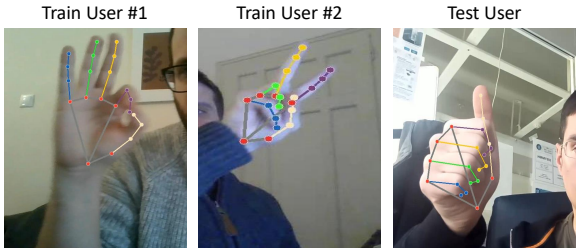


Fig. 2: Example frames with the corresponding skeleton hand landmarks, taken from the collected static gesture dataset.

To build an efficient gesture recognition system for this set, we extracted a set of intuitive "handcrafted" features from each skeleton and feed them into a multi-layer network comprised of fully connected modules, as described below.

Preprocessing: From each landmark in the hand pose (Figure 1b) we subtract the coordinates of point #0, i.e. $x_i^t = x_i - x_0$ to achieve translation invariance. Then we normalize the result with respect to the size of the palm, i.e. $x_i^s = x_i^t / ||x_5^t||_2$ to achieve scale invariance.

Our feature vector contains 21 elements derived from the normalized coordinates that intuitively describe the hand pose, including 3d hand direction, cosine similarities between each finger and the palm counterpart (e.g. for thumb it would be $cos(x_4 - x_2, x_2 - x_0)$)), between neighboring fingers (thumb-to-forefinger $cos(x_4 - x_2, x_8 - x_5)$ etc.) and relative distance between the forefinger and thumb tips and between the forefinger and middle finger tips (useful for "ok" and "peace" signs).

Note that the initial normalization does not affect cosine similarities (only the requested relative distances). Cosine features are helpful since they interpret the structure of the hand in simple geometric terms of angles between edges. In addition, cosine similarities and relative distances reinforce rotation invariance.

Augmentation: Due to the straightforward interpretation of the extracted features, we can easily construct "negative" examples for each class and assist the subsequent training procedure. One such example is the "closing" and "opening" of fingers for which we control the cosine similarities between fingers and their palm counterparts. Values close to 1 denote that the finger is extended, while values < 0 denote that the finger is close to the palm. Further basic augmentation is performed by adding Gaussian noise.

Model and Training: The selected model is a simple feed-forward network, which takes as input the extracted features (of size 21) and returns a vector of size equal to the number of classes (11). It is comprised of 4 hidden layers of size 128. Between these layers we use ReLU activations and Dropout. We choose binary cross entropy loss along with a sigmoid activation function on the output of the network in order to train for the existence or not of each class, along with the augmented negative examples which correspond to absence of classes. We train the network using Adam [28] optimizer for 60 epochs, with learning decay rate 0.1 at 30 and 45 epochs.

C. Training with Attributes

Next, we focus on learning "dynamic" gestures. To achieve this, we use two datasets that contain the higher level actions (move, rotate, click, approach) and directions (left/right, in/out, clock-wise/counter-clock-wise) that we were interested in.

The selected datasets are the widely-used and challenging NVGesture [13] dataset and the Dynamic Hand Gesture Recognition (DHGR) dataset [29]. The former has high variability in performed gestures and "temporal" noise, since each action is performed only in a subset of frames, while the latter has the exact same classes, plus two more, but in a much more constrained and less challenging setting. The gesture classes of these datasets are summarized in Table I.

Since many of the classes of interest (Fig. 1a) are not present in these datasets, we cannot sufficiently train a classification model. To tackle this, we adopted an attribute-based approach where each on of the existing classes (Table I) is translated into a set of useful attributes that can be used to "decode" all requested classes. The main idea behind the proposed

TABLE I: List of Gesture Classes included in NVGesture and DHGR datasets

	NVGesture	DHGR		NVGesture	DHGR
Move Hand Left	✓	✓	Show Three Fingers	✓	✓
Move Hand Right	✓	✓	Push Hand Up	✓	✓
Move Hand Up	✓	✓	Push Hand Down	✓	✓
Move Hand Down	✓	✓	Push Hand out	✓	✓
Move 2 Fingers Left	✓	✓	Push Hand In	✓	✓
Move 2 Fingers Right	✓	✓	Rotate Fingers CW	✓	✓
Move 2 Fingers Up	✓	✓	Rotate Fingers CCW	✓	✓
Move 2 Fingers Down	✓	✓	Push 2 Fingers Away	✓	✓
Click Index Finger	✓	✓	Close Hand Two Times	✓	✓
Call Someone	✓	✓	Thumbs Up	✓	✓
Open Hand	✓	✓	Okay Gesture	✓	✓
Shaking Hand	✓	✓	Thumb Left	✗	✓
Show Index Finger	✓	✓	Thumb Right	✗	✓
Show Two Fingers	✓	✓			

gesture recognition pipeline is that both the requested classes as well as the available classes from NVGesture and DHGR can be broken down into a common set of attributes of the form: **hand structure** (static gestures) + **action** + **direction**. Ideally, a $1-1$ correspondence between attribute vectors and classes should exist in order to perform classification in the attribute domain. Following the aforementioned rationale, the attributes are separated into three categories:
actions: show, move, rotate, click, reach, shake, call, swipe
directions: up, down, left, right, in, out, cw, ccw, nodir
hand structure: the 11 static gesture classes.

The representation of each class with respect to the selected attributes is straightforward (the correspondence tables were omitted due to space limitations). Each class is eventually represented by one one-hot vector for each attribute category. The final attribute vector is the concatenation of the per-category one-hot vectors. Building on the attribute-based approach for our problem, we describe the proposed system for recognizing dynamic gestures and its substeps as follows.

Preprocessing: To create the input for this task, we transform each frame's landmarks into a feature vector (as described in the previous section), concatenate them, and use them as input to the current system along with the difference of sequential frames, as an estimate of velocity. Overall, the proposed system has two inputs: a sequence of handcrafted features for the static module and a sequence of concatenated normalized landmarks and their velocity. Each frame corresponds to 2 (normalized landmarks & difference of consecutive landmarks) $\times$ 21 (landmarks) $\times$ 3 (3D coordinates) features.

Augmentation: We apply the following augmentations:
-*temporal speed:* simulate the same activity with different speed by temporal interpolation.
-*(temporal) rigid transformation*: find a rigid transform by randomly selecting 3 hand points and translate them. Perform this on a random number of frames. Interpolate for all frames.
-*change view:* apply a random global rotation to each frame to simulate extracted skeletons from different view angles.
-*reverse frames:* reverse the order of frames and change the classes/attributes accordingly (left $\rightarrow$ right, up $\rightarrow$ down etc.)
-*frame mask/point mask:* randomly zero entire skeleton on few frames or single hand points.
-*gaussian noise:* insert gaussian noise of small magnitude.

Model: The backbone of the proposed model is an efficient lightweight 1D CNN (to enable real-time recognition) that encodes the required temporal correlation of successive hand poses. Overall, a temporal segment scheme [30] was adopted in order to capture useful information across the whole video. Segments of 15 frames were extracted from the hand skeleton sequence. A 1D CNN is applied to each such segment. The CNN outputs of the segments, which are evenly distributed with respect to the duration of the video, are then combined with average pooling. The resulting feature vector is then transformed by feed forward networks to compute the final predictions (one fully-connected layer predicts classes and another one attributes).

The aforementioned pipeline is further fine-tuned using the following ideas: **1)** The average pooling operation across segments is implemented as weighting average, where the weights are also estimated by an extra single fully-connected layer and a sigmoid activation in a self-attention manner (see Figure 3). The goal of this module is to assist the network to discard non discriminative parts of a gesture video, e.g. parts at the beginning or the end of the video where no specific action is performed. The estimated weight value can be also used as a threshold to predict a gesture/no gesture mode. **2)** The static gesture classification model is used as an off-the-self subnetwork which can distinguish all the required "static" gestures. This network has also an attribute-like logic and estimates the per-frame probability for each static gesture to appear. A summarization of the static gestures' existence is provided by a max pooling over all frames. An overview of this approach is depicted in Figure 3.

Finally, we tune the described architecture to increase the generalization ability of the final system. Specifically, we "break" the hand skeleton representation into separate fingers, trying to introduce finger-independent representations. Each finger is then processed by the same 1D CNN and the overall feature vector that describes the processed segment is extracted as the average pooling of the per-finger features. The motivation is simple; we do not let the network explicitly learn that clicking can be performed only by one finger, since only such cases exist on the training sets, but rather learn the "clicking" action independently of the finger. The finger configuration can be then introduced by the static gesture module of our system.

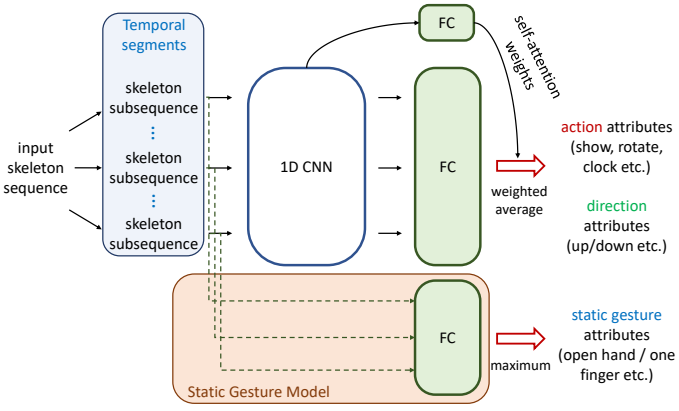


Fig. 3: Overview of the proposed system and its critical sub-components: 1) temporal segment approach, 2) 1D CNN 3) pre-trained static gesture model 4) self-attention weight averaging.

Overall, we distinguish two architectures to be used:

the first is a more typical end-to-end approach (depicted in Figure 3), while the second tries to "absorb" biases that may hinder generalization as described above. The former is dubbed as *Skeleton2Attributes*, while the latter is dubbed as *GSkeleton2Attributes*. Note that both alternatives are very compact with less than $1M$ parameters, which makes them appropriate for real-time applications.

Training: As shown in the model description, we predict both the original classes and their attribute representations. Thus, a multitask loss is adopted, that consists of two loss terms: cross entropy loss (along with softmax activation) for class prediction and binary cross entropy loss (along with sigmoid activation) for attribute prediction. The final multitask loss is balanced with respect to the individual loss terms: $L_{CE}(y_{classes}; t_{classes}) + L_{BCE}(y_{attributes}; t_{attributes})$. The target vectors are class labels for the class prediction case and a binary vector of attributes ('1' for existing attributes, '0' otherwise) for the case of attribute prediction.

Training is performed for 1000 epochs, using Adam [28] and a multi-step scheduling scheme, as previously. For the static gesture recognition submodule, we use the already trained weights following the approach of the previous section. These weights are kept frozen during this training procedure.

D. Evaluating with Attributes

Given a set of "unseen" classes, we perform classification by simply creating reference attribute vectors, translating each class into a set existent/non-existent attributes, and compare them with the predicted attributes. Comparison is performed by computing the cosine similarity. A gesture is classified as the class with the most similar attributes.

IV. Experimental Results

A. Static Gestures

First, we evaluated our trained DNN model designed for the static gestures (see III-B) on the collected dataset. The overall recognition rate for the considered 11 classes was 93.42%. Despite achieving over 96% over the majority of the classes, two specific classes are often confused: "L sign" and "Pointing forefinger" (the latter has only 65% accuracy). Nonetheless, such confusion is to be expected since both gestures are presented by extending the forefinger and since we change the orientation of the hand during capturing, "L sign" can be indeed interpreted as "pointing forefinger".

B. Training/Testing on NVGesture and DHGR Datasets

The proposed final gesture recognition system is trained either on NVGesture or on DHGR dataset and typical accuracy metrics on these datasets consist a basic measure of how effective our model is. Only single-view RGB information is considered for the NVGesture dataset (the dataset contains also IR and depth). As we can see in Table II, both architectures perform very well on the corresponding tasks, when trained and evaluated on the same dataset, while the generalization-oriented architecture (GSkeleton2Attributes) seems to perform slightly worse compared to its alternative. The recognition accuracy is reported using the class prediction head. Attribute-prediction branch leads to a drop in performance close to 3% in both datasets. We should highlight that state-of-the-art approaches that are based on the whole RGB images have similar performance with much larger architectures (e.g. 81.3% in the recent work of Abavisani et al. [17]), while the 3D CNN approach proposed by the creators of the NVGesture dataset[13], achieves only 74.1% using RGB input.

TABLE II: Recognition accuracy(%) in NVGesture/DHGR

Architecture	NVGesture	DHGR
Skeleton2Attributes	79.25	97.03
GSkeleton2Attributes	77.18	96.65

C. Generalization to BonsAPPs Unseen Classes

Even though the proposed models are fairly successful for the aforementioned setting, the main goal of this work is to test the attribute-based approach on the BonsAPPs classes. To this end, we collected an evaluation set that contains the classes we are interested in. These were performed by 2 subjects, different from the training set for the static gestures (5-10 times each gesture) resulting in 287 videos in total. Videos were captured with typical webcams, while no specific environment or protocol was enforced. The results of the proposed attribute-based classification are summarized in Table III. Apart from the two different architectures, we considered training only on NVGesture, only on DHGR, or on a merged set of both datasets. Taking into account the difficulty of the task at hand (recognition of unseen classes), we reported top-k results.

Even though top-1 accuracy is lower than 60%, our system achieves 87.8% accuracy when the top 3 candidates are considered. Notably, the generalization-oriented architecture seems to be effective in this setting (top-3 accuracy is only 80.14% for Skeleton2Attributes, compared to 87.8% for the GSkeleton2Attributes case), supporting our initial motivation of decoupling actions from hand structures (classes such as "rotate with two fingers" or "click with two fingers" benefit greatly from GSkeleton2Attributes architecture). As expected, pairs of classes such as "pointing with two fingers"/"reaching HMI with two fingers" or "open hand" and "reaching HMI with hand" are often confused, since they are closely correlated classes than can even confuse a human.

TABLE III: Top-K accuracy for the BonsAPPs Classes

	Skeleton2Attributes			GSkeleton2Attributes		
trained on:	NVGesture	DHGR	merged	NVGesture	DHGR	merged
k=1	54.35	42.15	54.35	53.31	43.9	58.18
k=2	67.94	55.4	65.16	68.64	61.32	74.21
k=3	79.79	67.00	80.14	82.57	73.87	87.8
k=4	86.41	75.96	88.85	89.9	84.32	91.99
k=5	90.24	83.62	94.08	93.73	88.85	94.43

D. Online Demo

An online web demo is available that runs in real-time and classifies user gestures performed in front of a webcam. The demo can be found at https://robotics.ntua.gr/wp-content/demos/gesture-attr-demo/.

V. Conclusion

In this paper we proposed a skeleton-based approach for real-time gesture recognition. We extract discriminative representations by decomposing gestures into a set of attributes, which enhances the generalization ability of our models. We demonstrate the efficacy of our approach on a set of unseen gesture classes.

REFERENCES

[1] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 1–54, Jan 2015.

[2] Z. Lv, A. Halawani, S. Feng, S. ur Réhman, and H. Li, "Touch-less interactive augmented reality game on vision-based wearable device," *Personal and Ubiquitous Computing*, vol. 19, no. 3, pp. 551–567, Jul 2015.

[3] N. C. Camgoz, S. Hadfield, O. Koller, and R. Bowden, "Subunets: End-to-end hand shape and continuous sign language recognition," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[4] E. Ohn-Bar and M. M. Trivedi, "Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations," in *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 6, 2014, pp. 2368–2377.

[5] N. Efthymiou, P. P. Filntisis, G. Potamianos, and P. Maragos, "Visual robotic perception system with incremental learning for child–robot interaction scenarios," *Technologies*, vol. 9, no. 4, p. 86, 2021.

[6] A. Zlatintsi, A. Dometios, N. Kardaris, I. Rodomagoulakis, P. Koutras, X. Papageorgiou, P. Maragos, C. Tzafestas, P. Vartholomeos, K. Hauer, C. Werner, R. Annicchiarico, M. Lombardi, F. Adriano, T. Asfour, A. Sabatini, C. Laschi, M. Cianchetti, A. Güler, I. Kokkinos, B. Klein, and R. López, "I-support: A robotic platform of an assistive bathing robot for the elderly population," *Robotics and Autonomous Systems*, vol. 126, p. 103451, 2020.

[7] https://google.github.io/mediapipe/solutions/hands.html.

[8] W. Huang, P. Ren, J. Wang, Q. Qi, and H. Sun, "Awr: Adaptive weighting regression for 3D hand pose estimation," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[9] T. Simon, H. Joo, I. Matthews, and Y. Sheikh, "Hand keypoint detection in single images using multiview bootstrapping," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 1145–1153.

[10] https://beta.bonseyes.com/.

[11] J. Wu, P. Ishwar, and J. Konrad, "Two-stream cnns for gesture-based verification and identification: Learning user style," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016, pp. 110–118.

[12] J. Wan, S. Escalera, G. Anbarjafari, H. Jair Escalante, X. Baro, I. Guyon, M. Madadi, J. Allik, J. Gorbova, C. Lin, and Y. Xie, "Results and analysis of chalearn lap multi-modal isolated and continuous gesture recognition, and real versus fake expressed emotions challenges," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

[13] P. Molchanov, X. Yang, S. Gupta, K. Kim, S. Tyree, and J. Kautz, "Online detection and classification of dynamic hand gestures with recurrent 3D convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4207–4215.

[14] Q. Miao, Y. Li, W. Ouyang, Z. Ma, X. Xu, W. Shi, and X. Cao, "Multimodal gesture recognition based on the ResC3D network," in *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2017, pp. 3047–3055.

[15] W. Zhu, J. Hu, G. Sun, X. Cao, and Y. Qiao, "A key volume mining deep framework for action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1991–1999.

[16] C. Lin, J. Wan, Y. Liang, and S. Z. Li, "Large-scale isolated gesture recognition using a refined fused model based on masked Res-C3D network and skeleton LSTM," in *Proceedings of the 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*, 2018, pp. 52–58.

[17] M. Abavisani, H. R. V. Joze, and V. M. Patel, "Improving the performance of unimodal dynamic hand-gesture recognition with multimodal training," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[18] J. C. Núñez, R. Cabido, J. J. Pantrigo, A. S. Montemayor, and J. F. Vélez, "Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition," *Pattern Recognition*, vol. 76, pp. 80–94, 2018.

[19] G. Chalvatzaki, P. Koutras, A. Tsiami, C. S. Tzafestas, and P. Maragos, "i-Walk intelligent assessment system: Activity, Mobility, Intention, Communication," in *Proceedings of the European Conference on Computer Vision Workshops (ECCVW)*, 2020.

[20] Y. Min, Y. Zhang, X. Chai, and X. Chen, "An efficient PointLSTM for point clouds based gesture recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5761–5770.

[21] P. Narayana, R. Beveridge, and B. A. Draper, "Gesture recognition: Focus on the hands," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[22] Z. Ren, J. Meng, J. Yuan, and Z. Zhang, "Robust hand gesture recognition with kinect sensor," in *Proceedings of the 19th ACM international conference on Multimedia*, 2011, pp. 759–760.

[23] Q. De Smedt, H. Wannous, and J.-P. Vandeborre, "Skeleton-based dynamic hand gesture recognition," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016, pp. 1206–1214.

[24] G. Devineau, F. Moutarde, W. Xi, and J. Yang, "Deep learning for hand gesture recognition on skeletal data," in *Proceedings of the 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*, 2018, pp. 106–113.

[25] F. Yang, Y. Wu, S. Sakti, and S. Nakamura, "Make skeleton-based action recognition model smaller, faster and better," in *Proceedings of the ACM multimedia Asia*, 2019, pp. 1–6.

[26] J. Liu, Y. Liu, Y. Wang, V. Prinet, S. Xiang, and C. Pan, "Decoupled representation learning for skeleton-based gesture recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 5750–5759.

[27] Y. Li, D. Ma, Y. Yu, G. Wei, and Y. Zhou, "Compact joints encoding for skeleton-based dynamic hand gesture recognition," *Computers & Graphics*, vol. 97, pp. 191–199, 2021.

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[29] G. Fronteddu, S. Porcu, A. Floris, and L. Atzori, "Dataset for dynamic hand gesture recognition systems," IEEE Dataport, 2021. [Online]. Available: https://dx.doi.org/10.21227/43mn-bb52

[30] J. Lin, C. Gan, and S. Han, "Tsm: Temporal shift module for efficient video understanding," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.