Hand Pose Ensemble Learning Based on Grouping Features of Hand Point Sets

Tianqiang Zhu Yi Sun Xiaohong Ma Xiangbo Lin
Dalian University of Technology
No.2 Linggong Road, Ganjingzi District, Dalian City, Liaoning Province, P.R.C.
{ zhutq, lslwf, maxh, linxbo }@mail.dlut.edu.cn

Abstract

In this paper, we mainly consider using 3D point sets as input to deal with the task of 3D hand pose estimation. We make some improvements to PointNet++ structure, including proposing adaptive pooling which introduces the self-attention mechanism to make the network could select features itself, and putting forward an ensemble strategy to fully utilize hand features. These improvements can enhance the expressive ability of features and make full use of the information contained in features. In addition, we propose a data augmentation method for point net, which directly transforms the original point cloud data without the aid of simulation models. Experiments results on three hand pose datasets demonstrate that our method can achieve comparable performance with state-of-the-arts.

1. Introduction

3D hand pose estimation has attracted the attention of many scholars for its role in the field of computer vision, virtual reality and robotics. It has made rapid progress in recent years, benefiting from the development of deep learning and advances of depth cameras. Nevertheless, due to the gesture diversity, finger self-similarity and self-occlusion, there are still many challenges to reach the practical requirements.

Most of the current depth-based hand pose estimation methods use convolutional neural network (CNN) directly for depth image and improve the test accuracy by designing unique network structure. For example, Oberweger *et al*. [15, 17] enforce the constraints of hand pose by learning a prior model. Chen *et al*. [3, 10, 18] force the network to learn more effective features for hand joints by guiding the network to focus on local areas. However, mapping from depth image to 3D coordinate of hand joints is highly nonlinear and brings challenges to achieve high prediction accuracy [31]. Hence, there are some attempts to use different forms of input recently. For example, Moon *et al*. [8, 13] use 3D voxels to make full use of 3D information. Ge *et al*. [6, 9] successfully use PointNet++ [20, 21] in hand pose estimation, achieve directly mapping from point cloud to hand joints. Because point cloud can represent the structure of hand surface with less data than voxel, we use the point cloud as network input in this paper. Different from [6], who directly uses the PointNet++ structure [20, 21] as the backbone of the network. We make some improvements to PointNet++ structure [20, 21]: the max-pooling operations are all replaced by the proposed adaptive pooling, as shown in Figure 2, which introduces the self-attention mechanism [33] and make the network has the ability to select features, thus greatly reducing the waste of features brought by the max-pooling. We call the advanced PointNet++ structure "A-PointNet++".

By use "A-PointNet++" structure, we can get many local features, which we call them "Coarse features", as shown in Figure 1. Unlike Ge *et al*. [6], they directly pool all features into one feature by max-pooling to directly regress all hand joints. In our opinion, those local features are certainly independent of each other, but the max-pooling operation will ignore this independence and result in the inability to acquire feature that can express the whole hand. Inspired by ensemble learning, we want to use more local features to predict hand pose and integrate those results by learning to get a more accurate result. In this way, we can bridge the gap caused by max-pooling, and make use of every local features.

In addition, this paper adopts a new method to augment dataset for point net, which can effectively improve the network performance by processing the original data directly. We first divide the hand into 16 parts according to Euler distance, where the centroids are calculated by the joints in label. Then we bend fingers according to kinematics to get new gesture. The new data not only enrich the diversity of gesture, but also can assist the network to improve the performance in original data. Because, for point net, the new data are also part of surface of the whole hand, which help the point net learn the relationship more clearly between joints and points of the hand surface.

Our main contributions are summarized as follows:

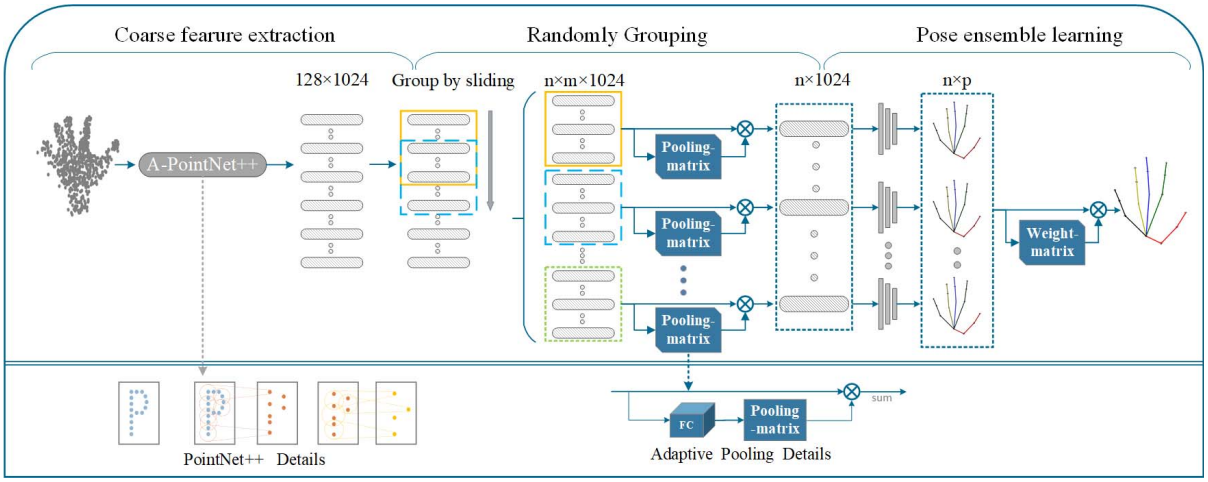


Figure 1: The framework of our proposed network structure.

- We propose an adaptive pooling method based on self-attention mechanism [33], which reduces the information loss of feature caused by max-pooling in PointNet++ structure.
- Inspired by ensemble learning, we fully utilize all local features and integrate all results by learning to obtain more accurate prediction result.
- We propose a data augmentation method for point net, by segmenting hand with joints in label and then rotating each part according to the hand kinematics to add gestures. These new data can improve the performance of network in original data significantly.

2. Related work

Hand pose estimation methods can be classified into three categories: generative methods, discriminative methods and hybrid methods, and have achieved great progress [5, 23, 26, 28, 38]. Our method belongs to the discriminative methods, relying on a large amount of data to predict 3D hand joints with deep learning. Specifically, we use hand point cloud as the input of our model, so we mainly focus on those methods, which regress hand joints coordination with the 3D form input.

3D learning for hand pose estimation: Most of the recently proposed 3D hand pose estimation methods [2,11,14, 19,24,32,37,39–41] are based on 2D CNN, where they use RGB images or depth images directly as the input of the network to predict the coordinates of hand joints. However, it is difficult to make full use of the 3D information contained in RGB or depth images. To tackle this problem, Ge *et al*. [7] project the depth image to three viewpoints and input them together to the network for improving the performance in original data. Nevertheless, predicting 3D joints pose directly from the RGB image and depth map is highly nonlinear, which hinders the extraction of three-dimensional information. Hence, Ge *et al*. [8] further propose to transform the depth image into voxel representation as the network input and use 3D CNN to generate volumetric heat-maps, which is a natural and effective method to make full use of 3D information. Similarly, Moon *et al*. [13] prove that the use of voxel can significantly improve the prediction accuracy compared with the direct use of depth image. However, those methods are computationally inefficient because of the huge consumption of computing resources. With the development of point cloud processing network in recent years, Ge *et al*. [6, 9] attempt to use the point cloud as the network input. Compared to voxel representation, both of them intuitively express the three-dimensional information of data, but point clouds can be obtained directly from depth sensors, and data preprocessing is simpler and more direct. Therefore, in this paper, we also use point cloud as network input.

Figure 2: Adaptive pooling.

PointNet++ [21] is a neural network specially designed to deal with unordered point sets. Its previous version, basic PointNet [21], directly map the input points to per-point features by a multi-layer perception networks (MLP), and then pool those per-point features into a global feature by max-pooling operation. But the basic PointNet [21] does not have the ability to explore local details of point cloud, so PointNet++ [21] builds a hierarchical grouping of local features based on basic PointNet [21] and progressively abstract larger and larger local regions. However, we find that it is inappropriate to use PointNet++ [21] directly into hand

pose estimation like Ge *et al*. [6, 9]. Because max-pooling would lead to the loss of some useful information, and the last pooling layer would also destroy the independence of each local feature. Hence, we make some improvements of PointNet++ [21] and propose a new network structure that is more suitable for hand pose estimation.

Data augmentation methods: For data-driven methods, augmenting data with effective method can significantly improve performance [16]. Oberweger *et al*. [15, 17] improve the robustness of the network by randomly rotating, translating and scaling images in dataset. Further, Ge *et al*. [7] augment the dataset by rotating the hand in 3D space and projecting it to three viewpoints. But these methods do not increase the diversity of gestures and can not solve the problems caused by the high degree of spatial freedom of the hand. Rad *et al*. [22] effectively improve the prediction accuracy by using synthetic data to increase the diversity of gestures. Wan *et al*. [1, 25, 34] learn the latent space of input data through GAN or VAE, and then acquire the ability to generate new data. However, those methods need to use hand model or additionally train GAN and VAE to generate data, which is time consuming and laborious. Like Madadi *et al*. [12], we propose a non-rigid data augmentation by processing the original data. But different from Madadi *et al*. [12], which scale fingers and palm independently and increase the diversity of the size. Our method process the original input point cloud according to label, and directly augments the gesture diversity of datasets depending on kinematics.

3. Method

In this paper, a three-dimensional hand pose estimation network is designed. The network takes the hand point cloud as input and outputs the 3D coordinates of the hand joints, as illustrated in Figure 1. Our network structure is improved on the basis of PointNet++ [21], which is a pioneer to study deep learning on point sets but not suitable for hand joint estimation. Because a lot of work [3, 10, 18, 41] shows that hand pose estimation is a task that pays attention to not only global but also local information. However, the max-pooling in original PointNet++ structure will ignore the local information and produces an ambiguous global feature. This is not wise and will cause waste of features. Therefore, in this paper, we propose the adaptive pooling, which is based on self-attention mechanism and makes the network capable of choosing feature. Further more, for the per-point feature obtained from the last feature extraction layer, we do not directly use the pooling operation to get a global feature, but adopt ensemble strategy to make full use of each feature through learning. In addition, we propose a data augmentation method for point net. By segmenting hand with joints in label and then rotating each part according to the hand kinematics to add gestures, the performance of network in original data can be significantly promoted.

The network architecture of the proposed hand pose estimation method consists of three parts. In section **3.1**, we describe the first part which extracts the coarse features of the hand by proposed "A-PointNet++" structure, whose max-pooling layer is replaced by our proposed adaptive pooling. The second and third part are our ensemble strategy for making full use of local features and introduced in section **3.2**. In Section **3.3**, we will describe the data augmentation approach. Finally, the implementation details are introduced in section **3.4**.

3.1. The extraction of coarse features

In the first part of our network, we use the proposed "A-PointNet++" structure to extract coarse local features from input hand point set. Before entering the network, we need to pre-process the data. We use the same data preprocessing method as Ge *et al*. [6, 9], where a depth image is converted into a partial 3D point cloud with noise and the point set is then downsampled to 1024 points and normalized with oriented bounding box (OBB) [6]. In this paper, the "A-PointNet++" structure is composed by two set abstraction levels. In the first level, each of the 1024 input points passes through the same multi-layer perceptron (MLP) to obtain a per-point feature. Then those per-point features are partitioned into 512 overlapping local regions, and each region contains 64 nearest features to the clustering center. For each local region, we use proposed adaptive pooling instead of the max-pooling to generalize 64 per-point features into a local feature. As shown in Figure 2, adaptive pooling employs the idea of self-attention [33] to give network the ability to choose feature before pooling. In detail, each feature of this region will get an importance score vector through a shared full connection (FC) layer, and then use softmax to normalize the feature along the direction of the dimension to be reduced. Finally, the original features is weighted by these normalized score vectors and synthesized into one feature by the sum operation. This structure gives the network a better ability to combine scattered features into useful features. In this way, the useless features in this local region can be discarded, and the remaining useful and certainly independent features can be combined into a feature which can represent the local region more effectively. After the first level, we got 512 per-point features about local region. The second level is similar to the first level, except that the input has changed from 1024 points to 512 per-point features, and the output is 128 per-point features.

3.2. Hand pose estimation by ensemble learning

After extracting features through "A-PointNet++", 128 per-point features were obtained. A natural approach is to pool these features into a global feature to regress hand

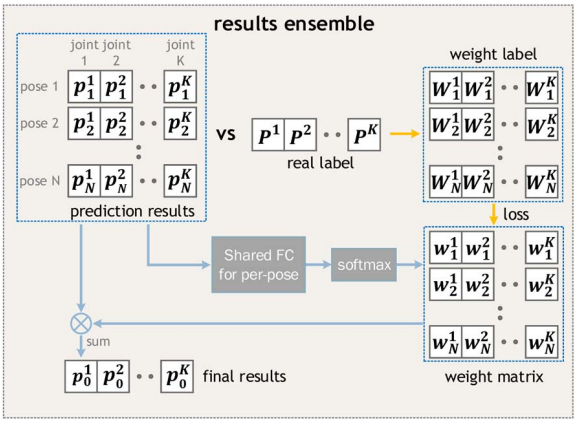


Figure 3: The results ensemble strategy, in which each value of all results is weighted, and each weight is learned by the network itself. Meanwhile, the label of weight matrix is generated online.

joints directly. But we don't think this method can make good use of these features that were painstakingly learned before. Because a lot of work [3, 10, 18, 41] has shown that hand pose estimation is a task that pays attention to not only global but also local information. Here we adopt an ensemble strategy to solve this problem skillfully and easily, which can make better use of local features.

We first randomly divide the 128 features obtained in the first part into n overlapping sets, each with m features, and then pool those m feature of each set to one feature by adaptive pooling, as shown in the second part of our network in Figure 1. In this way, each set of features can maintain a certain degree of globality. What needs to be explained is that point cloud are actually unordered, so randomly partition can be approximated by sliding in sequence as shown in Figure 1, and this also ensures that all 128 features are used.

The third part of our network is the proposed ensemble strategy. After the second part, we can obtain n sets of hand features, each predicts a hand pose including the coordinates of $K \times 3$ joint points, where K represents the number of joints. The labelled joint coordinates of the hand are used to supervise every pose estimation as follows

$$L_1 = \sum_{n=1}^{N} \sum_{k=1}^{K} \left\| P^k - p_n^k \right\|^2 \tag{1}$$

where P^k and p_n^k respectively correspond to the GT and predicted 3D joint coordinates. We believe that such n different groups of hand features will predict n different hand poses, which can complement each other. Inspired by the ensemble learning, we propose to improve the prediction accuracy by integrating these poses. This is a more reasonable strategy for pose estimation which reduce the difficulty of one-time global hand pose estimation. By superimposing these n results in proportion to their importance, the coordinates of all the joints of the hand are obtained.

In order to measure the importance of each hand pose, we generate a $N \times K$ weight matrix where N represents the number of poses and K represents the number of joints per pose, as shown in Figure 3. Any element $\mathrm{w}_\mathrm{n}^\mathrm{k}$ in the weight matrix represents the influence or importance of the same joint in each hand pose on the final ensemble result. The greater the weight parameter $\mathrm{w}_\mathrm{n}^\mathrm{k}$, the greater the contribution to the final results, and vice versa. The weight matrix can be learned by the network itself during training process. It is supervised by an online generated ground truth matrix where any element W_n^k is denoted as:

$$W_n^k = \frac{\ln Z_n^k}{\sum_{n=1}^{N} \ln Z_n^k} \tag{2}$$

where we define z_n^k as:

$$z_n^k = \left\| P^k - p_n^k \right\|_2 \tag{3}$$

In order to ensure the stability of formula (2), we set a threshold ε. When z_n^k is the minimum and less than ε, W_n^k is set to 1, and the rest is 0. The $N \times K$ weight matrix is trained end-to-end to minimize the following cost

$$L_2 = \sum_{k=1}^{K} \left\| P^k - p_0^k \right\|^2 + \sum_{n=1}^{N} \sum_{k=1}^{K} \left\| W_n^k - \mathrm{w}_\mathrm{n}^\mathrm{k} \right\|^2 \tag{4}$$

where the second part of the cost adds another constraint and leads the entire network to converge to the hand pose as accurate as possible.

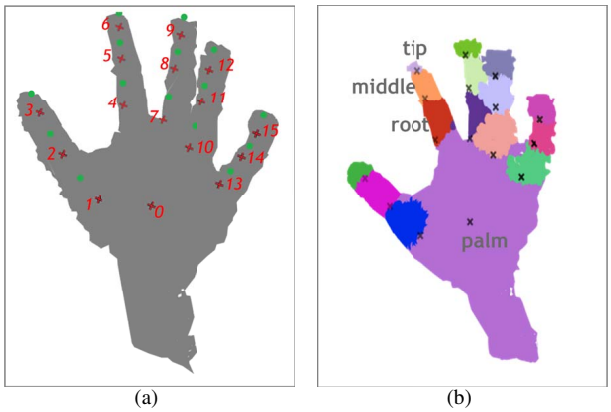
3.3. Kinematic data augmentation with label

In this paper, we propose an effective data augmentation method. Different from simple rotation, translation and scaling, our method increases the richness of gestures complied with hand kinematics constraints. Firstly, we select 16 joints in label to calculate the centroids of clustering, as the red 'X' shown in Figure 4(a). Then we divide the hand into 16 parts, as shown in Figure 4(b), according to Euler distance and rotate each part according to hand kinematics to expand the gesture.

Figure 5 shows some examples of data augmentation, where (a) is the original point cloud and (b) is the augmented data. Although these new data do not conform to the distribution of the original data obtained from the sensor, these points after rotated are actually on the hand surface, in other words, the new data is still a sampling of the complete hand surface. Therefore, when new data and original data are input into the network together, this augmentation can help the point net learn the relationship more clearly between joints and points of the hand surface.

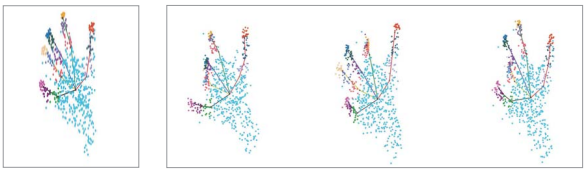
3.4. Implementation details

We use Adam [14] optimizer with initial learning rate 0.001, momentum 0.9, batch size 35 and regularization strength 0.00001 to train our network. The learning rate



(a) (b)

Figure 4: (a) We select 16 joints in label as the red 'X', and calculate the clustering center as the green point. (b) Hand point cloud segmentation using k-means.



(a) (b)

Figure 5: (a) The original hand point cloud. (b) Hand point cloud after changing gesture.

is divided by 10 after 30 epochs, and we stop the training process after 60 epochs.

Our experimental platform is Pytorch with Intel Core i7 7700K, 32GB of RAM and an Nvidia GTX 1080 GPU.

4. Experiment and Result

We evaluate our proposed method on three public hand pose datasets: ICVL [29], NYU [30], and MSRA [27].

The ICVL dataset [29] contains $22K$ training frames from 10 different subjects and 1596 testing frames with two sequences of 702 and 894. The ground truth of each frame contains $M = 16$ joints.

The NYU dataset [30] contains 72757 training frames from one subject and 8252 testing frames from two subjects. The annotation of each frame is 36 joints and we estimate a subset of $M = 14$ joints following previous works [8, 18, 30].

The MSRA dataset [27] contains 76500 frames from 9 subjects and each subject contains 17 gestures. The annotation of each frame is $M = 21$ joints. For evaluation, we used the nine-fold cross-validation method.

We use two metrics to evaluate the hand pose estimation performance: the first metric is the joint mean error distance over all test frames; the second metric is the proportion of good frames in which the worst joint error is below a threshold.

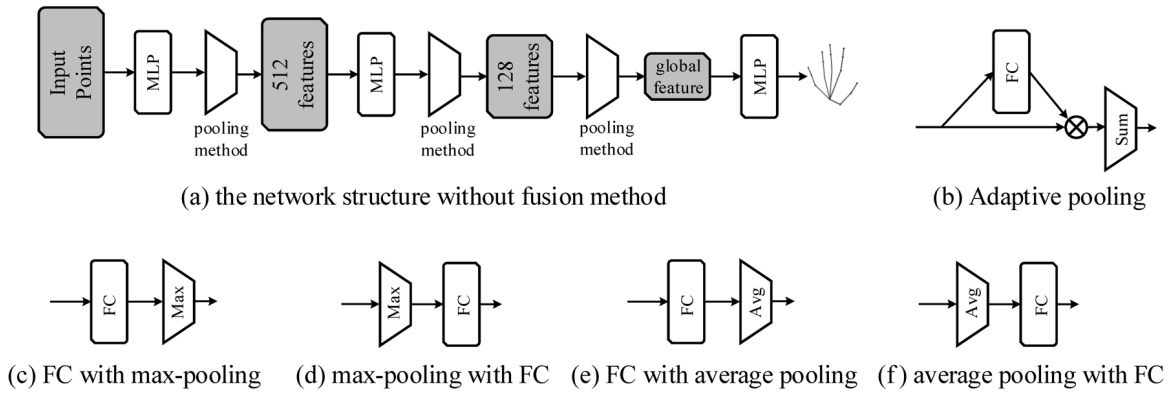
framework	NYU	ICVL
Max	12.21mm	8.12mm
Avg	12.89mm	8.03mm
FC+Max	12.09mm	8.02mm
Max+FC	12.17mm	8.07mm
FC+Avg	12.56mm	7.86mm
Avg+FC	12.71mm	7.94mm
adaptive pooling	11.36mm	7.62mm
Group+ average	11.21mm	7.50mm
our	10.33mm	6.88mm

Table 1: Comparison of average joint error between different framework for ablation experiments

4.1. Effect of adaptive pooling

As mentioned in Section **3.1**, in order to better retain the effective information of the original features after pooling, we use adaptive pooling instead of maximum pooling. In this subsection, we will demonstrate the effectiveness of this method through experiments on NYU [30] and ICVL [29] datasets.

We use the network structure shown in Figure 6(a) to compare the pooling methods, which is the backbone of our overall network, in other words, the overall network removal ensemble method. There are three parts for pooling, which can be replaced by different pooling methods in the experiment. Our baseline use max-pooling as same as PointNet++ [21], and we also compare with average pooling. In addition, since adaptive pooling (Ada), as shown in Figure 6(b), has more network parameters than max-pooling, we also compare four pooling methods 'FC+Max', 'Max+FC', 'FC+Avg', 'Avg+FC' as shown in Figure 6(c)-(f) in order to compare fairly with the same parameter. The results of comparison are shown in Table 1. As we can see, the average joint error of Ada is 11.36mm on NYU, which is 0.85mm less than 'Max', 1.53mm than 'Avg', and 0.73mm, 0.81mm, 1.2mm and 1.35mm than 'FC+Max', 'Max+FC', 'FC+Avg', 'Avg+FC'. And on ICVL, it is 0.5mm, 0.41mm, 0.4mm, 0.45mm, 0.24mm, 0.32mm less than the other six frameworks. Hence, this effectively proves that the feature pooled by proposed adaptive pooling can better retain hand information. In addition, it can be found that max-pooling is better than average pooling on NYU, while the opposite is true on ICVL, so it can be seen that the choice of pooling methods has a direct impact on the result.



(a) the network structure without fusion method (b) Adaptive pooling

(c) FC with max-pooling (d) max-pooling with FC (e) FC with average pooling (f) average pooling with FC

Figure 6: The framework for verifying the effectiveness of adaptive pooling.

Methods	Original	Gestures $\times 5$	Gesture $\times 20$	Scale $\times 5$	Scale $\times 20$
Ada	7.6mm	7.2mm	6.7mm	7.4mm	7.2mm
Group+weight	6.9mm	-	6.24mm	-	-

Table 2: Effects of our data extension method on ICVL dataset [29]. "Gestures" means our data augmentation method and "Scale" comes from [6]. "$\times 5$" means augmenting the dataset five times, and so on.

4.2. Effect of ensemble strategy

As mentioned in Section **3.2**, we propose an ensemble strategy where a weight matrix is learned to guide results fusion. In order to evaluate the impact of different ensemble strategy, we compare the result of two model, one is our overall network (Group+weight), which uses the ensemble strategy mentioned above to obtain the final prediction result, and the other is Group+average, where the multiple hand poses are averaged as a final result. As can be seen in Table 1, the Group+weight can achieve superior performance on NYU [30] and ICVL [29] datasets. Specifically, on NYU dataset, the average joint error of our overall network is 10.33mm, which obviously lower than the Group+average (our), which is 11.21mm. On ICVL dataset, the average joint error of Group+weight is reduced by 0.62mm. This proves the weight matrix learned by network itself can effectively guide the ensemble of results, so as to get more accurate results.

4.3. Effect of data augmentation

We verify the effect of data augmentation on ICVL dataset [29]. We augment the data offline by 20 times with randomly bending fingers, and then we train two models with this augmented dataset. One is the PointNet++ whose max-pooling is replaced by the adaptive pooling (Ada). The other is our overall network (Group+weight). As can be seen from Table 2, when augmenting the data set by 20 times, the test error of Ada can be reduced by 0.9mm compared with that without data augmentation, and the test error of Group+weight can be reduced by 0.66mm. Through this experiment, it can be proved that it is very effective for improving the performance of the network by using our proposed gesture extension method.

In addition, we also compare the data augmentation method with Ge *et al*. [6], which increases the diversity of scales and augments the data by five times on ICVL dataset [29]. Hence, we also augment ICVL dataset [29] by five times to keep the same experimental conditions, and use these two datasets to train Ada. As can be seen from Table 2, by comparing columns 2 to 4, it can be found that the average joint point error decreases significantly with the increasing amount of data. From the 4 and last columns, we can see the average joint error of our augmentation method is 0.2mm less than that of the method in Ge *et al*. [6]. Hence, it proves that our method that increases the diversity of gestures is more effective than the method of increasing diversity of scales. Therefore, we believe that the high freedom of gesture space is more important for limiting the accuracy of hand pose estimation.

4.4. Comparisons with state-of-the-arts

We compare our method with latent random forest(LRF) [29], 2D CNN in hand model (DeepModel) [40], feedback loop based on 2D CNN (Feedback Loop) [18], 2D CNN with priors (DeepPrior) [17] and its evolution (DeepPrior++) [15], Lie group based 2D CNN (Lie-X) [36], multi-view 3D CNN (3DCNN) [8], region ensemble network (REN) [10], pose guide structured REN (Pose-REN) [3]dense 3D regression (DenseReg) [35], voxel-to-voxel (V2V) [13], SHPR-net [4], hand regression with hierarchical PointNet (HandPointNet) [6], and its improved version Point-to-Point [9]. The proportion of good frames over different error thresholds and the per-joint mean error distance of different methods on ICVL [29], NYU [30] and

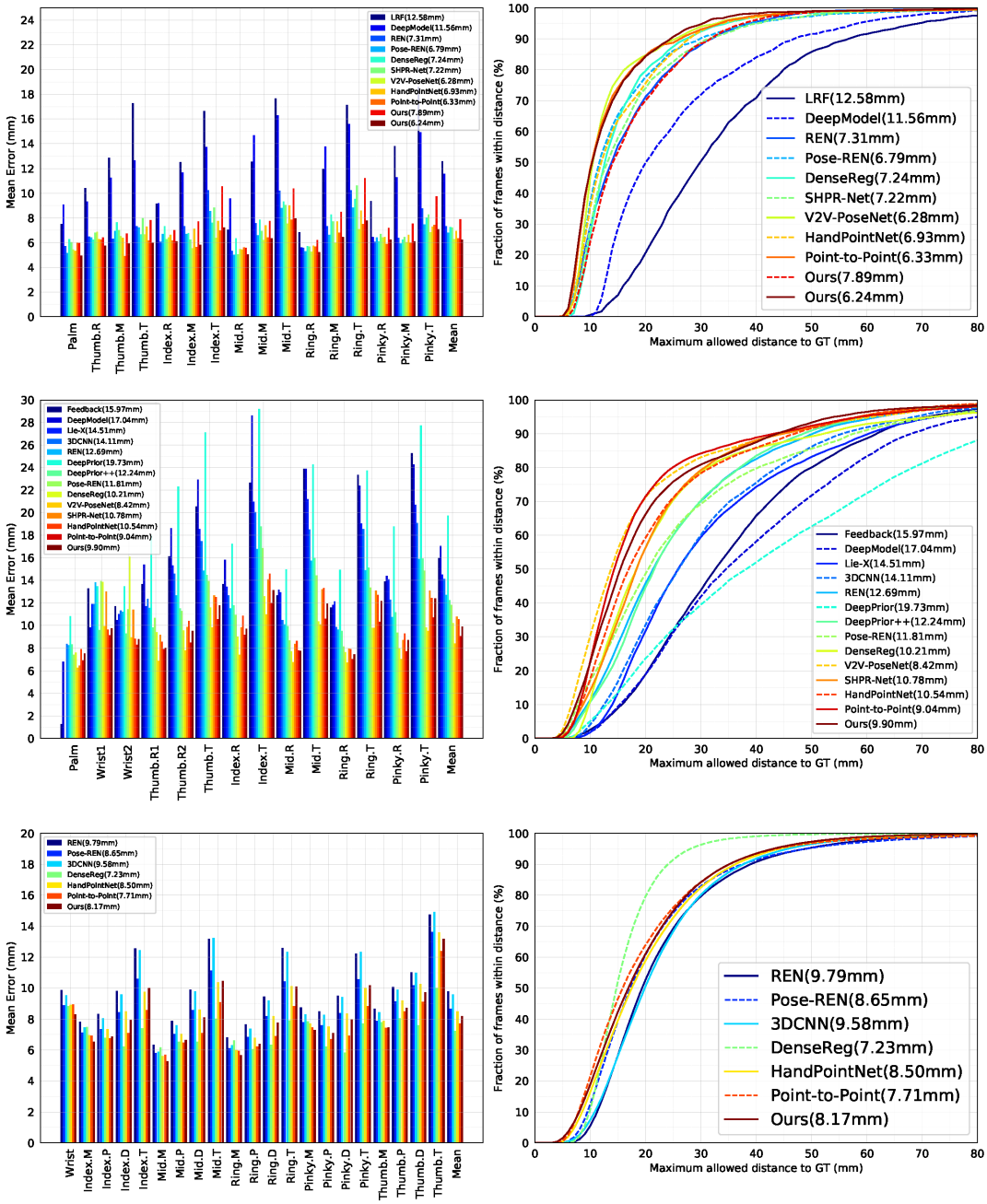


Figure 7: Comparison of our proposed method with state-of-the-arts methods. Left: the percentage of success frames over different error thresholds. Right: 3D distance errors per hand keypoints. Top row: ICVL dataset, Middle row: NYU dataset, bottom row: MSRA dataset

MSRA [27] datasets are presented in Figure 7.

On ICVL dataset [29], to evaluate the capability of our methods, we augment the data offline by 20 times, totaling $440K$ point clouds. As shown in the top of Figure 7, our method demonstrate good performance compared with state-of-the-art methods over most of the error thresholds. Specifically, the average joint error of our method is slightly lower than V2V [13] and Point-to-Point [9]. This proves that our method is effective for hand pose estimation tasks.

On NYU dataset [30], in order to fairly compare our method with HandPointNet [6] and Point-to-Point [9], we double the data as they did. As shown in the middle of Figure 7, our method achieve 9.90mm in average joint error, which shows comparable performance with the two best methods V2V [13] and Point-to-Point [9]. And compared with HandPointNet [6], our method can reduce the average

Model name	HandPointNet	Point-to-Point	V2V	Ours
Parameters	10.3M	17.2M	457.4M	11.6M

Table 3: Comparison of parameters between different models.

joint error by 0.64mm, and the proportion of good frames of our method is better than HandPointNet [6] over almost all the error thresholds. Those demonstrate that our method can achieve comparable performance with state-of-the-arts.

On MSRA dataset [27], we mainly compare the effect of our network structure with other methods. As shown in the bottom of Figure 7, our network can still achieve satisfactory results without augmenting data. Specifically, compared with HandPointNet [6], which directly use the Pointnet++ structure and then adjusts the position of each fingertip joint, our approach shows better performance. This proves that our network can extract fine features from point clouds and make full use of them to hand pose estimation tasks.

Based on the above performance, we think that our network structure makes more effective use of the 3D information contained in the hand point cloud. At the same time, our data augmentation method can significantly improve the performance of 3D hand pose estimation method based on point cloud.

4.5. Computational complexity

The runtime of our method is 14.5ms in average, including 4.2ms for point sampling, 10.3ms for the hand pose regression network to predict result. Thus, our method runs in real-time at over 69.0fps. In addition, our network model size is 11.6MB, and the comparison with some related work is shown in Table 3.

5. Conclusion

We propose a hand pose ensemble learning approach based on grouping features of hand point sets. Every group of features is used to predict a hand pose and all groups of features are then integrated to improve the prediction accuracy. We segment hand with joints in label and augment the datasets by adding gestures to increase the diversity of the datasets. The two ideas boost the performance significantly and make the our model achieve comparable performance with state-of-the-arts.

References

[1] Seungryul Baek, Kwang In Kim, and Tae-Kyun Kim. Augmented skeleton space transfer for depth-based hand pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8330–8339, 2018.

[2] Yujun Cai, Liuhao Ge, Jianfei Cai, and Junsong Yuan. Weakly-supervised 3d hand pose estimation from monocular rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 666–682, 2018.

[3] Xinghao Chen, Guijin Wang, Hengkai Guo, and Cairong Zhang. Pose guided structured region ensemble network for cascaded hand pose estimation, 2017.

[4] Xinghao Chen, Guijin Wang, Cairong Zhang, Tae-Kyun Kim, and Xiangyang Ji. Shpr-net: Deep semantic hand pose regression from point clouds. *IEEE Access*, 6:43425–43439, 2018.

[5] Ali Erol, George Bebis, Mircea Nicolescu, Richard D Boyle, and Xander Twombly. Vision-based hand pose estimation: A review. *Computer Vision and Image Understanding*, 108(1-2):52–73, 2007.

[6] Liuhao Ge, Yujun Cai, Junwu Weng, and Junsong Yuan. Hand pointnet: 3d hand pose estimation using point sets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8417–8426, 2018.

[7] Liuhao Ge, Hui Liang, Junsong Yuan, and Daniel Thalmann. Robust 3d hand pose estimation in single depth images: from single-view cnn to multi-view cnns. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3593–3601, 2016.

[8] Liuhao Ge, Hui Liang, Junsong Yuan, and Daniel Thalmann. 3d convolutional neural networks for efficient and robust hand pose estimation from single depth images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1991–2000, 2017.

[9] Liuhao Ge, Zhou Ren, and Junsong Yuan. Point-to-point regression pointnet for 3d hand pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 475–491, 2018.

[10] Hengkai Guo, Guijin Wang, Xinghao Chen, Cairong Zhang, Fei Qiao, and Huazhong Yang. Region ensemble network: Improving convolutional network for hand pose estimation. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 4512–4516. IEEE, 2017.

[11] Umar Iqbal, Pavlo Molchanov, Thomas Breuel Juergen Gall, and Jan Kautz. Hand pose estimation vi4 latent 2.5 d heatmap regression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 118–134, 2018.

[12] Meysam Madadi, Sergio Escalera, Xavier Baró, and Jordi Gonzalez. End-to-end global to local cnn learning for hand pose recovery in depth data. *arXiv preprint arXiv:1705.09606*, 2017.

[13] Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee. V2v-posenet: Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation from a single depth map. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5079–5088, 2018.

[14] Franziska Mueller, Florian Bernard, Oleksandr Sotnychenko, Dushyant Mehta, Srinath Sridhar, Dan Casas, and Christian Theobalt. Ganerated hands for real-time 3d hand tracking from monocular rgb. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 49–59, 2018.

[15] Markus Oberweger and Vincent Lepetit. Deepprior++: Improving fast and accurate 3d hand pose estimation. In *Pro-*

ceedings of the IEEE International Conference on Computer Vision, pages 585–594, 2017.

[16] Markus Oberweger, Gernot Riegler, Paul Wohlhart, and Vincent Lepetit. Efficiently creating 3d training data for fine hand pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4957–4965, 2016.

[17] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Hands deep in deep learning for hand pose estimation. *arXiv preprint arXiv:1502.06807*, 2015.

[18] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Training a feedback loop for hand pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3316–3324, 2015.

[19] Paschalis Panteleris and Antonis Argyros. Back to rgb: 3d tracking of hands and hand-object interactions based on short-baseline stereo. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 575–584, 2017.

[20] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[21] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017.

[22] Mahdi Rad, Markus Oberweger, and Vincent Lepetit. Feature mapping for learning fast and accurate 3d pose inference from synthetic images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4663–4672, 2018.

[23] Toby Sharp, Cem Keskin, Duncan Robertson, Jonathan Taylor, Jamie Shotton, David Kim, Christoph Rhemann, Ido Leichter, Alon Vinnikov, Yichen Wei, et al. Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 3633–3642. ACM, 2015.

[24] Ayan Sinha, Chiho Choi, and Karthik Ramani. Deephand: Robust hand pose estimation by completing a matrix imputed with deep features. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4150–4158, 2016.

[25] Adrian Spurr, Jie Song, Seonwook Park, and Otmar Hilliges. Cross-modal deep variational hand pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 89–98, 2018.

[26] Srinath Sridhar, Franziska Mueller, Antti Oulasvirta, and Christian Theobalt. Fast and robust hand tracking using detection-guided optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3221, 2015.

[27] Xiao Sun, Yichen Wei, Shuang Liang, Xiaoou Tang, and Jian Sun. Cascaded hand pose regression. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 824–832, 2015.

[28] James S Supancic, Grégory Rogez, Yi Yang, Jamie Shotton, and Deva Ramanan. Depth-based hand pose estimation: data, methods, and challenges. In *Proceedings of the IEEE international conference on computer vision*, pages 1868–1876, 2015.

[29] Danhang Tang, Hyung Jin Chang, Alykhan Tejani, and Tae-Kyun Kim. Latent regression forest: Structured estimation of 3d articulated hand posture. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3786–3793, 2014.

[30] Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Transactions on Graphics (ToG)*, 33(5):169, 2014.

[31] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in neural information processing systems*, pages 1799–1807, 2014.

[32] Aggeliki Tsoli and Antonis A Argyros. Joint 3d tracking of a deformable object in interaction with a hand. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 484–500, 2018.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[34] Chengde Wan, Thomas Probst, Luc Van Gool, and Angela Yao. Crossing nets: Dual generative models with a shared latent space for hand pose estimation. In *Conference on Computer Vision and Pattern Recognition*, volume 7, 2017.

[35] Chengde Wan, Thomas Probst, Luc Van Gool, and Angela Yao. Dense 3d regression for hand pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5147–5156, 2018.

[36] Chi Xu, Lakshmi Narasimhan Govindarajan, Yu Zhang, and Li Cheng. Lie-x: Depth image based articulated object pose estimation, tracking, and action recognition on lie groups. *International Journal of Computer Vision*, 123(3):454–478, 2017.

[37] Qi Ye, Shanxin Yuan, and Tae-Kyun Kim. Spatial attention deep net with partial pso for hierarchical hybrid hand pose estimation. In *European conference on computer vision*, pages 346–361. Springer, 2016.

[38] Shanxin Yuan, Guillermo Garcia-Hernando, Björn Stenger, Gyeongsik Moon, Ju Yong Chang, Kyoung Mu Lee, Pavlo Molchanov, Jan Kautz, Sina Honari, Liuhao Ge, et al. Depth-based 3d hand pose estimation: From current achievements to future goals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2636–2645, 2018.

[39] Shanxin Yuan, Bjorn Stenger, and Tae-Kyun Kim. Rgb-based 3d hand pose estimation via privileged learning with depth images. *arXiv preprint arXiv:1811.07376*, 2018.

[40] Xingyi Zhou, Qingfu Wan, Wei Zhang, Xiangyang Xue, and Yichen Wei. Model-based deep hand pose estimation. *arXiv preprint arXiv:1606.06854*, 2016.

[41] Yidan Zhou, Jian Lu, Kuo Du, Xiangbo Lin, Yi Sun, and Xiaohong Ma. Hbe: Hand branch ensemble network for real-time 3d hand pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 501–516, 2018.