SO-HandNet: Self-Organizing Network for 3D Hand Pose Estimation with Semi-supervised Learning

Yujin Chen[1], Zhigang Tu[1*], Liuhao Ge[2], Dejun Zhang[4], Ruizhi Chen[1], Junsong Yuan[3]
[1]Wuhan University, Wuhan, China
[2]Nanyang Technological University, Singapore
[3]State University of New York at Buffalo, Buffalo, NY, USA
[4]China University of Geosciences, Wuhan, China
{yujin.chen, tuzhigang, ruizhi.chen}@whu.edu.cn ge0001ao@e.ntu.edu.sg
zhangdejun@cug.edu.cn jsyuan@buffalo.edu

Abstract

3D hand pose estimation has made significant progress recently, where Convolutional Neural Networks (CNNs) play a critical role. However, most of the existing CNN-based hand pose estimation methods depend much on the training set, while labeling 3D hand pose on training data is laborious and time-consuming. Inspired by the point cloud autoencoder presented in self-organizing network (SO-Net) [18], our proposed SO-HandNet aims at making use of the unannotated data to obtain accurate 3D hand pose estimation in a semi-supervised manner. We exploit hand feature encoder (HFE) to extract multi-level features from hand point cloud and then fuse them to regress 3D hand pose by a hand pose estimator (HPE). We design a hand feature decoder (HFD) to recover the input point cloud from the encoded feature. Since the HFE and the HFD can be trained without 3D hand pose annotation, the proposed method is able to make the best of unannotated data during the training phase. Experiments on four challenging benchmark datasets validate that our proposed SO-HandNet can achieve superior performance for 3D hand pose estimation via semi-supervised learning.

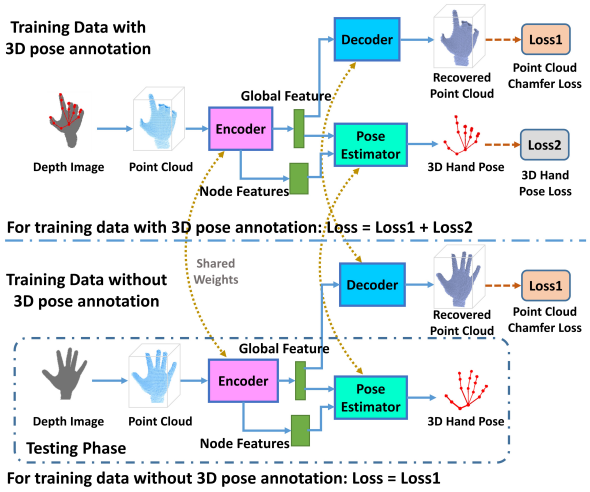

Figure 1. Overview of our proposed SO-HandNet for 3D hand pose estimation. In the training phase, we simultaneously use the annotated data and unannotated data to train the model. For training data with 3D hand annotation, both 3D hand pose and a new point cloud are predicted from encoded features. And the network is optimized by point cloud Chamfer loss and hand pose loss. For training data without 3D pose annotation, we use the point cloud Chamfer loss to optimize the HFE and HFD. In the testing phase, the HFE and the HPE are utilized to estimate 3D hand pose.

1. Introduction

Hands are critical parts of human body in our daily activities. Automatic real-time 3D hand pose estimation has attracted a lot of attentions due to its wide applications such as human-computer interaction (HCI), computer graphics and virtual/augmented reality, etc.

After many years of intensive research, 3D hand pose estimation has advanced significantly both in accuracy and efficiency [5, 7, 10, 11, 12, 13, 21, 22, 23, 24, 29, 33, 35, 36, 39, 41]. Most of the proposed state-of-the-art 3D hand pose estimation methods are based on convolutional neural networks (CNNs). Since CNNs perform well in processing images, many works modify 2D CNNs to deal with depth images [36] or their 2D projections[11]. However, features extracted by 2D CNNs are not directly suitable for 3D pose estimation due to the lack of 3D spatial information. To better capture the geometric characteristics of depth data, recent

*Corresponding author: tuzhigang@whu.edu.cn

2380-7504/19/$31.00 ©2019 IEEE
DOI 10.1109/ICCV.2019.00706

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 15:31:10 UTC from IEEE Xplore. Restrictions apply.

studies [12, 21] convert depth images to 3D volumetric representations and then use 3D CNNs to estimate hand pose. However, the 3D volumes have rather larger memory and computation requirements [37]. Although these methods have achieved significant progress in estimation accuracy, they usually require large amount of fully annotated training data which are difficult to obtain. Only few methods [3, 25, 31, 38] have considered to use unannotated data for training hand pose estimation networks. [31, 38] use generative deep neural network with a shared latent space to learn hand model. [38] attempts to learn a manifold of hand poses via a combination of variational autoencoder and generative asversarial networks. However, their method requires a mapping function between two separate mainfolds which makes the network difficult to train. [3] leverages synthesized data to enrich existing datasets, but synthetic depth maps are different from real-world data. [31] proposes to learn a single laten space from images, which cannot fully utilize 3D information in depth images.

To tackle these problems, motivated by the recent work of SO-Net [18] which utilizes the spatial distribution to perform hierarchical feature extraction for point cloud and proposes a point cloud autoencoder as pre-training to improve network performance, we aim at regressing 3D hand pose directly from 3D point cloud and using point cloud autoencoder mechanism in semi-supervised training stage. [10] is the first work to regress hand pose directly from 3D point cloud. Compared with PointNet++ [28] in [10], the self-organizing network in our HFE perfoms hierarchical feature extraction with systematically adjust the receptive field overlap. Accordingly, the feature encoder of our method is able to reveal the spatial distribution of the input point cloud. Most importantly, we apply an autoencoder structure whose decoder recovers point cloud from the global representation of point set. To learn a more discriminative global feature, we compare the recovered point cloud with the original point cloud. Therefore, we are able to combine annotated data with unannotated date to train the network. The idea of autoencoder has been recently applied to the hand pose estimation task [31]. Different from our method, they directly deal with RGB image or depth image.

As illustrated in Figure 1, we propose an end-to-end regression method for 3D hand pose estimation from a single depth image. The depth image is first converted into a set of 3D points. Then the point set is sampled and normalized before fed into the network. An encoder is utilized to encode the input point cloud into a global feature through a hierarchical extraction mechanism, and intermediate features which are called node features are also collected. Both the global feature and node features are used for the 3D hand pose estimation. Additionally, in the training phase, the obtained global feature can reconstruct a point cloud to compare with the original input point cloud. Thereby, we optimize the representation ability of the global feature by minimizing the gap between the reconstructed and original point cloud. The advantage of the proposed method is clear when applied to the case that part of the training set is annotated and the rest is unannotated. In this condition, we use the labeled data to train the whole network and the unlabeled data to help train the encoder and decoder. In summary, our method has the following contributions:

- We propose to estimate 3D hand pose directly from 3D point cloud with semi-supervised learning. We design a semi-supervised training strategy which uses few annotated data to train the whole pipeline and makes full use of unannotated data to optimize the network.
- For 3D hand pose estimation, a novel point cloud encoder-decoder mechanism is presented to extract and evaluate features. The self-organizing encoder models the spatial distribution of point cloud by hierarchically extracting features guided by a self-organized map. The decoder reconstructs hand point cloud from the encoded global feature, which helps to learn the point cloud encoder.
- We conduct comprehensive experiments on four hand pose estimation datasets. Experimental results show that our proposed SO-HandNet performs better than recent semi-supervised methods. Besides, it outperforms or is comparable with state-of-the-art fully-supervised methods.

2. Related work

Hand Pose Estimation. The field of depth-based hand pose estimation has become attractive thanks to the significant advance and progress of cost-effective depth sensors, such as Microsoft Kinect [46] and Intel RealSense [15]. Methods of depth-based hand pose estimation can be categorized into generative approaches, discriminative approaches and hybrid approaches. Comprehensive review of hand pose estimation can be found in [34, 43]. Our 3D hand pose estimation method is related to discriminative approaches with application of deep neural networks. Tompson *et al*. [36] firstly apply CNNs in hand pose estimation task. They train CNNs to output heat-map images and then infer the corresponding 3D hand pose. However, 3D spatial information is lost in 2D heat-maps. Ge *et al*. [11] address this issue by projecting the depth images onto multi-views and then recovering 3D coordinates from multiple heat-maps. Ge *et al*. [12] encode point cloud as 3D volumetric representation of hand and use 3D CNNs to directly regress 3D hand pose. Ge *et al*. [10] propose a Hand-PointNet to directly process the 3D point cloud for hand pose estimation, and design a fingertip refinement network to refine the fingertip location. Moon *et al*. [21] exploit voxel-to-voxel predictions that use a 3D voxelized grid and estimate the per-voxel likelihood for each keypoint. Additionally, there are methods combine semantic segmentation [6], augment data in the skeleton space [3] and guide learn-

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 15:31:10 UTC from IEEE Xplore. Restrictions apply.

ing by synthetic images [29]. Recently, semi-supervised learning has been employed to hand pose estimation task. Wan *et al.* [38] use two deep generative models with a shared latent space to model the statistical relationships of depth images and corresponding hand poses. They design an architecture which learns from unlabeled data in a semi-supervised approach. Spurr *et al*. [31] propose to learn hand model by a cross-modal trained latent space via a gennerative deep neural network. They also make use of unlabeled data via cross-training. [25] uses labeled synthetic and unlabeled real data to learn the feature mapping between real data and synthetic data. Our method is inspired by [10], but is essentially different from it. Multi-scale and multi-resolution grouping are employed to combine features from multiple scales. Besides, our feature encoder can explicitly models the spatial distribution of the input point set. Moreover, the proposed method designs a semi-supervised training mode for specific case.

3D Deep Learning. 3D deep learning is rising along with big 3D datasets such as ShapeNet [4] and ModelNet [40]. 3D data can be represented as rasterized form (e.g., multi-view images and volumetric) or geometric form (e.g., polygonal mesh, point cloud and primitive-based CAD models), and there are deep learning methods [14, 16, 20, 26, 27, 28, 30, 32, 40, 42] to process them. Our work is closely related to methods that directly take point cloud as input. Qi *et al*. [26] present PointNet which is the pioneer in directly processing point cloud by deep learning. They use symmetric max pooling to aggregate local point features into a global descriptor which is invariant to the permutation of the input points. Later, they design PointNet++ [28] to group points into several groups in different levels to hierarchically extract feature from different scales. By combining and modifying the prior work PointNet [26] and NetVLAD [2], Uy *et al*. exploit a PointNetVLAD method [1]. This deep network allows end-to-end training and inference to extract the global descriptor from the input 3D point set. Li *et al*. [19] present PointCNN which uses typical CNNs to learn features from point cloud. In SO-Net [18], Li *et al*. present a permutation invariant self-organizing network (SO-Net) which explicitly models the spatial distribution of input point cloud during feature extraction. The receptive field of the network can be systematically adjusted by conducting point-to-node k nearest neighbor search. In this paper, we use an architecture like SO-Net to perform hierarchical feature extraction from input point cloud. More details about SO-Net is given in Section 3.1.

3. Proposed Method

Similar to [10], our hand pose estimation approach is a regression-based method. Generally, the hand pose regression method takes a depth image containing a hand as input, and outputs the estimated locations of the 3D hand joints in the camera coordinate system (C.S.). The hand depth image is converted into a set of 3D points, and the points are sampled and normalized before inputted to the network. In this work, we display a pipeline which consists of three parts. The HFE is exploited to perform hierarchical feature extraction, and then the HPE is used to fuse multi-level features to regress the 3D hand pose. In addition, we utilize the HFD to optimize the feature encoding procedure in the training phase.

In the following, we first briefly introduce the pipeline of point cloud preprocessing and review the mechanism of the SO-Net which is designed for point cloud analysis, then describe our proposed hand pose estimation method.

3.1. SO-Net Revisited

SO-Net [18] is a type of permutation invariant architecture for deep learning with unordered point clouds. The network models the spatial distribution of a set of points by building a self-organizing map (SOM) [17], and then performs hierarchical feature extraction on individual points and SOM nodes, finally produces a discriminative feature of the input point cloud. As shown in Figure 2, a SOM with size of $M = m \times m$ is built to produce two-dimensional representation of the input N points. The SOM is constructed via unsupervised competitive learning approach. Given the output of the SOM, a point-to-node k nearest neighbors (kNN) search is conducted. In this process, kNN are searched on the SOM nodes S for each point p_i. By subtraction with the associated nodes, each p_i is normalized into k points, accordingly the point cloud is transformed into kN normalized points. A series of fully connected layers is employed to extract individual point features. Following the above kNN association, a channel-wise max pooling operation is conducted to get the node feature from the point features associated with the corresponding SOM node. Then the M node features are forward into a series of shared layers, and aggregated into a global feature that represents the input point cloud. Compared with PointNet++ [28] which handles the point cloud by grouping strategy, SO-Net utilizes an efficient separate-and-assemble

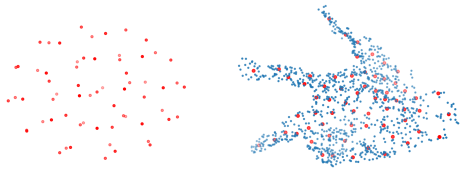

Figure 2. **Left:** The initial nodes of an 8×8 self-organizing map (SOM). **Right:** Example of a SOM training result. After an unsupervised competitive learning procedure, the nodes fit well with the input point cloud.

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 15:31:10 UTC from IEEE Xplore. Restrictions apply.

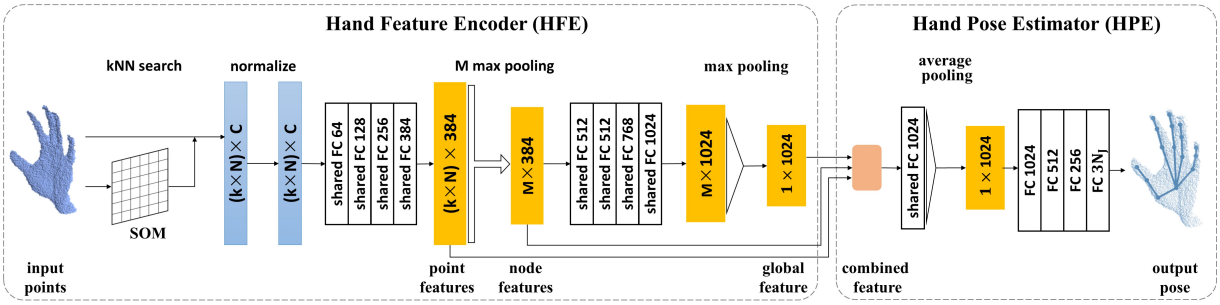

Figure 3. The architecture of our proposed SO-HandNet. In the HFE, input points $N \times C$ are normalized with the k-nearest SOM node. After a series of shared FC layers, normalized points are transferred into point features and later max-pooled into node features, these node features are aggregated into a global representation accordingly. In the HPE, multi-level features extracted by the HFE are combined and then forward into FC layers to regress the output pose. In this figure, N refers to the point number, C refers to the input channel (C equals to 3 when only inputs points and equals to 6 while the surface normal of each point is also applied), k refers to the k neighbors in kNN search, M refers to the node number, N_J refers to the joint number.

approach as the SOM is able to explicate the spatial distribution of points.

3.2. Point Cloud Processing

The hand depth image is first converted into a set of 3D points according to the intrinsic parameters of the depth camera. In order to promote computational efficiency, the 3D point set is sampled to N points. In our implementation, we set the number of sampled points N as 1024, and transform and normalize the sampled 3D point set into an oriented bound box coordinate system (OBB C.S) [10]. The original hand point cloud may have multiple orientations in camera C.S., but the orientation of point cloud is more consistent after normalized to OBB C.S. Experiments in [10] show that the OBB-based point cloud normalization can improve the performance of hierarchical point feature extraction.

3.3. Hand Pose Regression Network

We design an end-to-end trainable network for 3D hand pose estimation. The pose regression problem inputs a set of normalized points $X = \{x_i\}_{i=1}^{N} = \{(p_i, n_i)\}_{i=1}^{N}$ and outputs estimated pose $\hat{P} = \{pose_i\}_{i=1}^{N_J} = \{(x_i, y_i, z_i)\}_{i=1}^{N_J}$ with N_J hand joints of three dimensions, where p_i is the 3D coordinate of the point and n_i is the 3D surface normal. A regression function f_r is given by the following equation:

$$\hat{P} = f_r(X, \theta_r), \tag{1}$$

where θ_r is the trainable parameters of regression function f_r. In our method, we apply deep CNNs to optimize the parameters θ_r in order to minimize the gap between the estimated hand pose $\hat{P}$ and the ground truth hand pose P.

Hand Feature Encoder (HFE). Our HFE f_{HFE} hierarchically extracts multi-level features from the input point cloud. As shown in Figure 3, we reconstruct the encoder of SO-Net [18] to process our hand point cloud. With the guidance of SOM, the encoder is able to capture features hierarchically and output multi-level features including point features, node features and a global feature. The input of HFE can be only the normalized point coordinates or the combination of coordinates and surface normal vectors. With the application of kNN search which is guided by SOM, the input can be converted into kN normalized points, and then a series of shared fully connected layers are utilized to extract individual point features. The resulting point features are fed into a channel-wise max pooling to get the node features. Accordingly, the node features are forward to a series of shared layers and aggregated to a global vector which represents the whole input point set.

Hand Feature Decoder (HFD). We design a HFD f_{HFD} to recover the input point cloud from the encoded global feature vector. As shown in Figure 4, we generate point cloud from a network with two parallel branches [8, 18], i.e. a fully connected branch and a deconvolution branch. It has been proved in [8] that the two-branch approach has better performance in producing point cloud than the single-branch method. The fully connected branch predicts $\hat{N}_1$ points. This branch helps the decoder enjoys high flexibility since each point is predicted independently. The deconvolution branch predicts a feature matrix with the size of $3 \times W \times H$, where $\hat{N}_1 = W \times H$ is the number of points. Thanks to spatial continuity induced by the convolution layers, the recovered points are more geometric consistent. Additionally, weight sharing of this branch helps it requires less parameters compared to the fully connected branch. Above introduces the design of HFD, and we propose to use the Chamfer distance (CD) as our decoder loss ($Loss_D$) to evaluate the similarity between the recovered

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 15:31:10 UTC from IEEE Xplore. Restrictions apply.

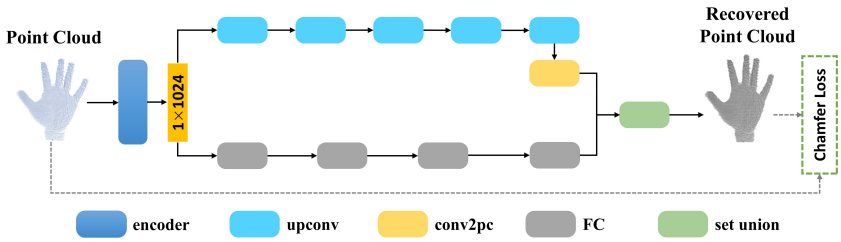

Figure 4. The architecture of the hand feature decoder (HFD) which takes the input point cloud and recovers a new point cloud. The FC branch predicts each point independently and shows good performance at describing intricate structures. The upconv branch consists of deconvolution and convolution, and used to exploit spatial continuity. The conv2pc module consists of two 1×1 convolution layers. The predictions of two branches are later merged together to form the whole set of points.

point cloud $X_r \in \mathbb{R}^3$ and the input point cloud $X \in \mathbb{R}^3$:

$$Loss_D(X_r, X) = \frac{1}{|X_r|} \sum_{x \in X_r} \min_{y \in X} \| x - y \|_2^2 + \frac{1}{|X|} \sum_{y \in X} \min_{x \in X_r} \| x - y \|_2^2. \quad (2)$$

Note that the number of points in X and X_r are not necessarily the same. For each point, CD finds the nearest neighbor in the other point set and sums the distances up.

Hand Pose Estimator (HPE). To recover hand pose from the features extracted in HFE, we construct a hand pose estimator f_{HPE}. Since multi-level features are obtained in the pipeline of encoder, they can be applied as the input of HPE. But whether those features have an impact on pose estimation needs to be verified. According to the characteristics of different level features as well as the combination methods, four variants of input features of HPE are constructed. The input can be the global feature or the other three variants (as shown in Figure 5), we compare the performance of these different fusion methods in Section 4.1. The integrated features are forward into a shared fully connected layer to ensure each channel has the same size, and then using average pooling to fuse the redundant information. Besides, a series of fully connected layers are applied to regress the coordinate of hand joints. When training the network, we use the Euclidean distance (ED) as the loss function on predicted poses as defined in the equation below:

$$Loss_E(\hat{P}, P) = \frac{1}{|N_J|} \sum_{i=1}^{N_J} (\| \hat{pose}_i - pose_i \|_2^2), \quad (3)$$

where $\hat{pose}_i$ is the predicted coordinate of the i-th joint and $pose_i$ is the corresponding ground truth coordinate.

3.4. Semi-supervised training

Annotating the ground-truth for 3D pose estimation is both challenging and time-consuming when constructing a dataset. We introduce a semi-supervised training method to use less annotated data to train an applicable hand pose estimation model by making full use of unannotated data (as shown in Figure 1). When using the unannotated data to train the network, the HFD recovers a new point cloud which is compared with the original point cloud and then a point cloud Chamfer loss $Loss_D$ is computed. In this case, the training loss $Loss_{t1}$ is defined as:

$$Loss_{t1} = Loss_D. \quad (4)$$

This training Loss $Loss_{t1}$ is applied to optimize the HFE and HFD. When using the annotated data to train the network, apart from the Chamfer loss $Loss_D$ is computed by HFD, the HPE predicts the 3D hand pose and computes the pose loss $Loss_E$. For the annotated data, the training loss $Loss_{t2}$ is defined as:

$$Loss_{t2} = \lambda \times Loss_E + Loss_D, \quad (5)$$

where λ is the weighting factor. The training loss $Loss_{t2}$ is used to optimize the whole network.

4. Experiments

In this section, to evaluate the proposed method, four challenging publicly available hand pose datasets are selected for experimenting. **ICVL Dataset** [35] contains 22,059 frames for training and 1,596 frames for testing. The dataset provides the ground truth of 16 hand joints of each frame. **MSRA Dataset** [33] contains more than 76K frames from 9 subjects. Each subject has 17 gestures captured and each gesture has about 500 frames. For each frame, the dataset provides the bounding box of the hand region as well as the coordinates of 21 hand joints. Following pervious works, we utilize the leave-one-subject-out cross-validation strategy for evaluation. **NYU Dataset** [36] contains 72,757 training-set frames and 8,252 testing-set frames. For each frame, the RGBD data from 3 Kinects is provided. In our experiment, we only use the depth image. The ground truth contains $J = 36$ annotated joints, we conduct evaluation on

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 15:31:10 UTC from IEEE Xplore. Restrictions apply.

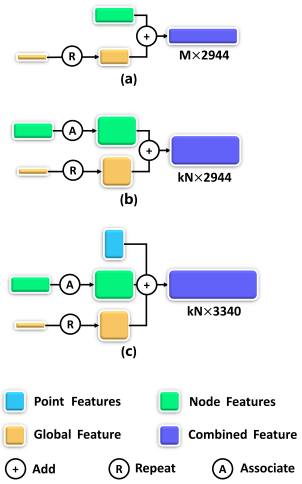

Figure 5. Strategies of feature fusion. (a) Fusing global and node features in M channels. (b) Fusing global and node features in kN channels. (c) Fusing global, node and point features in kN channels.

the subset of $J = 14$ hand joints as [7, 10, 12, 39] and also we only use *view 1* for both training and testing. **HANDS 2017 Challenge Frame-based Dataset** [44] contains 957k training and 295k testing frames, which are sampled from BigHand2.2M [45] and FHAD [9] dataset. This dataset has 21 annotated joints.

We evaluate the hand pose estimation performance with two commonly used metrics. The first metric is the per-joints mean error in Euclidean space over all test frames as well as the overall mean error for all joints over all test frames. The second metric is the fraction of good frames in which the maximum joint error is bellow a threshold.

For network architecture, we input the sampled and normalized points as well as the surface normal vectors. The number of sampled points N is set as 1024, and the k of kNN search is 3. We choose a SOM of size 8×8. Our experiments are conducted within PyTorch framework on a workstation with Intel Xeon E5-2620, 64GB of RAM and a NVIDIA TITAN Xp GPU.

4.1. Self-comparisons

Impact of fusion strategies. To better represent the input point cloud, the HFE hierarchically extracts multi-level features: point features, node features and global feature. First, we would like to find out whether the node feature and the point feature actually help the pose regression. We use the global feature as the input of HPE as the baseline method, and construct three fusion strategies to combine features (as shown in Figure 5). **(a) Global + Node features (v1)**. The global feature vector is repeated M times and then concatenated with the M node features. **(b) Global + Node features (v2)**. The global feature vector is repeated kN times and then concatenated with the kN associated node features. **(c) Global + Node + Point features**. Transfer global feature and node features into kN features as above, and then combine global, node and point features to an integration feature. We evaluate these different fusion strategies on ICVL dataset. Notably the size of different combined features have no influence on our estimation process, since a shared FC and average fooling are conducted to convert the combined features into a fixed size. As presented in Figure 6, **Global + Node features (v1)** gets the highest accuracy. Compared with only using global feature, fusing with node feature improves the performance. We also compare two fusion strategies to incorporate the global and node features, and find that **v1** outperforms **v2**. Besides, adding point features to the prior fusion makes no contribution and slightly damages the performance.

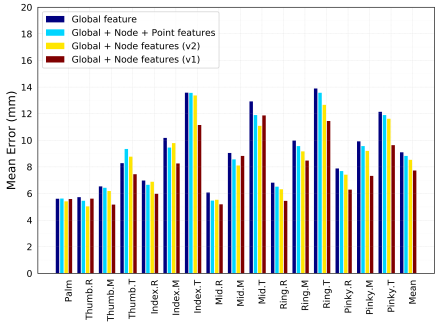

Figure 6. Self-comparison on ICVL dataset. The impact of fusion strategies on per-joint mean error and overall mean error are shown.

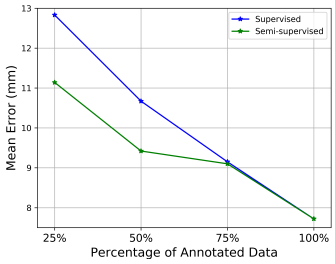

Figure 7. Self-comparison on ICVL dataset. The mean error of our model trained supervised and semi-supervised with the percentage of annotated data.

Impact of semi-supervised learning. We study the impact of semi-supervised learning on ICVL dataset and HANDS 2017 Challenge Frame-based dataset. With the same network architecture, we use part of the annotated data from the training set to train the whole network and meanwhile use the rest of data to train the autoencoder without using their pose information. As shown in Figure 7, the

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 15:31:10 UTC from IEEE Xplore. Restrictions apply.

performance is significantly promoted compared with the method that only trained by the same size of annotated data on ICVL dataset. When using 25%, 50% and 75% of annotated data to conduct model training, semi-supervised training witnessed an improvement at 13.2%, 11.7% and 0.7% accordingly. When the ratio of the annotated data is small e.g. 25%, using unlabeled data can get significant improvement on hand pose estimation. The performance of training with 25% annotated data via semi-supervised scheme is comparable to training with 50% annotated data in normal scheme. This characteristic can also be observed when comparing the performance of the method that using 50% annotated data for training in the semi-supervised manner with the method that using 75% annotated data for training in general way. On HANDS 2017 Challenge Frame-based Dataset, we use 10% of annotated data as well as the rest unlabeled data to conduct model training, and the improvement is also significant (as shown in Table 1). The mean error of fully-supervised training is 40.03 mm after 15 epochs, while the mean error of our semi-supervised training is 24.65 mm. Note that no data augmentation is implemented in our experiments.

Method	Fully-supervised	Our Semi-supervised
Mean error	40.03 mm	24.65 mm

Table 1. The mean error of our model trained supervised and semi-supervised with the 10% of labeled data on HANDS 2017 Challenge Frame-based Dataset.

4.2. Comparisons with State-of-the-arts

We compare our method with some of the state-of-the-art methods including LRF [35], Deep Model [47], DeepProir [23], Crossing Nets [38], Cascade [33], HBE [48], V2V-PoseNet [21]. As shown in Table 2, we achieved comparable accuracy when utilize all annotated training data under the condition that without any data augmentation. We get better result than all the other methods except to V2V-PoseNet. Remarkably, we use less data to train the network and the computational cost of us is also much lower (as shown in Table 3). In general, we obtain comparable performance with state-of-the-art supervised methods in real-time hand pose estimation.

To verify the effectiveness of the semi-supervised training strategy, we compare our method with the state-of-the-arts [3, 31, 38] which also aim at solving the challenging of data annotation. Different from the proposed method that reduces the amount of annotated data for training and making full use of unannotated data, the key idea of Baek *et al*. [3] is to synthesize data in the skeleton space for data augmentation. The percentage of annotated frames of Beak *et al*. is 100% and they train the model by the augmented set which is around 10 times larger than the original

Method	Mean Error(mm)
LRF	12.6
Deep Prior	11.6
Deep Model	10.4
Crossing Nets	10.2
Cascade	9.9
HBE	8.6
Ours	**7.7**
V2V-PoseNet	6.3

Table 2. Comparison of the mean error with the state-of-the-arts on ICVL dataset.

Method	Parameter Quantity	Testing Speed
V2V-PoseNet	457.5M	3.5fps
Ours	16.6M	58fps

Table 3. Comparison of parameter quantity and testing time on single GPU.

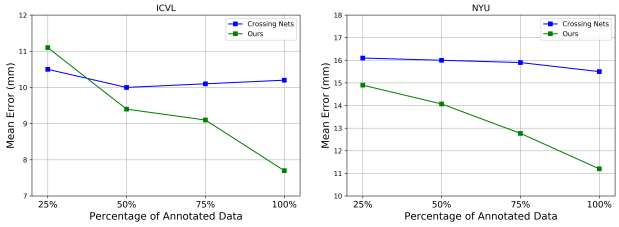

Figure 8. Comparison with Crossing Nets on ICVL (left) and NYU (right) datasets. The mean error of semi-supervised trained model with the percentage of annotated data.

training set. As can be seen in Table 4, comparing with [3], our method obtains better performance when using the same amount of annotated data, moreover, our method is also superior to them when using only part of annotated frames. The Crossing Nets [38] is one of the milestone works that perform accurate hand pose estimation in the semi-supervised setting. We compare our method to [38] with the same percentage of annotated training data. As shown in Figure 8 and Table 4, our method outperforms them in most of the experiments. Whats more, as the number of annotated frames increased from 25% to 75%, their method shows little improvement, whereas our method gets significant promotion. Figure 9 shows that our method has better performance than most recently semi-supervised methods [31, 38] over most of the error thresholds on three datasets. On NYU dataset, when the maximum allowed distance is between 20mm and 30mm, the fraction of good frames of our method is about 30% better than them. On MSRA dataset, when the maximum allowed distance is between 20mm and 30mm, the fraction of good frames of our method is about 15% better than them. On ICVL dataset,

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 15:31:10 UTC from IEEE Xplore. Restrictions apply.

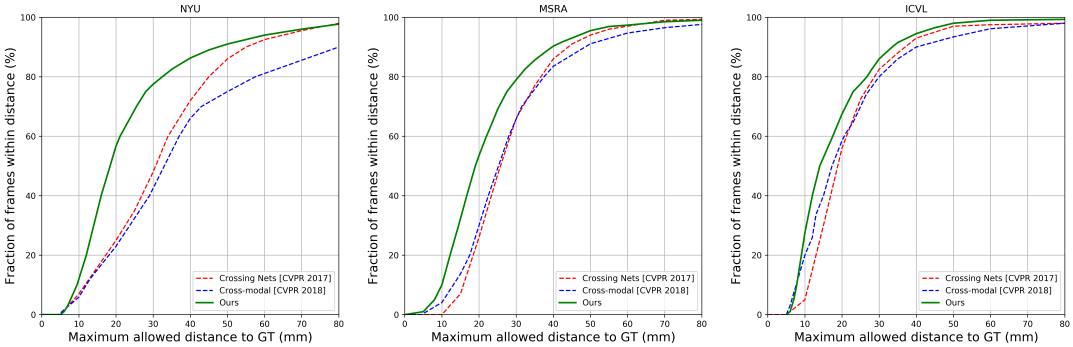

Figure 9. Comparison of our approach with recently state-of-the-art semi-supervised methods on NYU (left), MSRA (middle) and ICVL (right) datasets. The proportion of good frames over different error thresholds are presented in this figure.

Method	Annotated Frame Usage	Augmented Set	ICVL(mm)	NYU(mm)
Beak *et al*.(baseline)	100%	No	12.1	17.3
Beak *et al*.(w/o aug.; refine)	100%	No	10.4	16.4
Beak *et al*.(w/o refine)	100%	Yes, 10 times	9.1	14.9
Beak *et al*.	100%	Yes, 10 times	8.5	14.1
Crossing Nets	25%	No	**10.5**	16.1
	50%	No	10.0	16.0
	75%	No	10.1	15.9
	100%	No	10.2	15.5
Ours	25%	No	11.1	**14.9**
	50%	No	**9.4**	**14.1**
	75%	No	**9.1**	**12.8**
	100%	No	**7.7**	**11.2**

Table 4. Comparison of our work with semi-supervised methods on ICVL and NYU datasets. We evaluate the performance by test estimation error as well as the percentage of annotated data and total data used for model training. Our network produces better accuracy and more applicable to circumstance when the annotated data is limited.

when the maximum allowed distance is 15%, the fraction of good frames of our method is about 10% better than [31] and about 20% better than [38]. In summary, our method provides a practical mode to reduce the reliance on annotated data in hand pose estimation task and outperforms recent semi-supervised methods.

4.3. Runtime and Model Size

The testing time of us is 17.2ms in average, specially, 8.2ms for data processing including point sample and surface normal computation, 9.0ms for the hand pose estimation. Our approach runs in real-time at about 58fps. In addition, the size of HFE, HFD and HPE are 8.1M, 74M and 8.5M respectively. Since we only employ the HFE and HPE in the testing stage, the model size of our network is 16.6MB.

5. Conclusions

In this paper, we present a novel network for 3D hand pose estimation from a single depth image. To better represent the original data and perform more efficiently feature extraction, we convert the depth image into point cloud and extract multi-level features by a self-organizing encoder. Multi-level features are fused to regress accurate 3D hand pose. Moreover, we utilize a decoder to optimize the encoding process in the training phase. Additionally, to alleviate the burden of laborious 3D hand pose annotation on training data, we propose to train our hand pose estimation network in a semi-supervised manner with both annotated data and unannotated data. Experimental results on four datasets show that our proposed SO-HandNet achieves superior performance in semi-supervised training for 3D hand pose estimation from depth images.

Acknowledgement: This work is in part supported by the National Key Research and Development Program of China (grant Nos. 2018YFB1600600, 2016YFB0502200 and 2016YFB0502201). This work is also supported in part by startup funds from University at Buffalo.

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 15:31:10 UTC from IEEE Xplore. Restrictions apply.

References

[1] Mikaela Angelina Uy and Gim Hee Lee. Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4470–4479, 2018.

[2] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5297–5307, 2016.

[3] 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.

[4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[5] Xinghao Chen, Guijin Wang, Hengkai Guo, and Cairong Zhang. Pose guided structured region ensemble network for cascaded hand pose estimation. *Neurocomputing*, 2019.

[6] 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.

[7] Xiaoming Deng, Shuo Yang, Yinda Zhang, Ping Tan, Liang Chang, and Hongan Wang. Hand3d: Hand pose estimation using 3d neural network. *arXiv preprint arXiv:1704.02224*, 2017.

[8] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017.

[9] Guillermo Garcia-Hernando, Shanxin Yuan, Seungryul Baek, and Tae-Kyun Kim. First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 409–419, 2018.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] Evangelos Kalogerakis, Melinos Averkiou, Subhransu Maji, and Siddhartha Chaudhuri. 3d shape segmentation with projective convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3779–3788, 2017.

[15] Leonid Keselman, John Iselin Woodfill, Anders Grunnet-Jepsen, and Achintya Bhowmik. Intel realsense stereoscopic depth cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–10, 2017.

[16] Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 863–872, 2017.

[17] Teuvo Kohonen and Timo Honkela. Kohonen network. *Scholarpedia*, 2(1):1568, 2007.

[18] Jiaxin Li, Ben M Chen, and Gim Hee Lee. So-net: Self-organizing network for point cloud analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9397–9406, 2018.

[19] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems*, pages 828–838, 2018.

[20] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015.

[21] 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.

[22] Markus Oberweger and Vincent Lepetit. Deepprior++: Improving fast and accurate 3d hand pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 585–594, 2017.

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

[24] 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.

[25] Georg Poier, Michael Opitz, David Schinagl, and Horst Bischof. Murauer: Mapping unlabeled real data for label austerity. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1393–1402. IEEE, 2019.

[26] 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.

[27] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 15:31:10 UTC from IEEE Xplore. Restrictions apply.

multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2016.

[28] 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.

[29] 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.

[30] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017.

[31] 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.

[32] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] Zhigang Tu, Hongyan Li, Dejun Zhang, Justin Dauwels, Baoxin Li, and Junsong Yuan. Action-stage emphasized spatiotemporal vlad for video action recognition. *IEEE Transactions on Image Processing*, 28(6):2799–2812, 2019.

[38] Chengde Wan, Thomas Probst, Luc Van Gool, and Angela Yao. Crossing nets: Combining gans and vaes with a shared latent space for hand pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 680–689, 2017.

[39] 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.

[40] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[41] 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.

[42] Tan Yu, Jingjing Meng, and Junsong Yuan. Multi-view harmonized bilinear network for 3d object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 186–194, 2018.

[43] 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.

[44] Shanxin Yuan, Qi Ye, Guillermo Garcia-Hernando, and Tae-Kyun Kim. The 2017 hands in the million challenge on 3d hand pose estimation. *arXiv preprint arXiv:1707.02237*, 2017.

[45] Shanxin Yuan, Qi Ye, Bjorn Stenger, Siddhant Jain, and Tae-Kyun Kim. Bighand2. 2m benchmark: Hand pose dataset and state of the art analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4866–4874, 2017.

[46] Zhengyou Zhang. Microsoft kinect sensor and its effect. *IEEE multimedia*, 19(2):4–10, 2012.

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

[48] 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.

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 15:31:10 UTC from IEEE Xplore. Restrictions apply.