Deep Fusion and Ensemble Neural Networks for Point Sets Data

Yi Kang*, Zhang Na, Zhao Yuting and Lei Yiyan

1 College of Information and Communication, National University of Defense Technology, Xi’an, Shannxi, 710106, China

*Corresponding author’s e-mail: penfangs@163.com

Abstract. The typical character of point cloud is that its format is irregular, so most researchers transform such data into regular format. However, this conversion may induce some noise. In this paper, we design the dedicated neural network unit for point sets data processing. Firstly, to guarantee the permutation invariance of point sets, we infer the neural unit design principles. Secondly, we propose the deep fusion and ensemble neural networks for point sets data, in which we can not only combine different scales of feature representation, but also input the point data into the network directly. Finally, we evaluate the proposed networks on object classification and object part segmentation, which can verify the efficiency of our network.

1. Introduction

The typical character of 3D data, especially point cloud, is irregular. Thus, this specification impedes its direct usage in deep CNN. To solve this problem, most researchers firstly transform such data into regular format before applying a deep CNN model, such as voxel grid, 3D mesh, multi-view, tree-based methods. However, these conversion and geometric methods may induce some noise to fresh data which is harmful to the successive process. PointNet [1] and PointNet++ [2] are the pioneer works designed specifically to handle the irregularity of point clouds, which can manipulate raw point cloud data directly. In this paper, we present a theory of how to design the neural unit to process point data like PointNet.

Typical well-designed deep classification CNN like ResNet [3], GoogleNet [4], and deep-fused Nets [5] indicated the significance of combining different features for performance improvement. Another observation is that we can use simple ensemble techniques to enhance the final result, in which, the widely used approach is averaging the test results of numerous neural networks[6]. Being inspired by these ideas, we propose some architecture design methods to promote point sets data classification and part-segmentation. Finally, we transform our deep fusion network architectures to ensemble networks and experiments show that it is helpful to advance the final performance.

2. Permutation Invariance Neural Unit Design Theory

Here, we propose a neural unit design theory to guarantee the permutation invariance of the point data.

2.1. Single feature channel

Suppose point sets data are $X^N = \{x_1, x_2, x_3, \cdots, x_N\}$ which contains $N$ elements and each $x_i (i \in N)$ possesses single channel feature. The $f': X^N \rightarrow Y$ is a function which maps the point sets to a dedicated attribute value, and it can have a variety of expressions [7]. Suppose there are $\Pi_N$ permutation forms for $X^N$, the permutation invariance of $X^N$ in this model can be represented as
\[
f(\pi X^N) = f\pi(X^N) = Y \quad \forall \pi \in \Pi_N
\]  
(1)

where \(\pi\) represents one of permutation forms, and the action of \(\pi\) on point sets \(X^N\) should not influence their attribute value \(Y\).

Similarly, \(g : X^N \rightarrow Y^N\) denotes another function which maps \(X^N\) to a multi-dimensional attribute values. In this condition, the permutation invariance of point sets can be represented as

\[
g(\pi X^N) = g\pi(X^N) = \pi Y^N \quad \forall \pi \in \Pi_N
\]  
(2)

When applying point sets to a neural network, we should make sure weighted operation is equal to \(\pi\) operation to ensure permutation invariance of point sets. When the input is single channel feature, the operation of network can be written as

\[
\sigma(WX^N) = \sigma(\pi X^N) = \sigma\pi(X^N) \quad W \in R^{N \times N}
\]  
(3)

where \(X^N\) denotes the input, \(W\) is the weight matrix and \(\sigma\) is a nonlinear activation function.

**Theorem 1** The neural unit can keep input data permutation invariance, if the weight operation is designed as Equation 4

\[
WX^N = (\lambda - \theta)X^N + \theta(11^T)X^N \quad \lambda, \theta \in R, 1 = [1, \cdots, 1]^T \in R^N
\]  
(4)

where \(1\) is the identity matrix.

When we apply \(W\) to the input \(X^N\), the output will be consisted of weighted input data \((\lambda - \theta)X^N\) and the weighted summation \(\theta(11^T)X^N\). The summation item \(\theta(11^T)X^N\) will not be influenced by the permutation and the item \((\lambda - \theta)X^N\) is equivalent to operation on each point, so designing neural unit following Equation 4 can guarantee the permutation invariance on single feature channel data.

### 2.2. Multiple features channels

When the dimension of data in point sets is \(K\), it can be denoted as \(X^{N \times K} = \{x_1^K, x_2^K, \cdots, x_N^K\}\) and the output is \(Y \in R^{N \times L}\). There also exists a function: \(X^{N \times K} \rightarrow Y^{N \times L}\) which maps \(K\) features to \(L\) for each element. Now, the neural unit on multi-features channels data can keep permutation invariance, if the weight operation is designed as Equation 5.

\[
X^{N \times K}W^{K \times L} = X^{N \times K}(\Lambda - \Theta) + (11^T)X^{N \times K}\Theta \quad \Lambda, \Theta \in R^{K \times L}, 1 = [1, \cdots, 1]^T \in R^N
\]  
(5)

where the difference between Equation 4 and 5 is that we replace scalars \(\lambda\) and \(\theta\) with matrixes \(\Lambda\) and \(\Theta\). \(X^{N \times K}(\Lambda - \Theta)\) represents the matrix product of input data and weight, and \((11^T)X^{N \times K}\Theta\) is a weighted summation of each features column. They all can keep permutation invariance. Furthermore, the summation process can be replaced by maximization, minimization, mean and so on.

### 2.3. Neural unit design

Equation 5 is our focus in designing neural unit because the point cloud is usually denoted as multiple channel features. In practice, we use a variation of the Equation 5 which only uses a single parameter \(\Theta \in R^{K \times L}\). The variation will not disturb the permutation invariance, but the number of network parameters will be reduced. The simplified version can be written as follows

\[
X^{N \times K}W^{K \times L} = \left(X^{N \times K} - (11^T)X^{N \times K}\right)\Theta \quad \Theta \in R^{K \times L}, 1 = [1, \cdots, 1]^T \in R^N
\]  
(6)
where the input is the difference between each point features and their norm values, we call it sum-normalized operation. Accordingly, we can think of these operations as adding global features at each point. The process after norm operation is uniform with MLP in PointNet [11].

Based on above analysis, our designed neural unit can be denoted as Figure 1, where the primary input is a collection of point sets with size of $n \times c$, in which $n$ indicates independent points and $c$ denotes the feature dimension of each data, such as 3D coordinates, color values, normal directions and so on.

3. Network Construction
The typical forward process in PointNet [1] is shown in Figure 2, which includes a classification structure, e.g., a max-pool layer. Here, we define a description that the features before max-pool layer are called local features, and features after it are global features. For classification task, global features are the major evidence to judge category. However, for part segmentation task, concatenating the local features and global features from the max-pool layer is the key step. In this section, we propose two optimization strategies about features combination to enhance the performance of the CNN structure.

3.1. Base Network Architecture
In Figure 2, a point sets data are used as input to the network. To output 3D object categories, the features from last permutation invariance unit are pooled and passed through three fully connected layers. Between the three fully connected layers, dropout is applied to regularize the learning [9].

Object part segmentation network is an expanded network based on the classification network. It keeps all permutation invariance neural units and max-pool layer, and removes the last three fully connected layers. Then it concatenates the global features from max-pool layer with local features of some intermediate neural units. Finally, the four new permutation invariance neural units are added.

3.2. Deep Fusion Network Architecture
When we are judging the category labels, last CNN layer is the optimal feature representation [8]. However, in 3D object classification or segmentation tasks, the top layer is not enough. Since the influence coming from the change of location, angle and other uncertain changes, all of them should be considered. The features in early layers are helpful for locating edges or corners of object [10].

In practice, we present a strategy to integrate early features and last layer features into more complicated features, and we call it deep fusion. The idea of deep fusion is simple. It stresses a connection from shallow features produced by the intermediate neural unit to top layer features. For convenience, we named shallow features as SF and top layer features as TF.

According to the design of the base network, we take the connection as a combination problem that chooses one or more SFs from the four SFs to connect with TF, and there are 20 combinations. The detailed construction descriptions are given in Table 1. For clear understanding, we take Form6 as an
instance. Two forms of Form6 are shown in Figure 3(a), 3(b), in which the blue boxes correspond to the SF, the pink blocks represent pool operation and the green blocks represent combination operation. Different forms are distinguished by the features flowing [11]: (i) We call Figure 3(a) multidirectional fusion, as its global features are formed by concatenating pooled SF and pooled TF; and (ii) Figure 3(b) is called unidirectional fusion that concatenates original SF and TF followed by a max-pool layer. Multidirectional and unidirectional fusion have the same number of parameters, but shallow pooled feature is our fused goal, so multidirectional fusion will be a structure bed in the latter part. For part segmentation, the structure of capturing global features in fusion form is the same as the classification network, so the deep fusion process is still helpful.

Figure 3. Network architectures with fusion and ensemble. (a) Deep multidirectional fusion. (b) Deep unidirectional fusion. (c) An example of ensemble structure.

| Fusion forms | SF1 | SF2 | SF3 | SF4 | TF |
|-------------|-----|-----|-----|-----|----|
| Form1       | ✓   | ✓   | ✓   | ✓   | ✓  |
| Form2       | ✓   | ✓   | ✓   | ✓   | ✓  |
| Form3       | ✓   | ✓   | ✓   | ✓   | ✓  |
| Form4       | ✓   | ✓   | ✓   | ✓   | ✓  |
| Form5       | ✓   | ✓   | ✓   | ✓   | ✓  |
| Form6       | ✓   | ✓   | ✓   | ✓   | ✓  |

3.3. Ensemble Network Architecture

Ensemble technique is a popular method to design network and it can improve performance obviously [12]. Figure 3(a) shows a deep fusion form, where the global feature is composed of four SFs and one TF. There will be five information streams converging in concatenation block: four SFs representing intermediate units and one TF of the last layer. We can change the location in structure where the information converges, Figure 3(c) is one of all information stream variations based on Figure 3(a). The four information flow networks are depicted in Figure 3(c), where each net uses only a subset of global feature composition in Figure 3(a). We can see that the four networks, if combined together, can form an ensemble. The architecture resemblance between Figure 3(a) and Figure 3(c) suggests the close relation between the fused net and the ensemble form. Generally, a fused net can be transformed to some base ensemble networks. During practice, we first obtain some base ensemble structures, then take the mean value of the test results from all base structures as our final result.
4. Experiment

4.1. Object Classification
We evaluate point sets CNN model with deep fusion and ensemble strategies on ModelNet40 dataset, which consists of 40 classes of 3D objects. The classification network outputs $k$ class scores or categories information for an object where the biggest one represents the correct category.

**Base Network** The base classification network takes 1024 points from each object as input, applies five permutation neural units and two T-Nets, and then get global features by max-pool operation. Our evaluation metrics are the mean category accuracy and mean accuracy of all objects.

**Deep Fusion** In this part, we present the classification results from some deep fusion networks, where the base network is PointNet. The key step for deep fusion is the connection of SF and TF. Here, we consider six typical fused nets given in Table 1. In our network construction, we use max pooled SF and multidirectional fusion method.

The performance of deep fusion for classification task is competitive though weak in improving over PointNet. The experiment comparison with PointNet is given in Table 2. It can be seen that our fusion Form2 reaches about 0.1% mean accuracy and 0.3% per-category accuracy improvement. In contrast, there is a clear performance difference when the fusion is conducted with other SFs. The classification task takes the last layer categories information as key feature representation, and the shallow features near the input layer just include edges or corners information. When the global features produced by the last permutation neural unit are sufficient for classification, adding more SFs to enforce result is far-fetched. PointNet++ also mentioned that the weak point of PointNet is it does not capture local structures induced by the metric space [2]. Summarily, we verify two facts: the category features in top layer is reliability for PointNet and deep fusion has little effect for classification on raw network.

**Table 2. Classification performance of our deep fusion models compared to base structure.**

| Architecture       | accuracy avg. class(%) | accuracy overall(%) |
|--------------------|------------------------|---------------------|
| PointNet (Base network) | 86.2                  | 89.2                |
| Form1              | 85.6                   | 89.0                |
| Form2              | **86.5**               | **89.3**            |
| Form3              | 85.7                   | 88.9                |
| Form4              | 86.3                   | 89.1                |
| Form5              | 85.1                   | 88.7                |
| Form6              | 86.2                   | 89.2                |

4.2. Object Part Segmentation
We validate the performance of deep fusion and ensemble strategies in part segmentation on ShapeNet part dataset[13]. The goal of part segmentation is to assign part category information to each point of the object, and the network outputs $n \times m$ scores for $m$ part scores in each of the $n$ points.

**Base Network** The part of extracting local and global features is the same as the classification network and another part is connection features from some point sets neural permutation invariance units. Our evaluation metric is mean IoU (intersection over union): For every object IoU is computed as an average IoU of each part which is occurred in this object’s part categories.

**Deep Fusion** Here, we designed our network structures according to the forms given in Table 1. Table 4 reports our results. Compared with the classification fused structures, many of the part fused segmentation structures can achieve a performance boost. Our results show original part segmentation features are rough and adding intermediate features is helpful for parts recognition.
Table 3. Part segmentation performance of our deep fusion models compared to base structure.

| Architecture   | mean | airplane | bag  | cap  | car   | chair  | ear   | phone | guitar | knife | lamp | laptop | motor | mug | pistol | rocket | skate | board | table |
|----------------|------|----------|------|------|-------|--------|-------|-------|--------|-------|------|--------|-------|-----|--------|--------|-------|-------|-------|
| PointNet       | 83.7 | 83.4     | 78.7 | 82.5 | 74.9  | 89.6   | 73.0  | 91.5  | 85.9   | 80.8  | 95.3 | 65.2   | 93.0  | 81.2 | 57.9   | 72.8   | 80.6  |
| Form1          | 83.9 | 83.5     | 82.6 | 82.3 | 77.0  | 90.2   | 72.4  | 91.5  | 86.8   | 80.4  | 95.3 | 67.4   | 92.3  | 81.5 | 60.4   | 73.7   | 80.1  |
| Form2          | 83.4 | 82.6     | 77.6 | 81.6 | 75.5  | 89.0   | 71.7  | 90.9  | 83.8   | 78.9  | 95.3 | 66.5   | 91.7  | 81.2 | 58.2   | 74.2   | 81.0  |
| Form3          | 83.9 | 83.5     | 82.1 | 83.8 | 77.3  | 90.0   | 69.6  | 91.5  | 85.8   | 80.5  | 95.7 | 67.3   | 93.7  | 81.0 | 59.9   | 75.1   | 80.2  |
| Form4          | 83.8 | 83.7     | 79.1 | 84.3 | 77.1  | 89.8   | 70.2  | 91.4  | 85.4   | 80.5  | 95.2 | 67.5   | 91.8  | 81.5 | 60.9   | 72.9   | 80.5  |
| Form5          | 83.5 | 83.5     | 83.9 | 85.5 | 77.1  | 90.1   | 70.7  | 91.3  | 85.8   | 80.2  | 95.3 | 66.6   | 92.7  | 81.3 | 62.2   | 75.6   | 80.0  |
| Form6          | 83.5 | 82.6     | 77.9 | 79.6 | 74.4  | 89.6   | 72.2  | 91.2  | 85.6   | 80.0  | 95.4 | 65.4   | 91.1  | 81.4 | 58.1   | 72.6   | 80.9  |

**Ensemble** We adopt PointNet and fused networks as our sub-networks and then constitute them to some ensemble networks. Our results on ShapeNet are shown in Table 5. Our ensemble structures perform better than PointNet with a biggest increased mean IoU by 1.1%. The good performance indicates that utilizing the finer information from multiple networks is helpful for performance improvement and our deep fused networks are effective to summarize the detail features of the object.

Table 4. Performance comparison on the ShapeNet dataset with the ensemble strategy.

| Architecture | mean | airplane | bag  | cap  | car   | chair  | ear   | phone | guitar | knife | lamp | laptop | motor | mug | pistol | rocket | skate | board | table |
|--------------|------|----------|------|------|-------|--------|-------|-------|--------|-------|------|--------|-------|-----|--------|--------|-------|-------|-------|
| PointNet     | 83.7 | 83.4     | 78.7 | 82.5 | 74.9  | 89.6   | 73.0  | 91.5  | 85.9   | 80.8  | 95.3 | 65.2   | 93.0  | 81.2 | 57.9   | 72.8   | 80.6  |
| Base+Form1   | 84.5 | 84.3     | 81.8 | 83.0 | 76.6  | 90.2   | 74.2  | 91.6  | 85.9   | 81.1  | 95.3 | 66.8   | 92.1  | 83.0 | 61.1   | 74.4   | 81.5  |
| Base+Form2   | 83.6 | 83.2     | 78.7 | 85.0 | 74.0  | 89.2   | 74.2  | 90.9  | 83.6   | 79.5  | 95.2 | 65.5   | 91.7  | 82.4 | 58.0   | 74.2   | 81.4  |
| Base+Form3   | 84.4 | 84.3     | 80.7 | 85.3 | 77.3  | 90.3   | 72.4  | 91.7  | 85.8   | 81.4  | 95.5 | 67.4   | 91.9  | 83.1 | 60.6   | 74.2   | 81.0  |
| Base+Form4   | 84.3 | 84.1     | 79.9 | 84.9 | 76.4  | 90.0   | 74.1  | 91.4  | 85.0   | 81.3  | 95.3 | 68.1   | 91.9  | 82.0 | 60.2   | 72.9   | 81.5  |
| Form1+Form2  | 84.8 | 84.3     | 81.7 | 86.5 | 78.0  | 90.4   | 73.9  | 91.7  | 86.1   | 81.8  | 95.5 | 68.6   | 92.3  | 82.3 | 62.0   | 73.9   | 81.8  |
| +Form3       | 84.8 | 84.5     | 82.8 | 86.1 | 78.3  | 90.5   | 71.3  | 91.7  | 86.6   | 81.9  | 95.5 | 68.1   | 92.5  | 82.5 | 62.8   | 73.9   | 81.5  |

5. Conclusion

3D Point-based CNN is an increasing popular research areas. Due to its irregular nature, we provide a theoretical derivation for the design of permutation invariance neural unit. We also conduct a feature flowing analysis of base network and then provide some network variations with deep fusion and ensemble strategies. The experimental results prove that our methods can improve the classification and part segmentation performance although making little change on the basic network architecture.

References

[1] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. 2017. In Computer Vision and Pattern Recognition (CVPR).

[2] C. R. Qi, Li Yi, H. Su, and L. J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. arXiv:1706.02413, 2017.

[3] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep residual learning for image recognition. In Computer Vision and Pattern Recognition (CVPR).
[4] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. 2015. Going deeper with convolutions. In Computer Vision and Pattern Recognition (CVPR). 1–9.

[5] J. Wang, Z. Wei, T. Zhang, and W. Zeng. Deeply-Fused Nets. 2016. CoRR, abs/1605.07716.

[6] Harris Drucker, Corinna Cortes, Lawrence D. Jackel, Yann LeCun, and Vladimir Vapnik. Boosting and other ensemble methods. 1994. Neural Computation, 6(6):1289–1301.

[7] F.N. David, M.S. Kendall, and D.E. Barton. Symmetric Functions and Allied Tables. University Press, 1966.

[8] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Object Instance Segmentation and Fine-grained Localization using Hypercolumns. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016.

[9] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. JMLR. 2014.

[10] Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. CoRR, abs/1311.2901, 2013. Published in Proc. ECCV, 2014.

[11] M. Abdi and S. Nahavandi. Multi-residual networks. CoRR, abs/1609.05672, 2016.

[12] A. Veit, M. J. Wilber, and S. J. Belongie. Residual networks behave like ensembles of relatively shallow networks. CoRR, abs/1605.06431, 2016.

[13] Li Yi, Vladimir G. Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3D shape collections. ACM Trans. Graph. 2016.