Journal of Computers Vol. 34 No. 4, August 2023, pp. 163-173
doi: 10.53106/199115992023083404014

A Point Cloud Classification Method and Its Applications Based on Multi-Head Self-Attention

Xue-Jun Liu*, Wen-Hui Wang, Yong Yan, Zhong-Ji Cui, Yun Sha, Yi-Nan Jiang

Department of Information Engineering, Beijing Institute of Petrochemical Technology,
Beijing 102617, China

lxj@bipt.edu.cn, 1431786238@qq.com, yanyong@bipt.edu.cn, 295199696@qq.com,
{shayun, 0020220017}@bipt.edu.cn

Received 1 July 2023; Revised 20 July 2023; Accepted 25 July 2023

Abstract. In the monitoring the safety status of hazardous chemical warehouses by three-dimensional re-construction of deep camera point clouds, there are classification difficulties such as large space, sparse distribution of point clouds in cargo images, and similar distribution in low dimensions. Based on the above problem, a point cloud recognition method based on multi-head attention mechanism is proposed. The algorithm first normalizes the distribution of the point cloud data set through the affine transformation algorithm to solve the problem of sparse distribution. Then, the high-dimensional feature map is obtained by fusing the data down-sampling and curve feature aggregation algorithms to solve the problem of low-dimensional distribution approximation. The feature map is then encoded using a multi-head self-attention encoder to obtain features under different heads, which are then merged into a feature map. Finally, a multi-layer fully connected neural network is used as the decoder to decode the feature map into the final object classification. Comparative experiments were performed on the ModelNet40 dataset and the self-built dataset of warehouse goods, and the results showed that the accuracy of this paper was improved by 0.5% to 7.8% compared with that of other classification algorithms.

Keywords: point cloud classification, feature aggregation, multi-head self-attention, neural network

1 Introduction

With the rapid development of China’s social economy, various hazardous chemicals are also being used more and more widely. However, major safety accidents such as the Zhangjiakou 11.23 explosion and the Tianjin Port 8.12 explosion have shown that hazardous chemical storage safety management system still has serious shortcomings [1]. Traditional manual methods including inspection and duty are inefficient although they consume more time. The use of remote cameras to monitor increases efficiency, but is affected by problems such as obstruction and lighting [2]. Therefore, a more effective method of supervising hazardous chemical warehouses is needed.

With the development of 3D imaging technology [3], techniques such as structured light measurement and laser scanning have become mature, and the 3D coordinates of object surfaces can be obtained accurately and quickly, generating 3D data of the scene for better perception and understanding of the surrounding environment. 3D data contains depth information of the scene and can represent the shape of objects’ surfaces, with broad application prospects in fields such as robotics [4], AR/VR [5], human-computer interaction [6], remote sensing mapping [7], etc. 3D data can provide depth information that can help solve many problems in hazardous chemical warehouse storage, and thus using 3D data for classification and monitoring of goods in hazardous chemical warehouses has become a mainstream research direction in recent years [8].

However, unlike the regular arrangement of pixels in 2D images, point cloud data is unordered, making it difficult to directly apply convolution to obtain local correlation in-formation between 3D points. Additionally, due to the nature of data collection, point cloud data is often non-uniformly distributed, with varying point cloud densities in different local areas, posing challenges for data point sampling during feature extraction [9]. Furthermore, the deformation of objects in 3D space is more complex than that in 2D im-ages, as non-rigid deformation needs to be considered in addition to the affine transformation of the three dimensions [10]. Therefore, the main problems with point cloud classification are its sparsity and disorderliness.

* Corresponding Author

This work is organized as follows: Section 2 introduces some current methods of point cloud classification. In section 3, a point cloud classification model based on multi head attention mechanism is proposed in detail. Comprehensive experimental results are discussed in Section 4 for performance evaluations compared with other methods. Finally, concluding remarks and future directions are given in Section 5.

2 Related Works

Currently, there are three main methods for point cloud classification: multi-view-based methods, voxel-based methods, and point-based methods.

Multi-view-based methods project 3D point clouds onto 2D planes and classify the point clouds by processing the resulting images from different angles. The key challenge is how to fuse the features of multiple views into a distinctive global feature. Abdullah Hamdi [11] used an end-to-end multi-view transformation network (MVTN) to assign weights to images from different angles and find the best view for a specific task, but it requires a large number of views and has slow computation speed. Su [12] used MVCNN to encode depth maps from different views using an encoder and used contrastive learning to align depth features with visual features, but it is greatly affected by the domain gap between image depth.

Voxel-based methods voxelize point clouds and then classify them. Maturana [13] voxelized point clouds by VoxNET at multiple scales and used voxel convolution to extract features, followed by classification. However, this method is slow for high-resolution point clouds and requires large memory usage. Lifa Zhu [14] used Point-voxel adaptive feature abstraction (PVAFA) to voxelize point clouds and then used pyramid pooling to fuse features at multiple scales, followed by adaptive feature pooling. While voxel-based methods have achieved some success, they often lose important detail features, making it difficult to further improve performance.

Most existing methods are point-based methods, which process point clouds directly to retain their integrity. Charles [15] firstly proposed a new deep learning model PointNet for processing point cloud data and demonstrated its applicability to multiple cognitive tasks such as classification, semantic segmentation and object recognition. The network takes point data as input, uses feature transformations to extract features independently for each point, aggregates point features using max pooling, and finally uses a softmax layer to obtain the classification result of the point cloud. PointNet connects all points together and considers only global features, but lose the local information of each point. Zhang [16] then proposed LDGCNN, which connects hierarchical features of different dynamic graphs to extract features, and freezes the extractor during training the classifier. This method reduces the number of parameter and improves computation speed, but loses some local features.

3 Methods

This paper proposes a point cloud recognition model based on multi head attention mechanism, which classifies point clouds by extracting the relationships between points. Firstly, the point cloud is transformed by an affine transformation module to have a uniform spatial distribution. Then globally associated points are found for the feature points, and these points are aggregated into high-dimensional features. Finally, a decoder is used to decode high-dimensional features to obtain the final classification result. The algorithm flow is shown in Fig. 1:

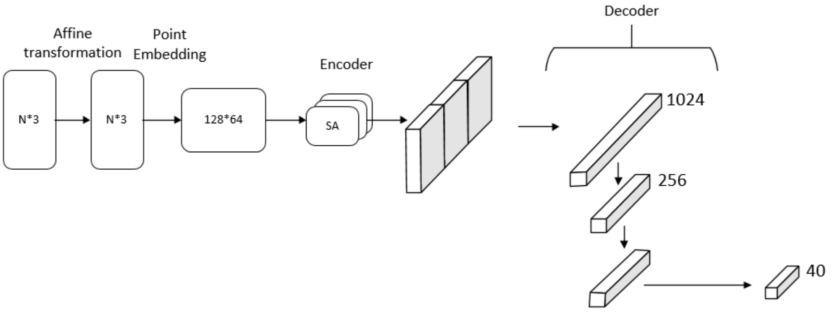


Fig. 1. Overall structure diagram of the algorithm

3.1 Affine Transformation

The distribution of point cloud is a sparse matrix, which has very discrete characteristic points in space. Therefore, simply using MLP to process the data will reduce the ac-curacy of the model because different point clouds have different levels of sparsity and geometric structures in space [17].

Therefore, an affine transformation is used to transform the points in space by translation and scaling, based on the characteristics of the dataset, so that they have a more consistent overall distribution. Assuming that $\{f_i\}_{i=1,2,\dots,N}$ is a point cloud with N points, each represented in the form of a three-dimensional vector, the coordinates of each point after affine transformation can be calculated by formula 1:

$$f_i = \alpha \cdot \frac{f_i - f}{\sigma + \varepsilon} + \beta. \quad \textbf{(1)}$$

where $f = \frac{1}{N}\sum_{i=1}^{N} f_i$ represents the centroid of the sample, $\sigma = \sqrt{\frac{1}{Nd} \cdot \sum_{i=1}^{N}\left(f - f_i\right)^2}$ represents the standard deviation of the dataset, *d* represents the dimensionality of the sample, α and *β* are two trainable parameters used to control the scale of the affine transformation during scaling and translation, ε is a control parameter, which is set to 1e-5 in this algorithm to ensure that there is no division by zero during the calculation process.

σ is related to the distribution of point clouds, so this method can adjust different distributed data sets into a unified form. It can be easily to find that the transformed dataset formally follows a normal distribution, while retaining the original geometric characteristics.

3.2 Point Embedding

The points in the point cloud have strong correlations with each other, so it is necessary to use the point embedding method to extract the associated features between points.

This paper proposes a point embedding method which includes down-sampling and feature aggregation to combine global features. Down-sampling can filter out some noisy points and make all samples have equal number of points, which can improve the classification performance of the model. Feature aggregation divides points that are associated to each other in space into one category, then merges them and map it to higher dimension, which increases the richness of features while reducing the number of useless features.

The point cloud down-sampling method generally uses the Farthest Point Sampling (FPS) method. The process of down-sampling N points to c points using FPS is as follows:

Step 1: Randomly select a point from the point cloud as the initial point of the sampled point set.

Step 2: At this time, put the selected points into set A, and put the points that have not been selected yet into set B. For each point p_b in B, record its minimum distance with j points in set A [18], which is denoted as t_b^i, as shown in formula 2:

$$t_b^i = \min\left(\left\{p_b^1, p_b^2, p_b^3, p_b^4, \dots, p_b^j\right\}\right). \quad \textbf{(2)}$$

Step 3: Use an array D with length N to store the distances from points in B to A, $D_{[i]} = t_b^i$, *i* = 1, 2, …, *N*.

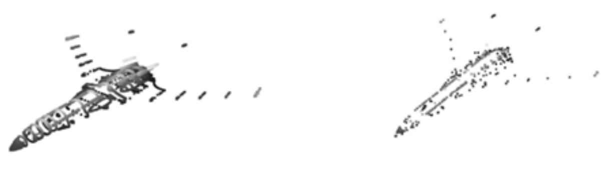
Step 4: Select the point in D with the maximum distance and move it from B to A.

Step 5: Update sets A and B, and then update the values in D according to formula 3:

$$t_b^i = \min(t_b^{i-1}, p_b^i). \quad \textbf{(3)}$$

Step 6: Repeat steps 4 and 5 until set A contains c points.

After multiple experiments, c is set to 128. The distributions of the point cloud before and after sampling are shown in Fig. 2:



(a) Point cloud before down-sampling (b) Point cloud after down-sampling

Fig. 2. Results and comparison of point cloud down-sampling

Points that can represent the features of point cloud are selected after down-sampling, and feature aggregation is used to make each point contains global features.

Assuming that the point cloud after down-sampling is set A and the original point cloud set is B. Feature aggregation finds N associated points in B for each point p_{Ai} in A, and combines these points into a single feature. The overall calculation process is shown in formula 4:

$$f_i = MP(LBR\{p_{A,i}, p_{B,j} \,|\, j = 1, 2, ..., N\}). \quad \textbf{(4)}$$

where $p_{A,i}$ represents the ith point in A, $p_{B,j}$ represents the jth associated point of $p_{A,i}$, LBR is a network composed of a Fully-connected (FC) layer, Batch Normalization layer, and activation function, the input and output dimension of LBR is 3 and 64, MP represents maximum pooling, f_i represents feature of $p_{A,i}$ and its associated point after feature aggregation.

The existing feature aggregation methods generally use k-means as the condition for selecting neighboring points, but k-means only uses distance as the clustering condition. However, the data of hazardous goods point clouds are relatively precise and have the characteristic of similar local features. Therefore, using only distance information for clustering will result in errors.

To solve the above problems, this paper proposes a curve clustering method to obtain global features. For the point $p_{A,i}$ in A, a series of points related to it in B are selected, called the intra-class points of $p_{A,i}$. In the selection process, a curve is used to connect $p_{A,i}$ and its intra-class points, and then a state descriptor is used to describe the current state of the curve. The next forward direction of the curve is determined based on the current state of the curve, then the next point is selected until $p_{A,i}$ contains enough intra-class points.

First, select the initial point $p_{A,i} \in \mathbb{R}^3$, and the process of calculating the initial feature descriptor is shown in formula 5:

$$s_0 = LBR(p_{A,i}). \quad \textbf{(5)}$$

The input dimension of LBR is 3 and the output dimension is 16. Then m nearest neighbor points are searched around $p_{A,i}$ as the center in B, and the score is calculated for each point selected, as shown in formula 6:

$$\alpha_j = LBR_2(MP(s_0, LBR_1(p_{B,1,j}))), j = 1, 2, ..., m. \quad \textbf{(6)}$$

The input dimension of LBR_1 is 3 and the output dimension is 16, the input dimension of LBR_2 is 16 and the output dimension is 1. Here, $p_{B,1_j}$ represents the j-th point in the neighborhood when selecting the first point, and α_j is the score of $p_{B,1_j}$. The point with the highest score is selected as the first point, $p_{B,1} = p_{B,1,\text{argmax}(\mathit{soft}\text{max}(\alpha))}$. After selection, the curve contains two points, so the state descriptor of the curve needs to be updated, as shown in formula 7:

$$s_1 = MP(s_0, LBP(p_{B,1})). \quad \textbf{(7)}$$

The input dimension of LBR is 3 and the output dimension is 16. Assuming that k points have already been selected, when selecting next point, m nearest neighbor points are searched around $p_{B,k}$ as the center in B. The process of calculating score of each neighboring point and updating the state descriptor is shown in formulas 8 to 10:

$$\alpha_j = LBR_2(MP(s_k, LBR_1(p_{B,k+1,j}))), j = 1,2,...,m. \tag{8}$$

$$p_{B,k+1} = p_{B,k+1}, \arg\max(soft\max(\alpha)). \tag{9}$$

$$s_{k+1} = MP(s_k, LBR_1(p_{B,k+1})). \tag{10}$$

The number of intra-class points for each point in set A should be determined based on the number of points in the original point cloud. Through experiments, this paper selects 32 intra-class points for each point in set A.

The process of selecting points on the curve is essentially to score all possible next points based on the current state of the curve, then select the point with the highest score as the class point. However, such naive approach can easily lead to loops as formula 8 will always have the same output for the same state. The loops that the curve may produce are shown in Fig. 3:

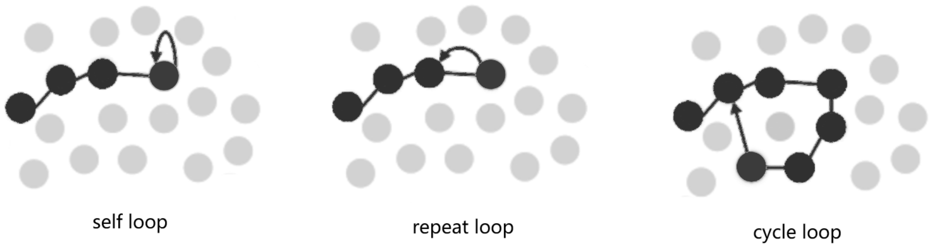


Fig. 3. Four possible loops

(The black points denote intra-class points, the grey points denote points in B except intra-class points)

For self loop, this can be avoided by excluding the self-point. For other possible loops, this paper avoids them by adjusting the direction of curve progression. Assuming the currently selected point is $p_{B,k}$, and the current direction of curve progression is $\vec{\beta}$, we calculate the direction vector $\overrightarrow{\beta_J}$ from $p_{B,k}$ to each candidate point in its neighborhood, and then calculate the directional weights for each neighborhood point by formula 11:

$$d_j = \min(1 + \cos(\vec{\beta}, \overrightarrow{\beta_J}), 1). \tag{11}$$

It can be seen in Formula 10 that the smaller the angle between the curve's direction of travel and the candidate point's direction of travel, the smaller the corresponding directional weight. When the two vectors are completely opposite, $d_j = 0$. The directional weight can change the score of each candidate point to avoid the loops, and Formula 8 becomes formula 12:

$$\alpha_j = d_j \cdot LBR_2(MP(s_k, LBR_1(p_{B,k+1,j}))), j = 1,2,...,m. \tag{12}$$

The above method selects 32 intra-class points for each point in set A. Formula 13 combines the point in A and its intra-class points into a feature vector:

$$f_i = MP(LBR(p_{A,i}, p_{B,k} \left| k = 1,2,...,32 \right.)). \tag{13}$$

where $p_{A,i}$ represents the ith point in set A, $p_{B,k}$ represents the kth intra-class points of $p_{A,i}$, The input dimension of LBR is 3 and the output dimension is 64. MP represents max pooling, f_i is the merged feature vector of $p_{A,i}$ and its intra-class points.

3.3 Self-Attention Classification Network

The commonly used method for point cloud classification is PointNet, which has achieved high accuracy. However, it simply extracts features from individual points, and thus loses the joint features of each point with its surrounding points.

Inspired by A. Vaswani [19], this paper proposes a method that uses multi-head self-attention modules to extract global features of point cloud for classification. The self-attention (SA) module is applied in the encoder part shown in Fig. 1, and it has inputs and outputs of the same dimension. Fig. 4 shows the SA calculation process in one dimension.

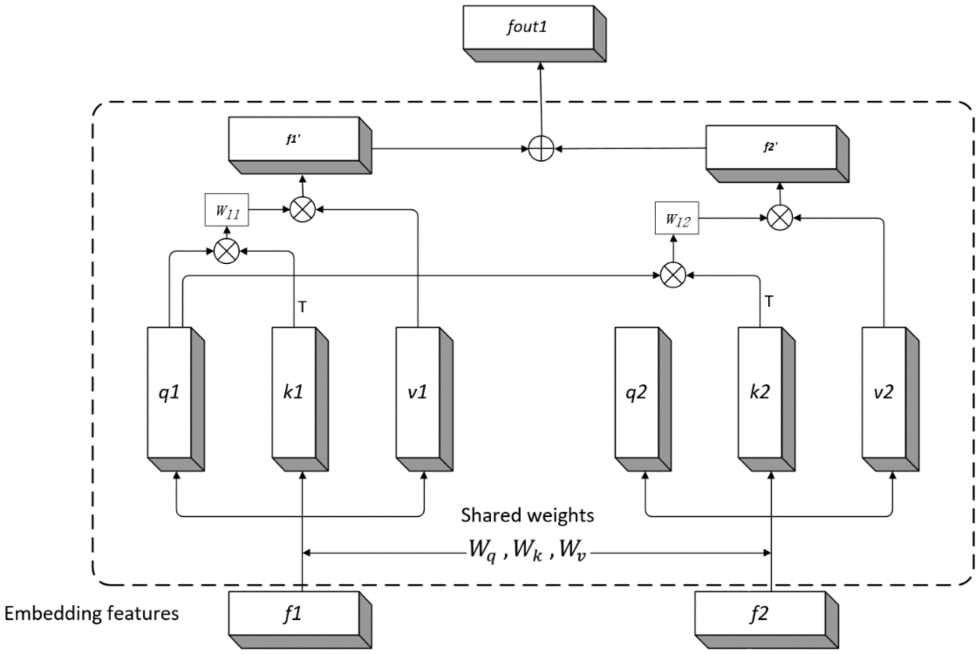


Fig. 4. Details of a SA in one dimension

Due to space limitations, only two features are shown in the Fig. 4. In the figure, f_1 and f_2 feature vectors after embedding, and each is multiplied with three weight matrices to obtain the query vector q, key vector k , and value vector v for each feature. For feature f_1 , the attention weight is calculated by multiplying $q1$ with the transpose of its own key $k1$, and then the weight feature f_1' is obtained by multiplying the weight and its own $v1$. Using the query vector $q1$ of f_1 and the k , v of each feature, the corresponding weight feature is calculated, and all weight features are summed to obtain fout1, which is the output of f_1 after the SA module.

When using the SA module on point cloud data, the entire point cloud can be regarded as a sentence, where each embedded feature is treated as a word. Assuming that in the previous section, the output of point embedding is a feature map $F_e \in \mathbb{R}^{N\times d_e}$ consisting of N vectors with d_e dimensional. For F_e , the SA module calculates the correlation between each feature and the similarity between different features. SA first extracts the query matrix Q, key matrix K, and value matrix V for the point cloud, as shown in formula 14:

$$Q,K,V = F_e \cdot (W_q, W_k, W_v). \tag{14}$$

Where $Q, K \in \mathbb{R}^{N\times d_a}$, $V \in \mathbb{R}^{N\times d_e}$, W_q, $W_k \in \mathbb{R}^{d_e\times d_a}$, $W_v \in \mathbb{R}^{d_e\times d_e}$, W_q, W_k, W_v are learnable linear transformations that share weights, d_a is the dimension of the query and key matrices, where d_a and d_e do not need to be equal. Then, Q and K are used to calculate the attention weights between different features to obtain the attention matrix as shown in formula 15:

$$\Phi' = Q \cdot \frac{K^T}{\sqrt{d_e}}. \tag{15}$$

Where *Φ'* is the attention matrix. Softmax is used for each row in *Φ'* and the result *Φ* is called weight matrix, which is shown in formula 16:

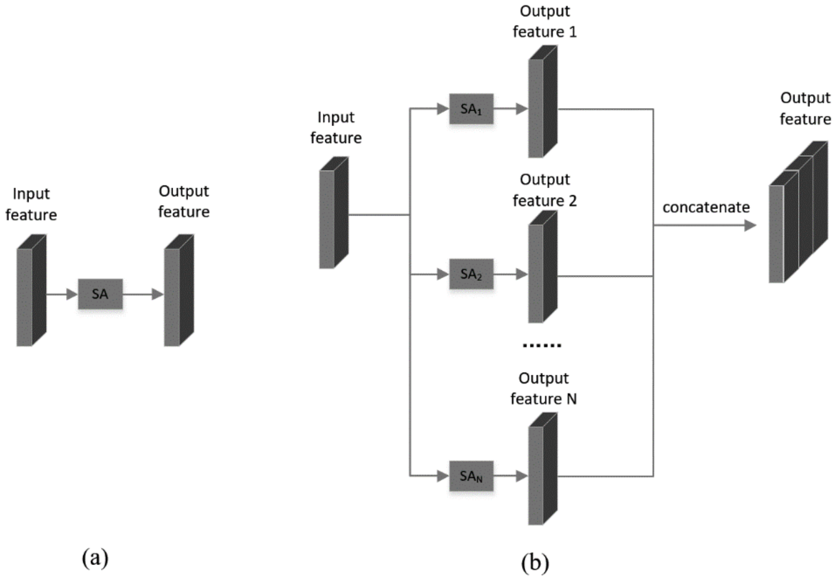
$$\varphi_{i,j} = soft\max(\varphi'_{i,j}) = \frac{\exp(\varphi'_{i,j})}{\sum_k \exp(\varphi'_{i,k})}. \tag{16}$$

The output of the SA is calculated by multiplying the weight matrix with the value matrix, as shown in formula 17:

$$F_{out} = \Phi \cdot V \ . \tag{17}$$

SA aggregates features of points in different positions in space based on their attention. However, weight matrices *Φ* may be different between different point cloud classes, while SA shares weights during processing. Therefore, a multi-head self-attention (MHSA) module is used, which uses multiple independent SA modules to extract features of point clouds under different weight matrices.

The processing of each SA in MHSA is the same as the above process, while different SA modules are independent of each other. Each feature is allocated attention only within the corresponding SA module, so it can be processed in parallel, as shown in Fig. 5:



(a) The calculation process of single SA module (b) The calculation process of multiple SA modules

Fig. 5. The calculation process of single SA module and multiple SA modules

The number of attention heads N_h is set to 8 after experiments. The dimension of final feature is $8d_e$, which is different from the input, so a 3-layer fully connected neural network with neuron numbers of $4d_e$, $2d_e$ and d_e is used to reduce the dimension to d_e . The output of the MHSA module is $F_{outm} \in \mathbb{R}^{N \times d_e}$.

F_{outm} is an aggregated feature that can be decoded using the decoder to obtain the point cloud classification result. Conceptually, the operation of decoder can be formulated as formula 18:

$$c_P = soft\max(LBR(LBR(LBR(MP(F_{outm}))))). \quad \textbf{(18)}$$

where MP represents max pooling, c_p is the probability distribution of classes. First, the output of the encoder is subjected to max pooling to retain the most salient features. Then, a three-layer fully connected neural network is used for decoding, and a softmax layer is used to output the probability distribution of classes, with the class with the highest probability being taken as the classification result of the object.

4 Experiments

The hardware environment for this experiment includes an Intel Core i9-10900k CPU, a Geforce RTX 3090 GPU, and 32GB of RAM. The environment is Windows 10, python 3.8.3, PyTorch 1.10.1, and CUDA version 11.3.

The ModelNet40 dataset [20] and a self-built dataset of hazardous goods in a ware-house were used for comparison experiments in this paper. The ModelNet40 dataset contains 12311 CAD models of artificial objects, and each has a corresponding class, for a total of 40 classes. 9,843 models in the dataset were used for training, and 2,486 models were used for testing. The self-built dataset of hazardous goods in a warehouse contains eight categories, including five hazardous goods: oil drums, cardboard boxes, glass bottles, ceramic bottles and steel bottles, as well as three non-hazardous items: tables, chairs, and people.

In terms of parameter settings, the optimization method was stochastic gradient de-scent. The training epoch was set to 250 as well as the batch size was set to 32, and the learning rate was exponentially decayed from 0.1 to 0.0001. Batch normalization and ReLU activation functions were used for each network layer, and a dropout layer with a parameter of 0.3 was added to each layer in the decoder.

4.1 Affine Transformation Analysis

Affine transformation normalizes the distribution of the object and reduces errors caused by differences in point cloud distribution and geometry. In this paper, we com-pared our method with some existing point cloud classification methods, with and with-out using affine transformation, on the ModelNet40 dataset. The evaluation metrics mainly include Overall Accuracy (OA) and mean Accuracy (mAcc). The experiments show that affine transformation can improve classification accuracy to some extent. The result is shown in Table 1.

Table 1. OA and mAcc of methods before and after affine transformation

Method	Before affine transformation		After affine transformation	
	OA (%)	mAcc (%)	OA (%)	mAcc (%)
PointNet	89.2	86.2	89.3	85.2
PointMLP	92.7	90.2	93.2	90.5
The method of this paper	93.1	92.3	93.7	92.9

4.2 Number of Attention Heads

To investigate the mechanism of the attention heads, this study also conducted re-search on different numbers of heads. Sangeetha [21] shows that the purpose of using multi-head attention is to increase the distribution of different attentions and improve classification accuracy. Therefore, it is necessary to ensure that the attention positions of different heads are different. Attention heads are essentially vectors, so distance can be used to measure the differences in attention levels between different heads. If the distance between two heads is small, it means that the positions they focus on are similar, and vice versa. Fig. 6 shows the trend of the average distance between different heads and the accuracy of the final classification as the number of heads increases.

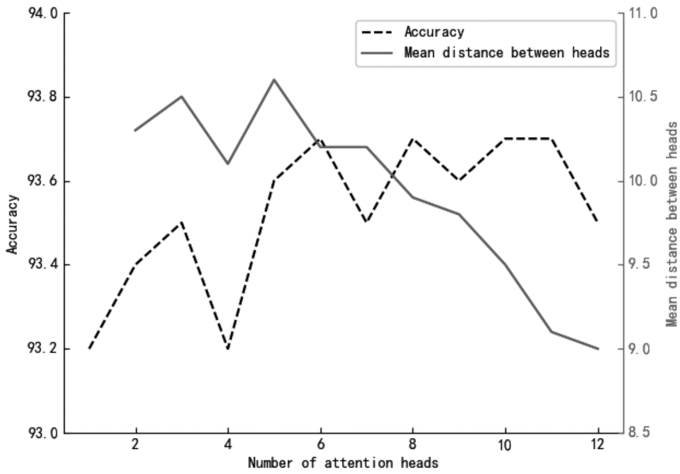


Fig. 6. The change of accuracy and mean distance between heads on different number of heads

(The solid line represents the mean distance between heads, and the dashed line represents the accuracy.)

From Fig. 6, it can be seen that the classification accuracy improves as the number of attention heads increases, reaching 93.7% when the number reaches 8. As the number of heads continues to increase, the distance between the heads decreases, indicating that the attention positions between different heads overlap to some extent. However, the accuracy no longer improves, and the speed of classification decreases due to the increased parameter amount. Therefore, this paper uses 8 attention heads.

4.3 Results on Public Datasets

In the ModelNet40 dataset, there are 40 categories of point clouds. The performance of the this paper is compared with other classification methods. The results show that the proposed methods has the best performance in both OA and mAcc. The specific results are shown in Table 2.

Table. 2. OA and mAcc on the ModelNet40 dataset

Method	Input	OA (%)	mAcc (%)
MVTN	Points, Figures	93.5	92.2
MVCNN	Points, Figures	90.1	-
VoxNET	Voxel	85.9	83.0
PVAFA	Points, Voxel	93.4	87.3
PointNet	Points	89.2	86.2
LDGCNN	Points	92.9	90.2
The method of this paper	Points	93.7	92.9

From Table 2, it can be seen that compared with the voxel-based classification method VoxNet, the feature aggregation method used in this paper avoids the loss of detailed features caused by voxelization, thus the overall classification accuracy is improved by 7.8%. Compared with the multi-view classification method MVCNN, the method of this paper directly processes the points, avoiding the feature loss during the process of converting 3D point clouds into 2D planes, and improving the overall classification accuracy by 2.6%. Compared with the point-based classification methods PointNet and LDGCNN, this paper considers the correlated features between points, improving the overall classification accuracy by 4.5% and 0.8%. Compared with PVAFA, the method of this paper improves mAcc by 5.6%.

4.4 Results on Self-built Datasets

In the self-built hazardous goods warehouse dataset, there are a total of 8 classes of point clouds. The performance of this paper is compared with other classification algorithms using OA and mAcc as evaluation metrics. The results show that the proposed method has the best performance in terms of OA and mAcc. The specific results are shown in Table 3.

Table. 3. OA and mAcc on the self-built dataset

Method	Input	OA (%)	mAcc (%)
MVTN	Points, Figures	78.3	77.4
MVCNN	Points, Figures	76.8	73.9
VoxNET	Voxel	72.1	70.4
PVAFA	Points, Voxel	78.4	76.9
PointNet	Points	75.9	72.7
LDGCNN	Points	72.9	70.4
The method of this paper	Points	78.4	77.3

From Table 1, it can be seen that compared with the classical method PointNet, the method of this paper improves the overall accuracy by 2.5%. Compared with voxel-based methods, the method of this paper improves the overall accuracy by 6.3%. Compared with multi-view classification methods, the method of this paper improves the overall accuracy by 0.1% to 1.6%.

5 Conclusions

This paper proposes a point cloud classification method based on multi-head self-attention, which effectively solves the problems of large space, sparse distribution of point cloud image data and similar distribution under low dimension of hazardous chemical goods warehouses. Compared with some classical or latest point cloud classifi-cation methods on ModelNet40 and self-built hazardous goods warehouse data sets, it has achieved good experimental results. Next, we can further explore a simplified global attention method and streamline the network to obtain a more lightweight classification network that can be deployed on the industrial edge with high real-time requirements.

Acknowledgement

This research was funded by National key research and development program (No. BIPTACF-008) and the National Science and Technology Major Project of the Ministry of Science and Technology of China (No. 2018AAA0102900).

References

[1] B. Wang, D. Li, C. Wu, Characteristics of hazardous chemical accidents during hot season in China from 1989 to 2019: A statistical investigation, Safety Science 129(2020) 104788.

[2] Y. Li, H. Wang, K. Bai, S. Chen, Dynamic intelligent risk assessment of hazardous chemical warehouse fire based on electrostatic discharge method and improved support vector machine, Process Safety and Environmental Protection 145(2021) 425-434.

[3] K.-S. Rajasree, K. Karlsson, T. Ray, S.-N. Chormaic, 1.6 GHz Frequency Scanning of a 482 nm Laser Stabilized Using Electromagnetically Induced Transparency, IEEE Photonics Technology Letters 33(15)(2021) 780-783.

[4] J. Dufek, X.-S. Xiao, R.-R. Murphy, Best Viewpoints for External Robots or Sensors Assisting Other Robots, IEEE Transactions on Human-Machine Systems 51(4)(2021) 324-334.

[5] S. Zhang, M.-Y. Li, M.-G. Jian, Y.-J. Zhao, F.-F. Gao, AIRIS: Artificial intelligence enhanced signal processing in reconfigurable intelligent surface communications, China Communications 18(7)(2021) 158-171.

[6] A. Srinivasan, B. Lee, J. Stasko, Interweaving Multimodal Interaction With Flexible Unit Visualizations for Data Exploration, IEEE Transactions on Visualization and Computer Graphics 27(8)(2021) 3519-3533.

[7] J.-J. Wang, Y.-F. Zhong, Z. Zheng, A.-L. Ma, L.-P. Zhang, RSNet: The Search for Remote Sensing Deep Neural Networks in Recognition Tasks, IEEE Transactions on Geoscience and Remote Sensing 59(3)(2021) 2520-2534.

[8] L. Li, Z. Li, S. Liu, H.-Q. Li, Occupancy-Map-Based Rate Distortion Optimization and Partition for Video-Based Point Cloud Compression, IEEE Transactions on Circuits and Systems for Video Technology 31(1)(2021) 326-338.

[9] Y. Kim, I. Alnujaim, D. Oh, Human Activity Classification Based on Point Clouds Measured by Millimeter Wave MIMO Radar With Deep Recurrent Neural Networks, IEEE Sensors Journal 21(12)(2021) 13522-13529.

[10] L.-X. Zhan, W. Li, W.-D. Min, FA-ResNet: Feature affine residual network for large-scale point cloud segmentation, International Journal of Applied Earth Observation and Geoinformation 118(2023) 103259.

[11] A. Hamdi, S. Giancola, B. Ghanem, MVTN: Multi-View Transformation Network for 3D Shape Recognition, in: Proc. 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021.

[12] H. Su, S. Maji, E. Kalogerakis, E. Learned-Miller, Multi-View Convolutional Neural Networks for 3D Shape Recognition, in: Proc. 2015 IEEE International Conference on Computer Vision (ICCV), 2015.

[13] D. Maturana, S. Scherer, VoxNet: a 3D convolutional neural network for real-time object recognition, in: Proc. 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015.

[14] L.-F. Zhu, C.-W. Lin, C. Zheng, N.-H. Yang, Point-Voxel Adaptive Feature Abstraction for Robust Point Cloud Classification. <https://arxiv.org/abs/2210.15514>, 2022 (accessed 30.Oct.2022).

[15] R.-Q. Charles, H. Su, M. Kaichun, L.-J. Guibas, PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, in: Proc. 2017 IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[16] K.-G. Zhang, M. Hao, J. Wang, C.-W. Silva, C.-L. Fu, Linked Dynamic Graph CNN: Learning on Point Cloud via Linking Hierarchical Features. <https://arxiv.org/abs/1904.10014>, 2019 (accessed 6.Aug.2019).

[17] X. Ma, C. Qin, H.-X. You, H.-X. Ran, Y. Fu, Rethinking Network Design and Local Geometry in Point Cloud: A Simple Residual MLP Framework. <https://arxiv.org/abs/2202.07123>, 2022 (accessed 29.Nov.2022).

[18] T. Hussain, K. Muhammad, A. Ullah, J.-D. Ser, A.-H. Gandomi, M. Sajjad, S.-W. Baik, V.-H. Albuquerque, Multiview Summarization and Activity Recognition Meet Edge Computing in IoT Environments, IEEE Internet of Things Journal 8(12)(2021) 9634-9644.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.-N. Gomez, L. Kaiser, I. Polosukhin, Attention Is All You Need, in: Proc. 2017 the International Conference on Advances in neural information processing systems, 2017.

[20] Z.R. Wu, S.R. Song, A. Khosla, F. Yu, L.G. Zhang, X.O. Tang, J.X. Xiao, 3d shapenets: A deep representation for volumetric shapes, in: Proc. 2015 Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[21] K. Sangeetha, D. Prabha, RETRACTED ARTICLE: Sentiment analysis of student feedback using multi-head attention fusion model of word and context embedding for LSTM, Journal of Ambient Intelligence and Humanized Computing 12(2021) 4117-4126.