Classification Algorithm of Urban Point Cloud Data based on LightGBM

Xiaosong Shi*, Yinglei Cheng¹ and Doudou Xue¹

¹Information and Navigation College, Air Force Engineering University, Xi’an, Shaanxi Province, 710077, China

*Corresponding author’s e-mail: shixiaosong321@163.com

Abstract. In order to improve the accuracy and efficiency of airborne LiDAR point cloud data classification algorithm, a classification algorithm of point cloud based on LightGBM was proposed, and the classification effect of the algorithm on urban point cloud data was tested. In this paper, LightGBM-1 classifier was used to roughly classify point cloud data firstly. Then ground points were extracted to normalize non-ground points. After that, multi-scale neighborhood features of building points and vegetation points were extracted, and then building points and vegetation points were finely classified by LightGBM-2 classifier. The algorithm was verified by urban point cloud data, and the classification effect was evaluated by analyzing classification accuracy and time. Experimental results show that, compared with other algorithms, this algorithm can effectively improve the effect of point cloud data, and realize the effective classification of point cloud data in urban areas.

1. Introduction

With the wide application of LiDAR (Light Detection and Ranging) in the field of mapping remote sensing technology, the classification of three-dimensional point cloud data has become a research hotspot in the field of mapping remote sensing technology[1]. At present, there are still many problems in point cloud classification. Point cloud data are randomly arranged in three-dimensional space, and there is no necessary connection between points and no necessary semantic information for classification. In addition, the urban environment is complex with staggered distribution of ground and objects. There are many problems such as building shading during the LiDAR scanning, and the point cloud data obtained are lack of information. Aiming at the above problems, designing classification algorithms and classifying large-scale urban point cloud data are hotspot and difficulties in current research[2].

Early researchers used morphological filtering, irregular triangular filtering and other point cloud classification methods[3]. From the 1990s, modern machine learning theory began to take shape gradually, and entered a vigorous development stage at the beginning of this century[4]. During this period, researchers put forward a series of excellent machine learning algorithm models, such as Support Vector Machine (SVM), Boosting series models, Random Forests (RF) and so on, and keep improving them. With the development of machine learning, more and more researchers have applied machine learning methods to classification of point cloud data in recent years. Anguelov et al.[5] used markov random field model to perform multivariate classification of point cloud data. Dong et al.[6] used the decision tree method to divide point cloud data into different categories, such as buildings, trees and ground. Lodha[7] et al. used extended support vector machine to complete the fine classification of point cloud data. Through continuous research, the existing algorithm model makes
use of the powerful self-learning ability of machine learning models to effectively improve the self-learning ability of point cloud classification algorithm and reduce the complexity of the algorithm. However, it remains to be further studied to reduce computation and improve the accuracy of LiDAR point cloud classification.

This paper designs a classification algorithm based on LightGBM model proposed by Ke[8] et al in 2017, which is used for fine classification of urban point cloud data. LightGBM algorithm model, which is a Boosting ensemble learning algorithm proposed based on the series of algorithm models, belongs to the series of GBDT[9] (Gradient Boosting Decision Tree) algorithm. LightGBM algorithm can effectively reduce the amount of computation while ensuring good accuracy. In large-scale urban point cloud data classification process, there are many problems such as large computation amount and low classification accuracy. Therefore, this paper aims to use the advantages of LightGBM model, such as fast parallel computing speed and high classification accuracy, to conduct fine classification of urban point cloud data, reduce the computation and improve the classification accuracy.

2. Algorithm flow and principle

2.1. Algorithm flow

In this algorithm, LightGBM model is used as a classifier, and point cloud data in urban areas are used as classification objects for fine classification. Point cloud data are classified into four categories: ground, vegetation, buildings and man-made objects (such as cars). The algorithm is divided into two parts. The first part is rough classification process. LightGBM classifier is trained by using existing features of training data set to form feature vectors. The classifier will classify the input point cloud data into four categories: ground, vegetation, buildings and man-made objects. According to ground points, the elevation value of object points is normalized. The purpose of first part is to extract ground points for normalization of object points, and extract vegetation points and building points for fine classification. The second part is the process of fine classification. The normalized building points and vegetation points are extracted to form the new point set. Extraction of multi-scale neighborhood features is carried out for the new point set, and neighborhood features is added to the feature vectors. LightGBM classifier is trained with new data to conduct secondary classification of building points and vegetation points. The second part aims to solve the problem of misclassification due to limited features and improve the classification accuracy of building points and vegetation points. The specific flow chart is shown in figure 1.

![Algorithm flow chart](image-url)
2.2. Classifier principle
Like other GBDT algorithms (such as XGBoost[10]), LightGBM still uses the method of fitting negative gradient when training decision tree[11]. Aiming at the problem of low efficiency and large computation amount of previous GBDT algorithm, LightGBM introduces improved mechanisms, such as GOSS (Gradient-based One-Side Sampling) and EFB (Exclusive Feature Bundling), to solve the defects of the previous GBDT algorithm, which is time-consuming and inefficient in the face of high feature dimensions or large data volume. GBDT algorithm approximates the original objective function with Taylor expansion, and takes the first and second derivative of the loss function in the expansion. The approximate objective discriminant function is shown in equation 1.

$$Obj = \sum_{i=1}^{n} [g_i f(x_i) + \frac{1}{2} h_i f^2(x_i)] + \Omega(f)$$

In the formula, $g_i$ and $h_i$ are the first and second derivatives of the loss function respectively, and the objective discriminant function is the sum of Taylor expansion and regularization terms of all the samples. The derivative of the loss function and the residual error fitting in the direction of negative gradient of the loss function make the objective decision function smaller, and the decision tree with better structure is obtained.

GOSS is a way to maintain a balance between reducing training data and ensuring accuracy. It retains all samples with large gradient and randomly samples with small gradient, which is similar to the principle of Adboost[12] algorithm. GOSS retains some small gradient samples through sampling to prevent data distribution from changing and affect training accuracy. The principle of GOSS is shown in equation 2.

$$G = \sum_{A \in A} g_i + \frac{1-a}{b} \sum_{A \in B} g_i$$

In the formula, $A$ is the large gradient sample set and $B$ is the small gradient sample set after sampling. Before sampling, the samples are arranged in descending gradient order. The proportion $a\%$ is set to select the sample set $A$ and the remaining samples are randomly sampled at the sampling rate of $b\%$ to get the sample set $B$. $A$ and $B$ constitute the new training samples. EFB makes use of the mutual exclusion of many features in the high-dimensional sparse feature space to bind mutually exclusive features into a single feature, reducing the amount of features to be calculated. In addition, leaf-wise and parallel strategies ensure LightGBM efficiency. Studies show that the training speed of LightGBM is more than 20 times faster than the training speed of previous GBDT algorithm, with almost the same accuracy. LightGBM has the characteristics of high precision and high efficiency when dealing with large data volume, and it is capable of dealing with the classification of urban point cloud data.

2.3. Normalization and neighborhood feature extraction
Urban terrain environment is complex, and the changes of terrain will make the elevation value difference of the same point. The change of elevation will lead to the misclassification of point cloud data. In addition, due to the dense distribution and shading between buildings and vegetation, the point cloud data between buildings and vegetation will generate misclassification. In order to eliminate the influence of the above factors, this paper normalizes object points based on ground points to eliminate the influence of topographic relief [13]. At the same time, the extraction method of neighborhood feature is used to comprehensively analyze the information of all the points in the neighbourhood. The process is done to improve the classification accuracy of buildings and vegetation. The specific methods of normalization are as follows:

**Step1**: Traverse $Pn(n=1,2,3...,)$, judge whether the current point $Px$ is ground object point. If $Px$ is ground object point, execute Step2; otherwise, execute Step3. If traversal is completed, execute Step4;
Step2: Use the nearest search method to find the nearest ground point of the current ground object point \( P_x \). The elevation value \( Z \) of the land object point subtracted from the elevation value \( Z_0 \) of the nearest ground point to obtain the normalized elevation value \( Z_{\text{Norm}} \), and continue to execute Step1;

Step3: Set the elevation value of the current ground point \( P_x \) to 0, and continue to execute Step1;

Step4: After traversing, get \( P^*n(n=1,2,3,\ldots) \). The normalized point cloud computing formula is shown in equation 3.

\[
Z_{\text{Norm}} = Z - Z_0
\] (3)

In the formula, \( Z \) is the elevation value of the object point, \( Z_0 \) is the elevation value of the ground point closest to the object point, and \( Z_{\text{Norm}} \) is the elevation value of the object point after normalization.

The process of fine classification combines features of multi-scale neighborhoods after extracting neighborhood features of each point, and the new feature vector reflects the features of this point in different scale neighborhoods. The obvious differences of multi-scale neighborhood features among various objects can enhance the separability of objects[14]. LightGBM classifier is trained by combination of multi-scale features and single point features, and the building points and vegetation points are finely classified to improve the classification accuracy.

3. Experiment and result analysis

3.1. Experimental data and parameter setting

The Vaihingen urban LiDAR data set, provided by the international society for photogrammetry and remote sensing, was used to test the effectiveness of the classification algorithm. The average point distance of the data set was 0.66m. The classification algorithm divided the point cloud data into four categories: ground, buildings, vegetation and man-made objects. There were 57,302 discrete points in the data set in this paper. The data set was divided into 90% training sets (including 20% validation sets, which were used to verify whether the classification performance of the LightGBM model in the training process reaches the anticipated standard) and 10% test sets. The training set was used to train the classifier model, the test set was used to test the classification effect of the model, and the categories of point cloud in the data set had been manually marked. In this paper, point cloud data with an area of 22995m\(^2\) (including 57,302 discrete points) was selected as the test set to verify the classification accuracy of algorithm model in this paper. In the experiment, sklearn module in Python[15] was used to build LightGBM classification model, and indicators such as precision, recall and F1-score were provided to evaluate the classification effect. The training degree of the model was adjusted by setting learning_rate, num_leaves, max_depth and other major parameters. By setting the parameters, the classification accuracy was improved, the over-fitting problem was prevented, and the training speed is increased as much as possible.

The purpose of rough classification was to extract ground points for the next step of point cloud normalization, and to separate building points and vegetation points for the next step of fine classification. In order to find the optimal parameters, multiple alternative values should be set for the main parameters. Alternative num_leaves values are \{10, 50,100,150,200\}, alternative max_depth values are \{4,6,8\}, alternative learning_rate values are \{1,0.1,0.01\}, and other parameters were set as default values. Through the GridSearchCV module in sklearn, different parameters were searched automatically, and the optimal parameter combination was selected as the model parameter through the cross-validation of the validation set. Through experimental analysis, the rough classification model trained a total of 100 decision trees, num_leaves was 100, max_depth was 8, and learning_rate was 0.1. After iterating all training data by 10 turns, the rough classification model lightgbm-1 was obtained. During fine classification, elevation variance and echo intensity variance were extracted as neighborhood features. Neighborhood features of different scales were selected according to the size of vegetation and building samples. Three scales, which were 50 points, 100 points and 200 points, were selected according to the size of vegetation, bungalow and building in the samples. Multi-scale
neighborhood features and single point features were used to train the fine classifier. Through GridSearchCV module analysis, num_leaves was 1000, max_depth was 10, and learning_rate was 0.1.

3.2. Experimental results analysis
In order to test the classification performance of the model, F1-score (obtained by precision and recall) was used to evaluate the classification accuracy of the model on the test set, and time was used to evaluate the training and classification speed of the model. In order to evaluate the effectiveness of normalization processing and multi-scale neighborhood features, the experiment finely classified the three types of data: point cloud without normalization, point cloud after normalization and point cloud with neighborhood features after normalization, and evaluates the classification effect, as shown in table 1.

| Category     | Point cloud without normalization | Point cloud after normalization | Point cloud with neighborhood features after normalization | Rough classification |
|--------------|----------------------------------|--------------------------------|----------------------------------------------------------|-----------------------|
| Vegetation   | 92.1%                            | 94.7%                          | 96.1%                                                     | 91.8%                 |
| Building     | 94.5%                            | 95.0%                          | 95.9%                                                     | 94.1%                 |

It can be seen from table 1 that the classification accuracy of object points can be effectively improved by removing the influence of terrain factors through normalization. After extracting multi-scale neighborhood features and finely classification, most of the misclassification points were corrected, and the classification accuracy of vegetation points and building points was improved.

In order to verify the performance of the classification algorithm in this paper, the overall classification results of this algorithm were compared with the classification results of SVM classifier and RF classifier to verify whether the classification performance of the algorithm was improved. F1-score and classification time were used to evaluate the classification results, as shown in table 2.

| Algorithm in this paper | Classification accuracy of every category | Times/s |
|-------------------------|------------------------------------------|---------|
| Support Vector Machine  | Ground                                   | 86.2%   |
|                         | Vegetation                               | 81.2%   |
|                         | Building                                 | 84.5%   |
|                         | Artificiality                            | 50.5%   |
|                         | Ground                                   | 86.5%   |
| Random Forests          | Vegetation                               | 87.4%   |
|                         | Building                                 | 88.2%   |
|                         | Artificiality                            | 45.1%   |
|                         | Ground                                   | 97.7%   |
| Algorithm in this paper | Vegetation                               | 96.1%   |
|                         | Building                                 | 95.8%   |
|                         | Artificiality                            | 74.1%   |

It can be seen from table 2 that this algorithm had better classification effect and faster classification speed compared with other machine learning models. The classification results of the three methods were shown in figure 2. The first vertical column was classified as manual classification result, the second vertical column was classified as SVM classification result, the third vertical column was classified as RF classification result, and the fourth vertical column was classified as the algorithm classification result in this paper.
6

Figure 2. Classification results of different algorithms

LightGBM always selects the most effective feature for the growth of decision tree, and there is no need to conduct feature selection before model training, so it can be used for the assessment of feature importance. The assessment of feature importance by lightgbm-1 and lightgbm-2 classifiers was shown in figure 3. It can be seen from figure 3 that the coordinates and echo intensity of point cloud were the most important features, which were the main features of point cloud classification. Other features were less important, which complemented the main features and make finely classification of vegetation points and building points.

Figure 3. Table of feature importance

4. Conclusion
In this paper, an urban point cloud data fine classification algorithm based on LightGBM was proposed, and the algorithm flow and corresponding technical theory were described in detail. The effectiveness of the algorithm was verified by experiments. The results show that the rough classifier can effectively classify point cloud data, and the normalization of point cloud data can eliminate the impact of terrain, and the extraction of multi-scale neighborhood features of buildings and vegetation can further improve the classification accuracy. The fine classification of buildings and vegetation was carried out by a fine classifier, and the effective classification of buildings and vegetation was realized. The algorithm in this paper only used point cloud features, without adding features such as spectrum and texture, and still needed to extract features manually. In the follow-up work, automatic extraction of features can be further studied, considering the combination with new technologies such as deep learning, to further improve the automation and classification accuracy of and classification algorithm.

References
[1] Wu J, Liu R, Guo N, et al. (2013) Aerial LiDAR data classification using weighted support vector machines. Geomatics and Information Science of Wuhan University, 38(1): 1-5.
[2] Axelsson P. (1999) Processing of laser scanner data—algorithms and applications. Isprs Journal of Photogrammetry & Remote Sensing, 54(2–3):138-147.

[3] Sithole G, Vosselman G. (2004) Experimental comparison of filter algorithms for bare-Earth extraction from airborne laser scanning point clouds. Isprs Journal of Photogrammetry & Remote Sensing, 59(1):85-101.

[4] Zhang R, Wang Y B. (2016) Research on Machine Learning with Algorithm and Development. Journal of Communication University of China(Science and Technology), 23(2):10-18.

[5] Anguelov D, Taskar B, Chatalbashev V, et al. (2005) Discriminative Learning of Markov Random Fields for Segmentation of 3D Scan Data. Proc Cvpr, 2:169-176.

[6] Dong B G, Ma H C, Che S, et al. (2016) Method of Land Cover Refined Classification Supported by LIDAR Point Clouds. Remote Sensing Technology and Application, 31(1):165-169.

[7] Lodha S K, Kreps E J, Helmbold D P, et al. (2007) Aerial LiDAR Data Classification Using Support Vector Machines(SVM). In: International Symposium on 3d Data Processing, Visualization, and Transmission. North Carolina. pp. 567-574.

[8] Guolin K, Qi M, Thomas F, et al. (2017) LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In: The 31st Conference on Neural Information Processing Systems. Long Beach. pp. 1-9.

[9] Friedman J H. (2001) Greedy Function Approximation: A Gradient Boosting Machine. The Annals of Statistics, 29(5):1189-1232.

[10] Chen T, Guestrin C. (2016) XGBoost: A Scalable Tree Boosting System. In: The 22nd ACM SIGKDD International Conference. California. pp. 785-794.

[11] Chen L, Zhang T, Li T. (2017) Gradient boosting model for unbalanced quantitative mass spectra quality assessment. In: International Conference on Security. Shenzhen. pp. 394-399.

[12] Freund Y. (1995) Boosting a Weak Learning Algorithm by Majority. Information and Computation, 121(2):256-285.

[13] Liu Y, Dong L, Gao G F. (2015) The Machine Carries the Ground of Laser Radar Data Thing Classification Method and Treatment. Geomatics & Spatial Information Technology, 12:108-110.

[14] Yue C, Liu C J, Wang X F. (2016) Research on point cloud data classification algorithm for high and steep slope based on multi-scale dimension features and SVM. Geomatics and Information Science of Wuhan University, 41(7):882-888.

[15] Komer B, Bergstra J, Eliasmith C. (2014) Hyperopt-Sklearn: Automatic Hyperparameter Configuration for Scikit-Learn. Icml Automl Workshop, 14:33-39.