LIDAR POINT CLOUD CLASSIFICATION USING EXPECTATION MAXIMIZATION ALGORITHM

Nguyen Thi Huu Phuong

Department of Software Engineering, Faculty of Information Technology, Hanoi University of Mining and Geology, Hanoi, Vietnam

ABSTRACT

EM algorithm is a common algorithm in data mining techniques. With the idea of using two iterations of E and M, the algorithm creates a model that can assign class labels to data points. In addition, EM not only optimizes the parameters of the model but also can predict device data during the iteration. Therefore, the paper focuses on researching and improving the EM algorithm to suit the LiDAR point cloud classification. Based on the idea of breaking point cloud and using the scheduling parameter for step E to help the algorithm converge faster with a shorter run time. The proposed algorithm is tested with measurement data set in Nghe An province, Vietnam for more than 92% accuracy and has faster runtime than the original EM algorithm.

KEYWORDS

LiDAR, EM algorithm, Scheduling parameter, LiDAR point elevation, GMM model

1. INTRODUCTION

LiDAR (Light Detection and Ranging) is a survey method that measures the distance to a target by illuminating the target with the surrounding laser and measuring reflected pulses with a sensor. The difference in laser travel time and the laser wavelength can then be used to create digital representations of the target. The essence of LiDAR technology is long laser measurement technology, GPS / INS spatial positioning, and recognition of light reflection intensity [1]. The pulse of the laser is sent to the ground at a certain height. Laser waves are reflected from the ground or from object surfaces such as trees, roads or buildings, with each pulse measuring the time required to transmit and return signals, calculating the distance from the laser source to the object. At each laser pulse generation, the GNSS satellite positioning system will determine the spatial position of the emitting point and the inertial navigation system will determine the directional angles in the space of the scanning beam. With these combined measurements, the position (spatial coordinates) of the points on the ground is calculated [2]. From the set of reflection points, we have a point cloud.

In Vietnam, the application of LiDAR technology in many fields is not uncommon for studies on data and application of LiDAR technology, which has been included in the development policies of many ministries such as Ministry of Construction, Ministry of Information and Communications and Ministry of Resources and Environment. However, LiDAR research in Vietnam only stopped at the application and use the software with equipment. Through the implementation process, the problems that arise are unclassified classification when classified employees still have to use more aerial photos to get classification results as at Ministry of Resources and Environment. In addition, the copyright issues and ability to update are the factors that make LiDAR technology not really popular in our country. So far, studies have been published on the application of LiDAR technology, improvements in technology and data processing algorithms that are constantly published by scientists. The research and application directions of the authors are relatively broad.

LiDAR data is a relatively large data set and has a wide coverage across the scanning area, so sorting as a job is not easy. The point filtering process is a key for almost every application with the LiDAR point cloud. With point filtering algorithms, parameter settings and increased convergence thresholds increase the accuracy of the point filtering process. Based on that idea, the authors group Z.Hui, P.Cheng, Y.Yziggah, Y.Nie proposed the algorithm of non-threshold filtering based on the expected maximization algorithm (EM - Expectation Maximization). The target splits the point cloud into two groups of ground and non-ground. EM is used to estimate the maximum expected for the GMM model parameter (Gaussian Mixture Model). After a number of iterations, the points of the point cloud are labelled, and the authors found that the proposed algorithm worked well for points belonging to the non-ground group, with only 4.48% of errors smaller than 8 algorithms. Point filtering math is reported in ISPRS [5]. In his doctoral dissertation, author Artur Maligo proposed a two-class classification model, the first layer was built on the GMM model and the second consisted of repeating an intermediate layer to select the right grade for original classification purpose. Specifically, the GMM model is identified in the unattended classification training class and defines a set of intermediate classes. Test results show that the proposed system works well on two data sets with an accuracy of 0.8 - 0.89 [6].

The creation of 3D models of urban areas is currently of interest to scientists. For a data set with a high density point, containing a lot of noise, choosing an appropriate algorithm is absolutely necessary. Authors Qing Zhu, Yuan Li, Han Hu, Bo Wu have proposed a point cloud classification algorithm based on multi-level semantic relationships. The input of the algorithm is a point cloud through transformations based on the uniformity of the point and the binding of neighbouring points, which are classified into layers with an accuracy of 93.55%. However, the proposed algorithm is still limited with received noise [7]. The authors Suresh Lodha, David P.

Helmbold, Darren M.itzpatrick used the algorithm to maximize expectations in classifying LiDAR ground scattering data into four groups: roads, grass, tall buildings and trees. To carry out the classification problem, the authors used five characteristics of LiDAR data: altitude, elevation change, laser reflectance intensity and intensity image. With 94% accuracy was obtained for the 8 square mile area. Based on the selected parameters and models, the EM algorithm is appropriate for the classification area [8]. Meanwhile, the team of author Zhenyang Hui showed that the EM algorithm fully meets the requirements of the automatic DTM extraction problem. With the error of the proposed algorithm is 16.78% lower than the traditional PTD algorithm and reduces the system error to 31.95% [9]. The author Nallig Leal gave the observed data set and variable values for the GMM model in the first iteration, with each label c in the initialization cluster k, group of iteration is done by calculating the probability value of each point in each cluster of each iteration j. Then, update the parameters for the model and repeat until the convergence [10].

In the studies mentioned above, the expectation-maximization algorithm is an approach for performing maximum likelihood estimation in the presence of latent variables. It does this by first estimating the values for the latent variables, then optimizing the model, then repeating these two steps until convergence. It is an effective and general approach and is most commonly used for density estimation with missing data, such as clustering algorithms like the Gaussian Mixture Model. In the EM algorithm, the estimation-step would estimate a value for the process latent variable for each data point, and the maximization step would optimize the parameters of the probability distributions in an attempt to best capture the density of the data. The process is repeated until a good set of latent values and a maximum likelihood is achieved that fits the data.

- E-step: perform probabilistic assignments of each data point to some class based on the current hypothesis h for the distributional class parameters;

- M-step: update the hypothesis h for the distributional class parameters based on the new data assignments.

The EM classification algorithm is not based on distance. The algorithm calculates the probability for each observation belonging to each class based on the chosen distribution, the main purpose of the EM classification algorithm is to find classification solutions to maximize the overall probability for classification data with the required number of classes. Therefore, in the classification problem with EM, any difference in the range or scope of the variable selected for the analysis will not affect the classification results [11].

However, the EM algorithm is very sensitive to initialization values and is prone to errors with local minimum. In addition, the algorithm is difficult to focus, and covariance matrices corresponding to one or more components can become an error condition. Wishing to improve one of the EM algorithm's disadvantages, the author improved the EM algorithm by dividing the point cloud into smaller point clouds vertically, initialization model parameter, use EM to conduct point cloud classification (applied to each point cloud part) and assess the degree exactly. To improve the algorithm's convergence time, use the scheduling parameter β, where β is initialized with a very small value (approximately 0) [12].

2. Proposed Method

2.1. Basic EM Algorithm

The EM algorithm is an iterative algorithm to estimate the maximum probability when only a subset of data is available. The algorithm was proposed by Dempster, Laird, and Rubin in 1977. The algorithm is stated as follows:

- Suppose for the sample dataset $X = (X_1, X_2, \ldots, X_n)$, with density conditions, $f_{X|\Phi}(x|\theta)$ with $\theta = \Phi$. We have formula:

$$l(\theta;X) = \log f_{X|\Phi}(X|\theta) \qquad (1)$$

for log-likelihood function [13].

Steps to implement the algorithm [13]:

- Step 1: An initial prediction is made for model parameters and a delivery probability is generated. Sometimes called "E-Step" for the "Expected" distribution
- Step 2: New data observed based on the model – called E step
- Step 3: The probability distribution from step E is refined to include new data. This step is sometimes called "M-step".
- Steps 2 to 4 are repeated until a stable distribution is reached.

The EM algorithm has two main steps: E - step and M – step, and described as follows [14]:
Implementation:

+ E – step: is a step that estimates the distribution of labels for a given model: find $E(Z_{ij})$

- Assumption θ is known and fixed
- A (B) is the x_i event drawn from $f_1(f_2)$

- D the observed points from x_i
- The expected value of z_i is denoted as P (A |D)
- We have the formula for calculating the expectation D belongs to class A

$$P(A|D) = \frac{P(D|A)P(A)}{P(D)} \tag{2}$$

With P(D) = P(D|A)P(A) + P(D|B)P(B)

$= f_1(x_i|\theta_1)\tau_1 + f_2(x_i|\theta_2)\tau_2$ repeat for each x_i

+ M – step: Find the maximum of log-likelihood function θ
Assumption: $\sigma_1 = \sigma_2 = \sigma$; $\tau_1 = \tau_2 = 0.5 = \tau$.

$$L(\vec{x},\vec{z}|\theta) = \prod_{1\le i\le n} \frac{\tau}{\sqrt{2\pi\sigma^2}} \exp\left(-\sum_{1\le i\le 2} z_{ij}\frac{(x_i-\mu_i)^2}{2\sigma^2}\right) \tag{3}$$

$$\mathrm{E}[\log L(\vec{x},\vec{z}|\theta)] = \mathrm{E}\left[\sum_{1\le i\le n}\left(log\tau - \frac{1}{2}log2\pi\sigma^2 - \sum_{1\le j\le 2} z_{ij}\frac{(x_i-\mu_i)^2}{2\sigma^2}\right)\right] \tag{4}$$

$$= \sum_{1\le i\le n}\left(log\tau - \frac{1}{2}log2\pi\sigma^2 - \sum_{1\le j\le 2} z_{ij}\frac{(x_i-\mu_i)^2}{2\sigma^2}\right)$$

Using the result found E (Z_{ij}) in E - step we have the results:

$$\mu_i = \sum_{i=1}^{n} E[z_{ij}]x_i \,/ \sum_{i=1}^{n} E[z_{ij}] \tag{5}$$

The pseudocode of the algorithm was built by the author based on the idea of document [15], [17] as follows:

Input: D_L labelled dataset, D_U unlabelled dataset, parameter model $f(\theta)_\theta$
Output: $A \in D_U$ classified and $f(\theta)$
Initial: distribution rule π_0, parameter θ_0, $\Phi_0(\pi_0, \theta_0)$

Procedure:

1. Calculate the maximum response in each observation:

$$P[z_i = k|x_i, \Phi^{j-1}] = \frac{\pi_k^{i-1} f_{\theta_k^{i-1}}(X_i)}{\sum_{j=1}^{k}\pi_j^{i-1} f_{\theta_j^{i-1}}(X_i)} := \gamma_{i.k}$$

$$\gamma_{i.k}^{y} = \frac{\pi_k^{t-1} f_{\theta_k^{t-1}}(X_i)}{\sum_{k=1}^{K}\pi_k^{t-1} f_{\theta_k^{t-1}}(X_i)} \tag{6}$$

2. Update parameter $\Phi_0(\pi_0, \theta_0)$ to $\Phi_t(\pi_t, \theta_t)$ with :

$$\pi_t := \arg\max_{\pi:\sum\pi_k=1} \sum_{k=1}^{K} \log[\pi_k] \left(\sum_{i=1}^{n} \gamma_{i,k}^{t}\right) \tag{7}$$

And $\theta_k^t := \arg\max_{\theta} \sum_{i=1}^{n} \gamma_{i,k}^t \, log f_{\theta_k}(X_i)$

End

2.2. Proposed EM

Based on the idea of installing EM algorithm, the proposed method is implemented as the following figure 1:

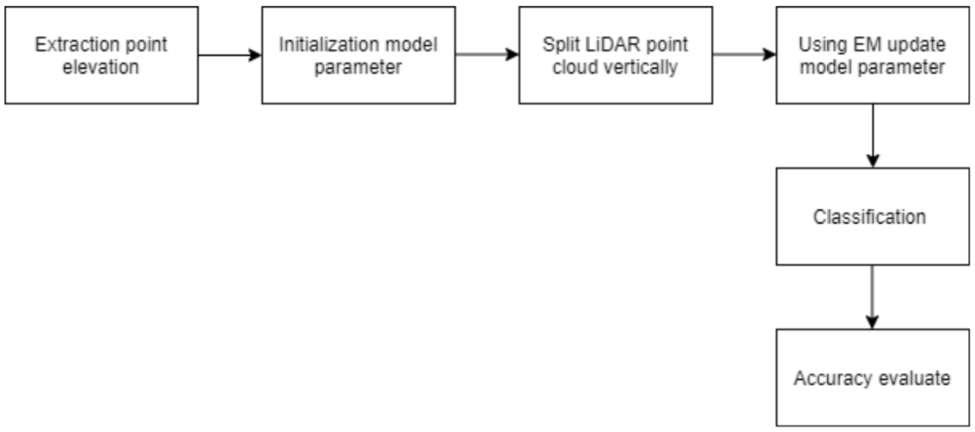

Extraction point elevation
Initialization model parameter
Split LiDAR point cloud vertically
Using EM update model parameter
Classification
Accuracy evaluate

Figure 1. Proposed method

We have LiDAR point cloud P = {P1, P2, …, Pn} with each point Pi in point cloud there is a set of values (Xi, Yi, Zi) show location (X, Y) and elevation (Z) of point. The value that author use to do classification problems is the value of the height Zi of the point (classifying points by height). In the point cloud data set with N points, the width of the data D = 3. The implementation steps of the method are explained as follows:

a. Step 1: Extraction LiDAR point elevation

From the point set in the point cloud, take the point height value as input for the later computation steps, which means we now reduce the width of the data D = 3 → D = 1.

b. Step 2: Initialization model parameter

Suppose there is a data set Z = {Z1, Z2, …, Zn} with Z is the point elevation. We have a probability model for the Gaussian distribution shown as follows:

$$P(Z_i) = \sum_{k=1}^{K} \Pi_k Gaussian(Z_i|\mu, \Sigma_{k,}, \mathrm{Ck}) \quad (8)$$

In which, μ - average value of the data set, C - mixing coefficient, σ - standard deviation, β - scheduling parameter (reduces the convergence time of the algorithm EM) [14], [16].

- Initialization μ, β:
With the value μ after extracting the elevation information of the point in step 1, we will calculate μ by taking the average value of the height of all points in the point cloud according to the formula (2):

$$\mu = \frac{\sum_{i=1}^{n} Z_i}{n} \quad (9)$$

β is initialized with a value of approximately 0. The parameter β can roughly be interpreted as the inverse of temperature [16]. At each β value algorithm iterates E step and M step until convergence.

- Initialization C, δ

The mixing coefficient is the probability to determine the proportion of data belonging to the k component and must ensure conditions:

$$\sum_{k=1}^{K} \mathrm{Ck} = 1 \tag{10}$$

Determining the mixing coefficients for k components of GMM model is calculated by the formula:

$$C_k = \frac{1}{N}\sum_{n=1}^{N} \gamma_k(Z_i) \tag{11}$$

With $\sum_{n=1}^{N} \gamma_k(Z_i)$ is the total number of points in the dataset belongs to the kth component. In this study, author initiates a mixing coefficient for all components of the model to be equal. There, σ is standard deviation which is calculated by the formula:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(r_i)^2} \tag{12}$$

With $r_i = Z_i - \mu$

c. Step 3: Split LiDAR point cloud vertically

With a very large point dataset in the point cloud, updating parameters and converging calculations takes a lot of time. Therefore, the graduate student divides the point cloud vertically based on the height of the point. However, choosing the number of parts to split to prevent the algorithm from falling into a never-ending state requires calculation.

To solve this problem, author took an example on a given point data set (number <3,000), the algorithm converges very well. Therefore, it is recommended that the number of points to be divided should be less than 10.

d. Step 4: Using EM estimate model parameter

- Increase value β by 1 unit for each loop (value of β is not greater than 1, βmax = 1). If β > 1, reset β value = 1.

- At step E of the algorithm calculates the probability that a point Zi belongs to the kth component by the formula:

$$P(k|Z_i) = \frac{(P(Z_i|k)P(k))^{\beta}}{P(Z_i)^{\beta}} \tag{13}$$

With, $P(Z_i) = \sum_{k=1}^{K} P(Z_i|k)P(k)$ (14)

And $P(Z_i|k) = \sum_{k=1}^{K} C_k Gaussian(Z_i|\mu, \Sigma_k)$ (15)

And the Gaussian distribution function is calculated according to the formula:

$$Gaussian(Z_i|\mu, \Sigma_k) = \frac{1}{2\pi\Sigma_k} * e^{-\frac{r_i^2}{2\sigma^2}} \tag{16}$$

- At step M, update the parameters of the model with the values of μ, Σ, C, σ respectively according to the following formulas:

$$\Sigma_k = \frac{\sum_{n=1}^{N} P(k|Z_i)(Z_i-\mu_k)(Z_i-\mu_k)^T}{\sum_{n=1}^{N} P(k|Z_i)} \quad (17)$$

$$\mu_k = \sum_{i=1}^{n} (\frac{P(k|Z_i)*Z_i}{\sum_{i=1}^{n} P(k|Z_i)} \quad (18)$$

$$C_k = \frac{\sum_{i=1}^{n} P(k|Z_i)}{\sum_{k=1}^{K} (\sum_{i=1}^{n} P(k|Z_i)} \quad (19)$$

$$\sigma_k = \sqrt{\frac{\sum_{i=1}^{n} ((Z_i-\mu_k)^2 * P(k|Z_i)}{\sum_{i=1}^{n} P(k|Z_i)}} \quad (20)$$

- Algorithm convergence: the algorithm converges when the new parameter does not change compared to the parameter that exceeds the threshold value ε, with ε calculated by the following formula:

$$\varepsilon = \sqrt{\frac{1}{n}\sum_{i=1}^{n} r_i^2} * 10^{-8} \quad (21)$$

e. Step 5: LiDAR point cloud classification

Based on the probability of a point belonging to a certain k class in the model, we can proceed to classify points. Based on observed data, if P (k | Z_i) ≥ 0.5 then conclude that Z_i belongs to class k.

f. Step 6: Precision evaluating

To evaluate the accuracy of the algorithm, author uses the following measurements: accuracy, recall and F1 measurement of the improved EM algorithm with the basic EM algorithm and the basic EM algorithm.

Pseudocode of algorithm can be performance below:

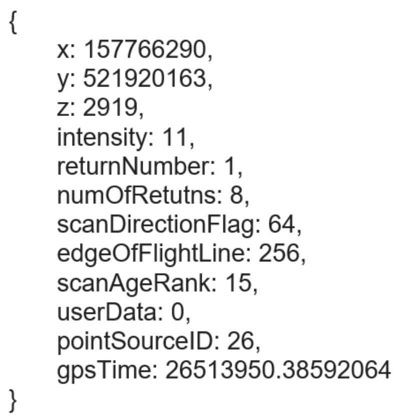
```
Begin
        Input: LiDAR point cloud P = {P1, P2, …, Pn}
        Output: Point cloud classified
        Initialization: the mean - μ, covariance matrix- Σ, mixing coefficient - C, standard deviation - σ, scheduling parameter - β
        Procedure:
                for i = 1 to n
                        for i = 1 to k
                                Increase β up to 1 unit
                                If β < 1 continue E step else set β = 1
                                E – step    $P(k|Z_i) = \frac{(P(Z_i|k)P(k))^\beta}{P(Z_i)^\beta}$
                                M- step     $\Sigma_k = \frac{\sum_{n=1}^{N} P(k|Z_i)(Z_i-\mu_k)(Z_i-\mu_k)^T}{\sum_{n=1}^{N} P(k|Z_i)}$
                                            $\mu_k = \sum_{i=1}^{n} (\frac{P(k|Z_i)*Z_i}{\sum_{i=1}^{n} P(k|Z_i)}$
                                            $C_k = \frac{\sum_{i=1}^{n} P(k|Z_i)}{\sum_{k=1}^{K} (\sum_{i=1}^{n} P(k|Z_i)}$
                $\sigma_k = \sqrt{\frac{\sum_{i=1}^{n}((Z_i - \mu_k)^2 * P(k|Z_i)}{\sum_{i=1}^{n} P(k|Z_i)}}$
            if (convergence) stop loop
            else repeat E and M step

        if P(k|Zi) > = 0.5 Zi belongs to class k
        else Zi not belongs to class k
        Precision evaluating
        End
```

2.3. Experiment

2.3.1. Experiment Database

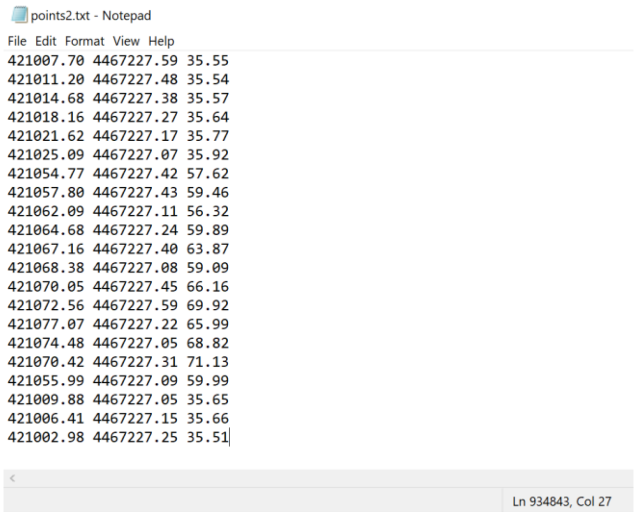
Dataset use in paper is LiDAR point cloud data. Point clouds are a powerful and dynamic information storage technology. By representing spatial data as a collection of coordinates, they can handle large datasets for a wide array of downstream processing. Primarily, in this instance, they are used as a middleman to turn the raw data collected by LiDAR processes into 3D models.

With the LiDAR data was investigated in Dien Chau, Nghe An Province. This is an area with mountainous terrain that occupies two thirds of the area and hilly slopes inclining from the north to the south. Data were collected with the full waveform airborne LiDAR survey. Data was measured from the 29st of September, 2016 to the 29h of September, 2016, with an area of 102 km^2, 0.25 pulse/m^2, and a density of 4.18 points/m^2. The data include 934.843 points, the format of data is '.las' version 1.2. Each point in the point cloud has the properties shown in Figure 3. However, the author uses the coordinates and the height of the point as the input of the classification problem and displays the 3D point cloud. In order to easily read and understand data from .las files, the author has saved it in .txt format. In .txt file the author only displays the x, y, and z values of the point The values of the points are shown in Figure 4. LiDAR point cloud 3D is showed in figure 2. Each point in the point cloud is represented by 3 values of x, y and z, where, the value of height z will be considered as the input value of the classification problem.



```
{
    x: 157766290,
    y: 521920163,
    z: 2919,
    intensity: 11,
    returnNumber: 1,
    numOfRetutns: 8,
    scanDirectionFlag: 64,
    edgeOfFlightLine: 256,
    scanAgeRank: 15,
    userData: 0,
    pointSourceID: 26,
    gpsTime: 26513950.38592064
}
```

Figure 2. Attribute of each point



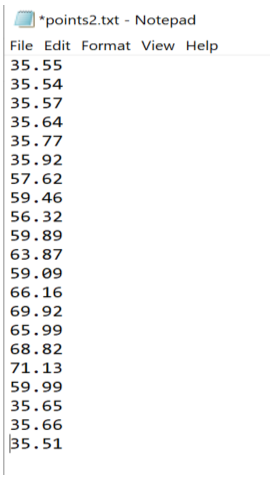
```
421007.70 4467227.59 35.55
421011.20 4467227.48 35.54
421014.68 4467227.38 35.57
421018.16 4467227.27 35.64
421021.62 4467227.17 35.77
421025.09 4467227.07 35.92
421054.77 4467227.42 57.62
421057.80 4467227.43 59.46
421062.09 4467227.11 56.32
421064.68 4467227.24 59.89
421067.16 4467227.40 63.87
421068.38 4467227.08 59.09
421070.05 4467227.45 66.16
421072.56 4467227.59 69.92
421077.07 4467227.22 65.99
421074.48 4467227.05 68.82
421070.42 4467227.31 71.13
421055.99 4467227.09 59.99
421009.88 4467227.05 35.65
421006.41 4467227.15 35.66
421002.98 4467227.25 35.51
```

Figure 3. X, Y, Z value of points

2.3.2. Results and Discussion

The proposed method tested with the above data set starts with the extraction of point height data. The accuracy of the algorithm will be assessed on accuracy, recall and F1 measurement.

First step, extract the elevation of the point in the LiDAR point cloud to reduce the dimension of data D = 1. The extracted results are shown in Figure 4.



```
35.55
35.54
35.57
35.64
35.77
35.92
57.62
59.46
56.32
59.89
63.87
59.09
66.16
69.92
65.99
68.82
71.13
59.99
35.65
35.66
35.51
```

Figure 4. Extraction of point elevation

Display points in height and divide the point cloud vertically with 9 parts (because the number of points is not too large, but the height distribution of points is uneven, there is a large elevation difference, so the selection of part numbers need to divide by 9 to avoid uneven height distribution between the parts to help the algorithm on each part converge quickly, then the point cloud is represented as follows:

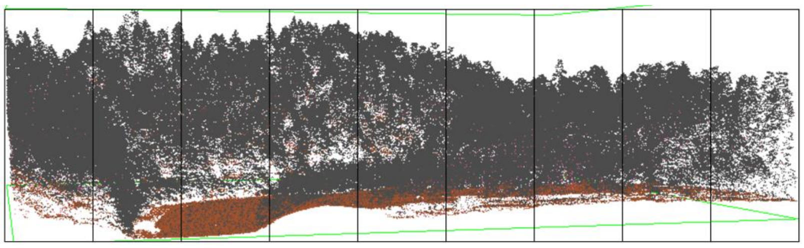
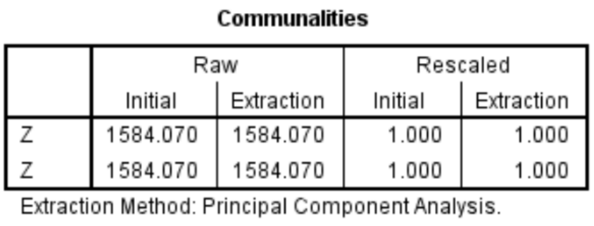


Figure 5. The following point cloud is divided into 9 parts

The model parameters are initialized on the Z value as follows:

μ = 94.38,
Σ = 1584.07,
δ = 39.80,
β = 10-8,

To determine the number of components of the model, the author used principal component analysis to find k. The finished matrix is shown in Figure 5, 6 and 7. Using the EM algorithm to recalculate the model's parameters on each part, with 25 iterations, the algorithm converges with the parameters changed in figure 8.

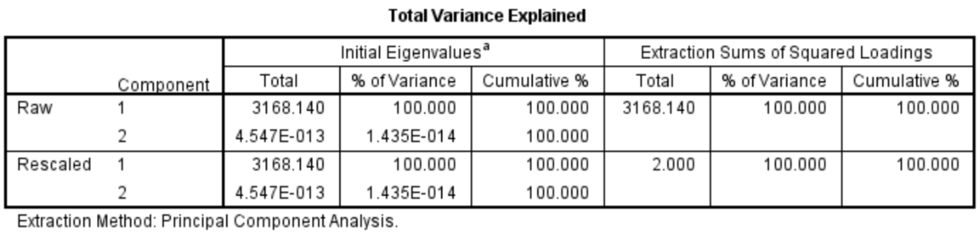


Communalities

	Raw		Rescaled	
	Initial	Extraction	Initial	Extraction
Z	1584.070	1584.070	1.000	1.000
Z	1584.070	1584.070	1.000	1.000

Extraction Method: Principal Component Analysis.

Figure 6. Correlation between data sets

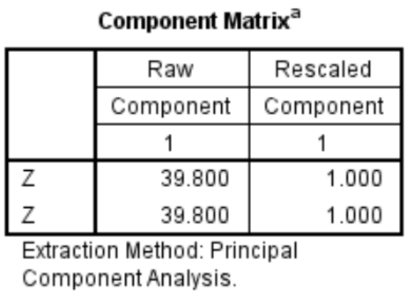


Total Variance Explained

	Component	Initial Eigenvalues[a]			Extraction Sums of Squared Loadings		
		Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %
Raw	1	3168.140	100.000	100.000	3168.140	100.000	100.000
	2	4.547E-013	1.435E-014	100.000			
Rescaled	1	3168.140	100.000	100.000	2.000	100.000	100.000
	2	4.547E-013	1.435E-014	100.000			

Extraction Method: Principal Component Analysis.

Figure 7. principal component analysis of dataset



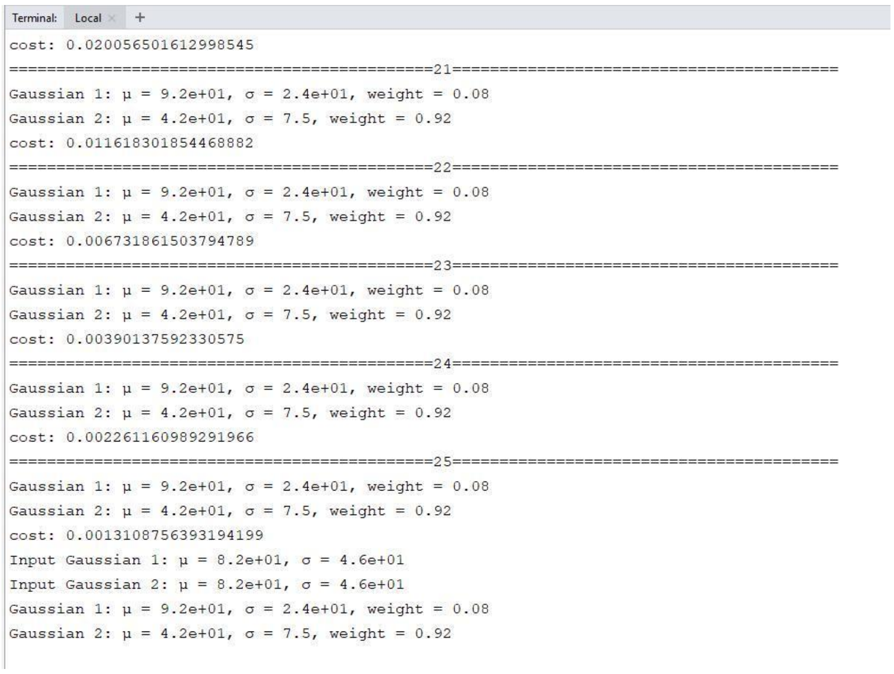
Component Matrix[a]

	Raw	Rescaled
	Component	Component
	1	1
Z	39.800	1.000
Z	39.800	1.000

Extraction Method: Principal Component Analysis.

Figure 8. Component matrix

We can see in the total covariance table that the cumulative value is greater than 50%, so we can choose the number of components for the model k = 2. The mixing coefficient is initialized equally with 2 components C1 = C2 = 0.5.



```
cost: 0.020056501612998545
============================================21============================================
Gaussian 1: μ = 9.2e+01, σ = 2.4e+01, weight = 0.08
Gaussian 2: μ = 4.2e+01, σ = 7.5, weight = 0.92
cost: 0.011618301854468882
============================================22============================================
Gaussian 1: μ = 9.2e+01, σ = 2.4e+01, weight = 0.08
Gaussian 2: μ = 4.2e+01, σ = 7.5, weight = 0.92
cost: 0.006731861503794789
============================================23============================================
Gaussian 1: μ = 9.2e+01, σ = 2.4e+01, weight = 0.08
Gaussian 2: μ = 4.2e+01, σ = 7.5, weight = 0.92
cost: 0.00390137592330575
============================================24============================================
Gaussian 1: μ = 9.2e+01, σ = 2.4e+01, weight = 0.08
Gaussian 2: μ = 4.2e+01, σ = 7.5, weight = 0.92
cost: 0.002261160989291966
============================================25============================================
Gaussian 1: μ = 9.2e+01, σ = 2.4e+01, weight = 0.08
Gaussian 2: μ = 4.2e+01, σ = 7.5, weight = 0.92
cost: 0.0013108756393194199
Input Gaussian 1: μ = 8.2e+01, σ = 4.6e+01
Input Gaussian 2: μ = 8.2e+01, σ = 4.6e+01
Gaussian 1: μ = 9.2e+01, σ = 2.4e+01, weight = 0.08
Gaussian 2: μ = 4.2e+01, σ = 7.5, weight = 0.92
```

Figure 9. Model parameters are updated through each iteration of steps E and M

Conducting classification with the condition of classification P (k | Zi) ≥ 0.5, conclude that Zi score belongs to class k (k = 2). We have a class 1 score of 768.970 points, a class 2 score of 165.873 points. We have the distribution of points into class 1 and 2 after the classification shown in Figure 10 and correlation between frequency of distribution and cumulative percentage of data is shown in table 1.

Table. 1. Correlation between frequency of distribution and cumulative percentage of data

Bin	*Frequency*	*Cumulative %*	*Bin*	*Frequency*	*Cumulative %*
1	0	0.00%	More	934842	100.00%
2	0	0.00%	1	0	100.00%
More	934843	100.00%	2	0	100.00%

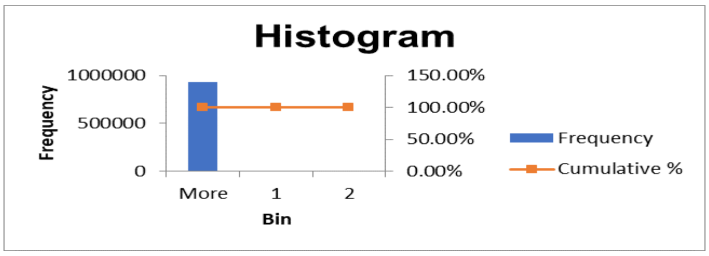


Figure 10. Histogram of point distribution after classification

To evaluate the accuracy of the proposed EM algorithm, the author compares the results with the results of basic EM. The results are shown in Table 2 with precision, recall and F1 measurements. The result is shown in table 2.

Table. 1 Compare result proposed EM with basic EM

	Precision	**Recall**	**F1**	**Running Time**	**Convergence**
Proposed EM	92.03%	92.06%	0.92045	102.3s	0.0013
Basic EM	91.80%	91.79%	0.91795	210.25s	0.00021

Through testing and comparing the evaluation of the proposed algorithm with the original EM algorithm with improved accuracy and faster running time by using scheduling parameters to help the algorithm converge better. However, due to the condition if $\beta > 1$ consider the condition for $\beta = 1$, the condition for the convergence algorithm is not as good as the original EM algorithm. With the proposed EM algorithm classification results, it meets the requirements of LiDAR point cloud classification, but more research is needed to make the algorithm's convergence conditions better and improved the accuracy.

3. CONCLUSIONS

EM algorithm is a popular algorithm in data mining and is used in many different problems. With the LiDAR point classification cloud problem, with the idea of classification based on a one-point posterior probability belonging to the classification class, EM algorithm proposed classification with more than 92% accuracy to meet the requirements of establishment. With the proposed EM algorithm based on the initialization parameter initialized with very small price and vertical point cloud subdivision to help the algorithm converge faster, while reducing the running time of the algorithm compared to Original EM algorithm. However, the algorithm needs further development when only classifying the cloud into 2 classes: first pulse and last pulse. While the number of unlabelled points is very large, this is a useful amount of information to study the nature and morphology of the object.

With the data set measured in Nghe An province, Vietnam, the algorithm gives satisfactory results and the running time with an acceptable convergence. However, more data sets need to be tested to verify the accuracy.

ACKNOWLEDGEMENTS

This research is funded by the Hanoi University of Mining and Geology under project number T19-03 and Ministry of Education and Training number CT.2019.01.07

REFERENCES

[1] Nallig Leal, Esmeide Leal, Sanchez Torres German A linear programming approach for 3D point cloud simplification.

[2] Zhenyang Hui, Dajung Li, Shuanggen Jin, Yao Yevenyo Ziggah, Leyang Wang (2019) Automatic DTM extraction from Airborne LiDAR based on expectation - maximization, Optics and Laser Technology, vol.112, pp. 43-55

[3] Kun Zhang, Weihong Bi, Xiaoming Zhang, Xinghu Fu, Kunpeng Zhu, Li Zhu (2015) A new kmeans clustering algorithm for point cloud, International Journal of Hybrid Information Technology, vol. vol. 8, no. 9, pp. 157-170.

[4] Keng FanLin, Chi Pei Wang, Pai Hui Su (2012) Object-based classification for LiDAR point cloud.

[5] Chao Luo, Gunho Sohn (2013) Line-based classification of terrestrial laser scanning data using conditional random field, International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol XL-7/W2, ISPRS2013.

[6] Xiao Liu, Congyin Han, Tiande Guo (2018) A robust point sets matching method [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1411/1411.0791.pdf. [Accessed 9 10 2019].

[7] Z.Hui, P.Cheng, Y.Y. Ziggah, Y.Nie (2018) A threshold-free filtering algorithm for airborne LiDAR point clouds based on Expectation Maximization, The International Archives of the Photogrammetry, RS and Spatial Information Sciences, vol. XLII-3.

[8] Suresh Lodha, David P.Helmbold, Darren M.Fitzpatrick (2007) Aerial LiDAR data classification using expectation – maximization, Research Gate.

[9] Yang HongLei, Peng JunHuan, Zhang DingXuan (2013) An Improved Em algorithm for remote sensing classification, Chinese Science Bullentin, vol. 58, no. 9, pp. 1060-1071.

[10] Xiao Liu, Congying han, Tiande Guo (2014), "arXiv”. [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1411/1411.0791.pdf. [Accessed 7 10 2019].

[11] Yu-chuan Chang, Ayman F.habib, Dong Cheon Lee, Jae Hong Yom (2008) Automatic classification of LiDAR data into Ground and non-Ground points, The International Archives of the Photogrammetry, RS and Spatial Information Sciences, vol. XXXVII, no. B4, pp.457-46.

[12] Borja Rodriguez - Cuenca, Silverio Garcia Cortes, Celestino Ordonez, Maria C.Alonso (2015) Automatic detection and classification of pole-like objects in urban point cloud data using an anomaly detection algorithm, Remote Sensing, vol. 7, pp.12680-12703.

[13] Serez Kutluk, Koray Kayabol, Aydin Akan (2016) Classification of Hyperspectral Images using Mixture of Probabilistics PCA models, European signal processing conference, pp. 1568-1572.

[14] Iftekhar Naim, Daniel Gildea (2012) Convergence of EM algorithm for GMM with unbalanced Mixing coefficients, International Conference on Machine Learning.

[15] Lawrence H. Cox, Marco Better (2009). Sampling from discrete distributions: Application to an editing problem, Research Gate.

[16] Naonori Ueda, Ryohei Nakano (1998) Deterministic Annealing Variant of the EM algorithm, [Online]. Available: https://papers.nips.cc/paper/941-deterministic-annealing-variant-of-the-em-algorithm.pdf. [Accessed 6 3 2020].

[17] S. Borman (2006) The expectation maximization algorithm a short tutorial.

AUTHOR

Born in 1985 in Ninh Binh province, Vietnam Country Graduated from the University of Mining and Geology University in 2008. Graduated Master of Science at the University of Natural Sciences - Vietnam National University, Hanoi in 2012. Author is currently a PhD student specialized in Information systems at the Institute of Information Technology, Vietnam Academy of Science and Technology. Currently working at: Faculty of Information Technology, University of Mining and Geology Research interests: Information System, Database, Data Mining, Geoinformatics.