UDC 528
DOI: 10.15587/1729-4061.2020.203866

DEVELOPMENT OF A DOCUMENT CLASSIFICATION METHOD BY USING GEODESIC DISTANCE TO CALCULATE SIMILARITY OF DOCUMENTS

Vo Trung Hung
Doctor of Computer Science
(Grenoble INP, France)
Professor, Vice-Rector
University of Technology and
Education University of Danang
48 Cao Thang, Danang, Vietnam, 55000
E-mail: vthung@ute.udn.vn

Currently, the Internet has given people the opportunity to access to human knowledge quickly and conveniently through various channels such as Web pages, social networks, digital libraries, portals. However, with the process of exchanging and updating information quickly, the volume of information stored (in the form of digital documents) is increasing rapidly. Therefore, we are facing challenges in representing, storing, sorting and classifying documents.

In this paper, we present a new approach to text classification. This approach is based on semi-supervised machine learning and Support Vector Machine (SVM). The new point of the study is that instead of calculating the distance between the vectors by Euclidean distance, we use geodesic distance. To do this, the text must first be expressed as an n-dimensional vector. In the n-dimensional vector space, each vector is represented by one point; use geodesic distance to calculate the distance from a point to nearby points and connect into a graph. The classification is based on calculating the shortest path between vertices on the graph through a kernel function. We conducted experiments on articles taken from Reuters on 5 different topics. To evaluate the proposed method, we tested the SVM method with the traditional calculation based on Euclidean distance and the method we proposed based on geodesic distance. The experiment was performed on the same data set of 5 topics: Business, Markets, World, Politics, and Technology. The results showed that the correct classification rate is better than the traditional SVM method based on Euclidean distance (average of 3.2 %)

Keywords: text classification, machine learning, geodesic distance, euclidian distance, SVM, NLP, kernel function

Received date 09.06.2020
Accepted date 16.07.2020
Published date 28.08.2020

1. Introduction

Currently, the Internet has given people the opportunity to access to human knowledge quickly and conveniently through various channels such as Web pages, social networks, digital libraries, portals. However, with the process of exchanging and updating information quickly, the volume of information stored (in the form of digital documents) is increasing rapidly. Therefore, we are facing challenges in representing, storing, sorting and classifying documents.

The classification is an important step to make the processing more efficient through the processing of a smaller group of documents (after classification) instead of having to deal with the entire block of documents. The classification is applied in many fields such as searching, automatic translation, automatic question and answer, data mining. If a manual classification process is conducted, we usually take a lot of time and costs for this. Therefore, implementing the automatic classification of digital documents is now an urgent issue.

There are many methods of text classification and most of them are based on machine learning techniques [1]. To classify a text based on machine learning, the volume and quality of the text are used to train the system to create a good classification model that is extremely important, deciding on the quality of the text classification system. However, building data warehouses for developing machine learning-based text classification applications is often quite expensive and less available, especially low user languages. Therefore, instead of using a Supervised Machine Learning method, people often use semi-supervised machine learning method so that they do not need a large amount of training data (labeled text) during classification. Non-Supervised Machine Learning method is rarely used because the classification quality is not high and the speed is low [2].

The essence of text classification is to try to assign classification labels to a new document based on the similarity of it to the text that has been labeled in the training. Many machine learning and data mining techniques have been applied to the text classification such as a decision method based on Na|ve Bayes, decision trees, closest neighbors, neural networks, etc. In recent research, the SVM is interested and used extensively in the field of text classification. The SVM method was born out of the statistical theory of Vapnik and Chervonenkis and has great potential for theoretical and practical applications [3].

However, the SVM method based on the semi-supervised machine learning model has some limitations. One of the issues that needs to be researched is how to calculate the distance between the vectors (the similarity between texts for grouping).

When calculating the distance between a point (vector representing a text) to other points correctly and quickly will help clustering, partitioning faster and more efficiently.

Therefore, research to improve algorithms and calculation methods to increase accuracy is scientifically necessary.

2. Literature review and problem statement

Text classification is the process of classifying information in text format by its content, i. e., by the messages that may be conveyed by the words contained within it. Automating this process is crucial in order to be able to classify a large amount of text-based information in a time-critical manner. Due to the vast quantity of textual information that needs to be processed, automated text classification finds widespread application in a variety of domains such as text retrieval, summarization, information extraction and question answering, among others. SVM is a method many researchers choose to use when classifying documents [4–6].

However, SVM has some limitations as follows:

– choosing a "good" kernel function is not easy to increase classification accuracy [4, 7]. Linear SVM is a parametric model, an RBF (Radial Basis Function) kernel SVM isn't, and the complexity of the latter grows with the size of the training set. Not only is it more expensive to train an RBF kernel SVM, but we also have to keep the kernel matrix around, and the projection into this "infinite" higher dimensional space where the data becomes linearly separable is more expensive as well during prediction. Furthermore, you have more hyperparameters to tune, so model selection is more expensive;

– long training time for large datasets [5, 8]. Obviously, very large datasets take longer time to train (days or weeks). Many times we need to train various models (SVM, Neural Network, etc.) to compare and find a better performance model. The problem is that we want results as quickly as possible but produce the best performance;

– SVM is initially meant to perform binary classification because of the way it creates the hyperplane to discriminate two classes. So, we need special strategies to make SVM work like an N-class classifier for classifying texts into N-number of classes [9, 10].

3. The aim and objectives of the study

The aim of this study is to develop a method for the classification of documents using a geodesic distance to calculate the comparison of documents. This will make it possible to increase the accuracy of multilayer classification.

To achieve this aim, the following objectives are accomplished:

– instead of calculating the distance between the vectors using the Euclidean distance, we use geographic distance. Geographic distance allows calculating the actual distance and thus increases the accuracy of the classifier;

– we study to convert texts into feature vectors linking data into a graph and connecting the data with the graph built at the vertex (vector) having the nearest Euclidean distance;

– based on the Isometric Feature Mapping (Isomap) method, we can easily find and classify these spaces using geodesic distance. To calculate the actual distance by using geodesic distance, a simple measure of the distance between two vertices in a graph is the shortest path between the vertices. Formally, the geodesic distance between two vertices is the length in terms of the number of edges of the shortest path between the vertices. For two vertices that are not connected in a graph, the geodesic distance is defined as infinite. We propose the kernel function of the vector support machine using geodesic distance combined with Gauss function. We easily classify into different classes based on Isomap and kernel function.

4. Related works

4. 1. Text classification

The problem of text/document classification can be stated as follows [11]: give a set of documents $D=\{d_1, d_2, ..., d_n\}$, d_i is the i^{th} document and class set $C= \{c_1, c_2, ..., c_m\}$, c_j is the j^{th} class. The purpose of the problem is to identify and assign the text d_i into the class c_j.

The objective is to find the function f:

$$f:D\times C\rightarrow\{\text{True, False}\}.$$

Inside:

– $f(d_i, c_j)$=True, if the document d_i belongs to class c_j;

– $f(d_i, c_j)$=False, if the document d_i does not belong to class c_j.

There are many cases of text classification such as binary classification (just need to determine a text belongs/does not belong to a given class), multi-layer classification (a text belonging to a certain class in the list of given classes), multivalued classification (a text can belong to more than one class in the list of given classes).

If there are 3 classes c_1, c_2, c_3 then we will have the combination (c_1, c_2), (c_1, c_3), (c_2, c_3). If there are n classes, there are $n^*(n-1)/2$ combinations.

Example:

There is text d through a combination (c_1, c_2)→c_2; d through a combination (c_1, c_3)→c_3; d through a combination (c_2, c_3)→c_2; → d belongs to class c_2 because the result is that c_2 which it appears most often.

4. 2. Support Vector Machine

To classify digital documents, many classification methods have been proposed. One of the common methods is classification based on the vector spatial model. From this space, probability models are built through machine learning for the purpose of automated classification. The Support Vector Machine (SVM) is one of the efficient automatic classification solutions that allows dividing the input texts/documents into different classes. SVM is mainly based on binary classification algorithms and is highly appreciated by researchers in the field of machine learning [12].

SVM is a family of methods based on kernel functions to minimize the estimated risk. The SVM method was born out of statistical theory and developed by Vapnik and Chervonenkis and has great potential for theoretical and practical applications [3]. Practical tests show that the SVM method has good classification ability for text classification problems

as well as in many other applications (such as handwriting recognition, human face detection in images, regression...).

The SVM algorithm divides two data layers by a (d–1) hyperplane when the number of directions of training data is d. Fig. 1 is an example of separating data belonging to two classes using SVM. Where $w.x - b=0$ is a hyperplane showing data decomposition [13].

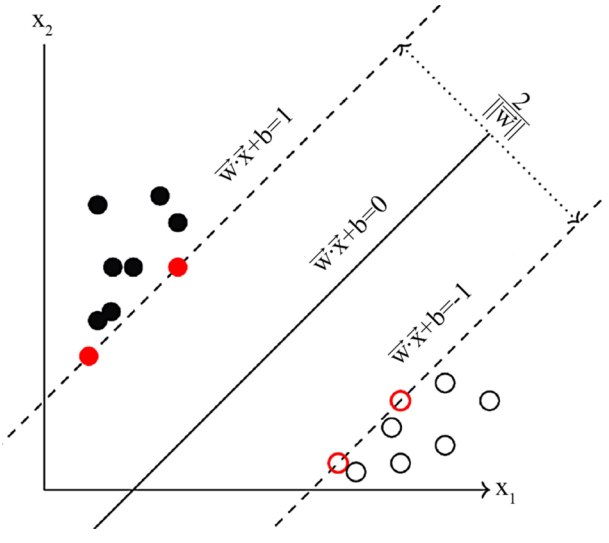

Fig. 1. Optimal hyperplane and boundary

Given the training set represented in a vector space, where each document is a point, the standard SVM method finds a decision hyperplane that can best divide the points on this space into two layers, separately corresponding to class (+) and class (–). The efficiency of determining this hyperplane is determined by the distance of the point closest to the plane of each layer. The larger the distance, the better the decision plane means the more accurate the classification and vice versa. The ultimate aim of the method is to find the maximum boundary distance.

The algorithm is described as follows:

– purpose: used to determine the class which new data belong to;

– input: give a training data set labeled as grades –1 or +1; set of data to classify;

– data output: labels for new data;

– algorithm description: we have a training set that is a set of n points of the form:

$$\mathcal{D}=\left\{\left(x_i,y_i\right)|x_i\in\mathbb{R}^p,\ y_i\in\left\{-1,1\right\}\right\}_{i=1}^{n}.$$

For y_i with the value +1 or –1, specify the class containing x_i point. Each x_i is a real vector with p dimensions. We need to find the hyperplane with the largest margin that separates the x_i points of the field of interest and is labeled y_i=1; and the x_i points of the field of non-interest and is labeled y_i=–1. Each hyperplane can be written as a set of points x satisfying the equation:

$$w.x-b=0,$$

with "." denoting the scalar product. The weight vector w, which is perpendicular to the hyperplane. The parameter b/w determines the deviation of the hyperplane from it to the vector w.

The above equation is equivalent to the following equation:

$$C+w_1.x_1+w_2.x_2+...+w_n.x_n=0.$$

Equivalent to the formula:

$$C+\Sigma w_i.x_i=0,\ i=1,...,\ n$$

with $w=w_1+w_2+...+w_n$ is a set of hyperplane coefficients or weight vectors, C is the offset, when changing w and C, the direction and distance from the origin to the hyperplane change.

We need to choose w and b to maximize the margin, or the distance between the two parallel hyperplanes so that they are as far as possible while still dividing the data. These hyperplanes are defined by:

$$w.x-b=1$$

and

$$w.x-b=-1.$$

Note that if training data can be split linearly (by a straight line), we can choose the two hyperplanes of the margin so that there is no point between them and try to maximize the distance between them. By geometry, we find the distance between the two hyperplanes $2/w$ and we want to minimize the value of $\|w\|$. To avoid data points falling into the margins, we add the following conditions, for each i we have:

$$w.x_i-b\geq 1 \text{ if } y_i=1 \text{ (first class)},$$

or

$$w.x_i-b\leq -1 \text{ if } y_i=-1 \text{ (second class)}.$$

It can be abbreviated as follows:

$$y_i(w.x_i-b)\geq 1,\ \forall 1\leq i\leq n.$$

We can put them together in an optimal problem:

Minimum according to w, b: $\|w\|$ with the condition for all i=1, ..., n.

$$y_i(w.x_i-b)\geq 1.$$

The optimization problems presented in the previous section are difficult to solve because it depends on $\|w\|$, the norm w, which includes a square root. Fortunately, it is possible to change the equation by replacing $\|w\|$ with 1/2 w^2(element 1/2 being used for convenience in math) without changing the solution (minimum of the original and modified equation with w and b). This is the problem of optimizing a quadratic equation. More clearly:

Minimum (in w, b): $\frac{1}{2}w^2$

$$y_i(w.x_i-b)\geq 1,\ (\forall\ i=1,...,n).$$

Returns the Lagrange equation with a multiplier α_i [7]:

$$\min_{w,b,}\left\{\frac{1}{2}w^2-\sum_{i=1}^{n}{}_i\left[y_i\left(w.x_i-b\right)-1\right]\right\}.$$

The SVM is the process of finding hyperplanes depending on the weight vector parameter w and offset C. The goal of the SVM method is to estimate w and C to maximize the margins between the positive and negative data layers. The different values of the margins give us different families of hyperplanes and the bigger the margin, the less the power of machine learning. Thus, maximizing margins is essentially finding a learning machine with the least capacity. The classification process is optimal when the classification error is minimal. After finding the hyperplane equation using the SVM algorithm, apply this formula to find the class label for new data.

If the learning data is not linear, add the variable η_i and replace the above equation with the equation:

$$\min_{W,b,} C = \sum_{i=1}^{n} \eta_i + \frac{1}{2} W^2 ,$$

with

$$y_i \left[W.x_i - b \right] + \eta_i \geq 1, \quad \eta_i \geq 0, \quad i = 1,\ldots,n.$$

From there we have the general equation of the hyperplane found by the SVM algorithm:

$$f(x_1, x_2,..., x_n) = C + \Sigma w_i.x_i,$$

$$\forall i = 1,...,n,$$

where n is the number of training data.

5. Proposed solution

5. 1. Geodesic distance model

In order to build a classification model, labeling files is essential. The more files that are labeled, the more accurate the classification. However, this labeling job is very costly and time consuming for experts in each field to perform. So, the problem is how to reduce the cost of labeling and still improve the classification efficiency. Therefore, in this study, the method of text classification based on geodesic distance is proposed. The geodesic distance model is a model that uses a measurement distance based on given real data, using the shortest correlation system (in text classification is the degree of proximity between texts/documents) to calculate the distance between two vectors [14]. This distance is called a geodesic distance and is different from the Euclidean distance. Building a reasonable geodesic model, that is, building a correlation between logical texts, the automatic text classification will be easier. Previous classification models often used Euclidean distances to calculate data distances. Therefore, on bended spaces, it is difficult to use a good classification model for using Euclidean distances. Fig. 2 below distinguishes the Euclidean distance and the geodesic distance of the x and y points in 3-dimensional space [15].

The Euclidean distance is the distance measured by the distance between the two points x and y calculated by the flying bird line.

The proposed model is shown in Fig. 3.

In 3-dimensional space, the vector data layers are independent of each other and these data spaces are separated by a distance. Previous classification models often used the Euclidean distance to calculate the data distance between two points x and y in a straight line, which would be inaccurate. Therefore, on bending spaces (a), the use of Euclidean distance calculation will be difficult to build a good classification model. This leads to unsuccessful classification results. In order to overcome the above model, in this study, we propose the model for calculating distance by geodesic distance, we can convert that text into a feature vector linking data into a graph (b) and connecting the data with the graph built at the vertex (vector) having the nearest Euclidean distance. Based on the Isometric Feature Mapping (Isomap) method, we can easily find (c) and classify these spaces using geodesic distance.

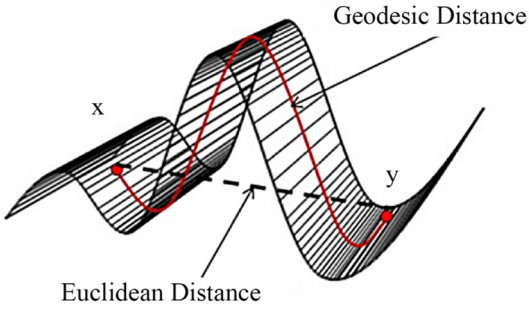

Fig. 2. Euclidean distance and geodesic distance

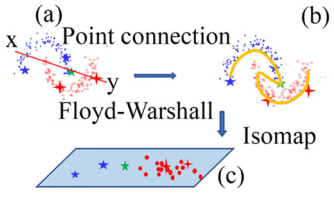

Fig. 3. Proposed model

Meanwhile, the geodesic distance is the actual distance along the curved terrain between x and y. When comparing these two distances, we often think of calculating the length of the path when crossing a mountain in reality. We cannot calculate the cost of the path by calculating the distance the crow flies because in fact we have to follow the ridge, climb to the top of the mountain and then descend by following the ridge on the other side to reach the destination. Therefore, measuring the actual distance can accurately calculate the cost of a distance. The data in text classification are the same, text is often distributed over curved spaces. Therefore, the use of geodesic distance will be able to increase the recognition efficiency more.

5. 2. Floyd-Warshall algorithm

There are many graphing algorithms like Depth-First Search (DFS), Breadth-First Search (BFS), or Dijkstra. The DFS algorithm is used to solve problems we want to have a solution (not necessarily the shortest distance), or we want to visit all vertices of the graph. The BFS algorithm also treats the vertices of the graph, but one important property is: if all edges are unweighted, the first time a vertex is visited, we have the shortest path to that vertex. The Dijkstra's algorithm is famous for finding the shortest path from one given vertex to the other vertices, in a graph with non-negative weight edges [16, 17].

In this study, to find the shortest path, we propose to use the Floyd-Warshall algorithm [18].

If the Dijkstra's algorithm solves the problem of finding the shortest path from a given vertex to every other vertex in the graph, the Floyd-Warshall algorithm will find the lengths (summed weights) of the shortest paths between all pairs of vertices by a single execution of the algorithm. Another feature is that the Floyd-Warshall algorithm can run on graphs with negative weighted edges, which is not limited as the Dijkstra's algorithm. However, note that in the graph there should be no round with the total number of edges is negative, if there is such a round we can not find the shortest path (each time going through this round the distance length is reduced, so we can go infinite times). The Floyd-Warshall algorithm compares all possible paths between each pair of vertices. It is a form of dynamic programming.

We can easily use the Floyd-Warshall algorithm to find geodesic distances for all vector pairs. This algorithm allows us to find the shortest path between any pair of vertices. If vertex k is on the shortest path from vertex i to vertex j, then the segment from i to k and from k to j must be the shortest path from i to k and from k to j respectively. Hence, we use matrix D to save the shortest path length between all pairs of vertices.

Floyd-Warshall Procedure
Input:
N: a set of documents (n vertices)
Output:
D: the matrix contains the shortest distance between any pair of vertices in the graph
Begin // Building graphs
for i from 1 to n do
for j from 1 to n do
D[i,j]= $x_k - x_l^2$
if (D[i,j]> ε) then D[i,j]= ∞
endif
endfor
endfor
// Find the shortest path between any pair of vertices
for k from 1 to n do
for i from 1 to n do
for j from 1 to n do
D[i,j]= min(D[i,j], D[i,k]+D[k,j])
endfor
endfor
endfor
end

5. 3. Clustering using geodesic distance

A clustering technique when the structure of each layer is relatively complex. Manifold Clusterings based on Geodesic Distance [19–21] is a technique for classifying unlabeled data on non-linearly bent spaces. In case the data of clusters are on each space and these spaces are separated by a distance, based on isometric feature mapping (Isomap), we can easily find and classify the spaces by using geodesic distance. The following is a summary of the clustering method using geodesic distance.

The main purpose of this method is to use geodesic distance, so initially we calculate the geodesic distance D_{kl} of all pairs of vertices corresponding to the data pairs in the training set k and l. This geodesic distance is calculated by the minimum distance between pairs of vertices on the neighborhood graph of the training set. This neighborhood graph is constructed by connecting pairs of vertices k and l having a smaller distance ε (ε-Isomap) or k is one of the K closest neighbors (K-Isomap) [22].

Geodesic distance D_{kl} is formulated as follows:

$$D_{kl} = \min\left\{d_{G;kl}, d_{G;ki} + d_{G;il}\right\}$$

with $d_{G;kl} = x_k - x_l^2$ if vertex k and l have a connection in the neighboring graph, otherwise we assign $d_{G;kl} = \infty$.

Next, the clustering method uses the optimal geodesic distance problem:

$$\min_{u,c} \sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij} d(i,j),$$

where

$$\sum_{j=1}^{C} u_{ij} = 1, \quad u_{ij} \in \{0,1\}.$$

The distance of each data to vector $\mathbf{c}_j$ is calculated by:

$$d(i,j) = \varphi(\boldsymbol{x}_i) - \mathbf{c}_j^2 =$$
$$= K_{ii} - \frac{2}{\sum_l^N u_{lj}} \sum_{l=1}^{N} u_{lj} K_{il} + \frac{2}{\left(\sum_l^N u_{lj}\right)^2} \sum_{k=1}^{N}\sum_{l=1}^{N} u_{ki} u_{lj} K_{kl}.$$

In this formula, $K_{kl} = \varphi(x_k)^* \varphi(x_l)$ is the kernel function of the classifier, which is the scalar product of two vectors mapped onto a new space for easier classification. This space has been proposed in previous studies [11–13]. Depending on the selection of the kernel function for the text classifier, different clustering units can be built.

For example, polynomial kernel $k(x_i, x_j) = x_i^T (x_j + 1)^p$ leading to polynomial clustering, Gaussian kernel

$$k(x_i, x_j) = \exp\left(-\gamma \| x_i - x_j \|^2\right),$$

leads to RBF clusters (Radial Basis Functions) [18, 23], and sigmoid kernel

$$k(x_i, x_j) = \tanh\left(\beta x_i x_j + c\right)$$

leads to sigmoid neuron network two layers.

5. 4. Calculation of geodesic distance

Before constructing a geodesic distance model for the text, each text needs to go through word segmentation and feature extraction to be represented by a vector. This vector represents the properties of the text that we need to classify and call as the feature vector.

In previous studies, there were many methods of word segmentation and feature extraction. However, in this research, we don't focus on the research and development of word segmentation or characterization methods but only as an available method. This study uses the Uni-gram statistical method for word segmentation, and the vector spatial model for characterization. Specifically, when giving a training set the documents are stored as ".doc" or ".pdf" files, we convert these documents into ".txt" file format. Then separate the words and organize them into a list. Words with a low frequency of occurrence indicate that they are uncommon and likely to be words that may cause interference in the classification model. Therefore, we remove the words with low frequency in the final list. From this list, we construct

a vector model that corresponds to a word from the list that will have an element in the vector. This study uses two main methods to construct vectors for the text, the frequency usage method and the TF-IDF method.

The problem here is how to calculate the geodesic distance in a given problem, in which we know only data of feature vectors, without knowing exactly how the space was bent. This geodesic distance is calculated by the minimum distance between pairs of vertices on the neighboring graph of the training set.

From the training set consisting of the feature vectors extracted at the word segmentation and feature extraction steps, we construct a graph connecting the vectors corresponding to each near point on the graph together. The two points are connected to each other to show the similarity in the frequency of words appearing in the two documents. That is, if two texts with vectors are connected to each other on the graph, the likelihood of having the same subject matter is high. Then, use the built graph to find all geodesic distances between any two vertices using the Warshll-Floyd algorithm and name D_{kl}. Where k, l are the ordering numbers of the text in the training set.

For documents that need to be classified but not included in the training set, we can convert that text into a feature vector and link it to the built graph at the vertex (vector) of the nearest Euclidean distance. We then calculate the geodesic distance of the text x to x_k in the training set as follows:

$$D_k(x) = D_{kl} + x - x_l^2,$$

where l is the ordering number of the text in the training set closest to the text to be classified:

$$l = \arg\min_u \left(x - x_u^2\right).$$

Based on the geodesic distance between documents, we will decide the classification into classes.

5. 5. Kernel function in SVM using geodesic distances

The SVM was originally a linear classification algorithm based on the application of kernel functions, the algorithm can find hyperplanes in variable nonlinear spaces.

Extending the scalar product via a mapping function for variables in a larger space H and possibly even infinite dimensions, whereby the equation remains true. In each equation, when we have a scalar product, we also calculate the scalar product through transformations of vectors and it is called the kernel function. The kernel function is used to define multiple non-linear input relations [23].

For the linear kernel function, we can define many quadratic or exponential functions. In recent years, a number of studies have gone into different kernel functions for SVM classification and for many other experimental statistics.

In this study, we propose the kernel function of the vector support machine using geodesic distance combined with the Gauss function as follows:

$$k(x_k, x_l) = \exp(-\gamma D_{kl}),$$

$$k(x_k, x_l) = \exp(-\gamma D_k(x)).$$

in which, $k(x_k, x_l)$ is the scalar product of 2 vectors of the text k and the text l on the classification space of the support vector machine $k(x_k, x_l)$. $k(x_k, x)$ is the scalar product of the vector containing any text and the vector of the text k on the classified space of SVM.

To calculate in SVM, the kernel matrix satisfies the condition that all private values must be non-negative. By empirical research, we have proved that the kernel function in the proposed model satisfies that condition.

5. 6. Text classification based on geodesic distance model

This method proposes using geodesic distance to calculate the kernel function in the kernel matrix of the support vector machine. This method uses all labeled data and unlabeled data of training to calculate geodesic distance, so the proposed method is one of semi-supervised learning methods.

This solution is mainly based on the support vector model. The difference of the proposed method is to use the geodesic distance to calculate the kernel function for the support vector machine instead of using the Euclidean distance.

The proposed model is as follows (Fig. 4):

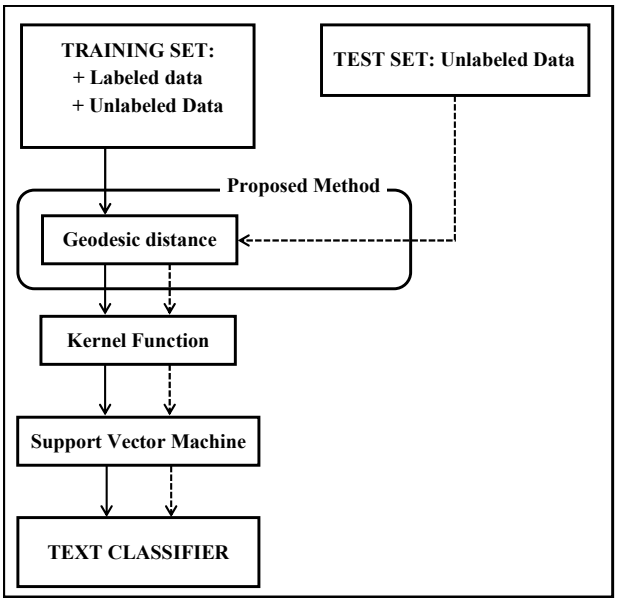

Fig. 4. Proposed model classifies text based on geodesic distance

In this model, there are 3 main modules:

– the first module: calculating geodesic distance for data including training data and test data. This section includes methods for extracting text features;

– the second module: using geodesic distance to calculate the kernel function in the kernel matrix of the support vector machine;

– the third module: using support vector machines to classify (assign labels) to digital texts.

6. Experiment

To experiment the text classification system, we used sources extracted from Reuters articles at the address https://www.reuters.com (Fig. 5).

Fig. 5. Topics on Reuters

The articles included 5 topics: Business, Markets, World, Politics, and Technology.

For each topic, we took 150 articles, after preprocessing, the textual content of each article was saved as a .TXT file of about 3–5 KB. The total size of text files used for the experiment is nearly 4.2 MB.

We experiment on 2 classification software: SVM classification using Euclidean distance and SVM classification using geodesic distance.

On each software, we conducted 5 tests. Each time, we randomly selected 50 documents (of each topic) as a training set and 100 of the remaining ones to test the classification (automatic labeling).

Classification results between tests were relatively similar, the difference between classifications was only 1–3 % (Table 1). The average test results of classification are as follows.

Table 1

Experimented results on SVM with Euclidean and geodesic distance

N°	Topics	Correct ratio	
		Euclidean	Geodesic
1	Business	89 %	92 %
2	Markets	88 %	92 %
3	World	90 %	94 %
4	Politics	91 %	93 %
5	Technology	92 %	95 %
Average ratio		90 %	93.2 %

The experiment was performed on the same data set of 5 topics: Business, Markets, World, Politics, and Technology. The results show that the classification based on geodesic distance gives higher results (average of 3.2 %).

7. Discussion of the research results of document classification by using geodesic distance

The SVM method using Euclidean distance is based on the determination of a hyperplane to separate points into two groups with a minimum distance according to a given threshold, so it mainly serves binary classification. If you want to classify multi-layer, you have to perform multiple binary classifications to divide the expenditure into different classes. Our method uses geodesic distance to determine the distance from a point to the nearest neighboring point. From there, based on graph theory to connect points into a series of consecutive points based on geographical distance. When forming a series of points, it is easy to divide them into layers by selecting the points that divide the series into different segments.

Our method shows that multi-layering is faster and more efficient than the old one using Euclidean distance.

8. Conclusions

1. Text classification is an important step in natural language processing and is applied in many different fields such as automatic translation, system/software multilingualization, automated question and answer, etc. Learning is constantly finding solutions to improve the quality of the text classification system.

2. Our research is based on semi-supervised machine learning and SVM models. Our outstanding contribution is to replace the Euclidean distance calculation method with geodesic distance calculation. To implement this method, we must first extract textual characteristics (training and classification) based on geographical distance. Next, the kernel function must be determined through an SVM-specific matrix.

3. Test results show that the classification according to this method has about 3.2 % higher accuracy than the old methods using Euclidean distance.

Acknowledgements

This research is funded by Funds for Science and Technology Development of the University of Danang under project number B2019-DN06-18.

References

1. Hartmann, J., Huppertz, J., Schamp, C., Heitmann, M. (2019). Comparing automated text classification methods. International Journal of Research in Marketing, 36 (1), 20–38. doi: https://doi.org/10.1016/j.ijresmar.2018.09.009
2. Kadhim, A. I. (2019). Survey on supervised machine learning techniques for automatic text classification. Artificial Intelligence Review, 52 (1), 273–292. doi: https://doi.org/10.1007/s10462-018-09677-1
3. Vapnik, V. N. (2013). The nature of statistical learning theory. Springer. doi: https://doi.org/10.1007/978-1-4757-3264-1
4. Pratama, B. Y., Sarno, R. (2015). Personality classification based on Twitter text using Naive Bayes, KNN and SVM. 2015 International Conference on Data and Software Engineering (ICoDSE). doi: https://doi.org/10.1109/icodse.2015.7436992
5. Shah, F. P., Patel, V. (2016). A review on feature selection and feature extraction for text classification. 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET). doi: https://doi.org/10.1109/wispnet.2016.7566545
6. Chatterjee, S., George Jose, P., Datta, D. (2019). Text Classification Using SVM Enhanced by Multithreading and CUDA. International Journal of Modern Education and Computer Science, 11 (1), 11–23. doi: https://doi.org/10.5815/ijmecs.2019.01.02
7. Sarkar, A., Chatterjee, S., Das, W., Datta, D. (2015). Text Classification using Support Vector Machine. International Journal of Engineering Science Invention, 4 (11), 33–37.
8. Cervantes, J., García Lamont, F., López-Chau, A., Rodríguez Mazahua, L., Sergio Ruíz, J. (2015). Data selection based on decision tree for SVM classification on large data sets. Applied Soft Computing, 37, 787–798. doi: https://doi.org/10.1016/j.asoc.2015.08.048

9. Cheng, F., Yang, K., Zhang, L. (2015). A Structural SVM Based Approach for Binary Classification under Class Imbalance. Mathematical Problems in Engineering, 2015, 1–10. doi: https://doi.org/10.1155/2015/269856
10. Dai, H. (2018). Research on SVM improved algorithm for large data classification. 2018 IEEE 3rd International Conference on Big Data Analysis (ICBDA). doi: https://doi.org/10.1109/icbda.2018.8367673
11. M. Ikonomakis, S. Kotsiantis, V. Tampakas (2005), Text Classification Using Machine Learning Techniques. WSEAS TRANSACTIONS on COMPUTERS, 8 (4), 966–974. Available at: http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=5F958D32F2F130AD5E61A077BA5D823A?doi=10.1.1.95.9153&rep=rep1&type=pdf
12. Joachims, T. (2012). Learning to Classify Text Using Support Vector Machines. Springer. https://doi.org/10.1007/978-1-4615-0907-3
13. Kowsari, K., Meimandi, K. J., Heidarysafa, M., Mendu, S., Barnes, L., Brown, D. (2019). Text Classification Algorithms: A Survey. Information, 10 (4), 150. doi: https://doi.org/10.3390/info10040150
14. Heylen, R., Scheunders, P. (2012). Calculation of Geodesic Distances in Nonlinear Mixing Models: Application to the Generalized Bilinear Model. IEEE Geoscience and Remote Sensing Letters, 9 (4), 644–648. doi: https://doi.org/10.1109/lgrs.2011.2177241
15. González-Castro, V., Fernández-Robles, L., García-Ordás, M. T., Alegre, E., García-Olalla, O. (2012). Adaptive pattern spectrum image description using Euclidean and Geodesic distance without training for texture classification. IET Computer Vision, 6 (6), 581–589. doi: https://doi.org/10.1049/iet-cvi.2012.0098
16. Smys, S., Bestak, R., Rocha, Á. (Eds.) (2020). Inventive Computation Technologies. Lecture Notes in Networks and Systems. doi: https://doi.org/10.1007/978-3-030-33846-6
17. Even, S. (2011). Graph Algorithms. Cambridge University Press. doi: https://doi.org/10.1017/cbo9781139015165
18. Aggarwal, A., Chandra, A. K., Snir, M. (1990). Communication complexity of PRAMs. Theoretical Computer Science, 71 (1), 3–28. doi: https://doi.org/10.1016/0304-3975(90)90188-n
19. Fazakis, N., Karlos, S., Kotsiantis, S., Sgarbas, K. (2016). Self-Trained LMT for Semisupervised Learning. Computational Intelligence and Neuroscience, 2016, 1–13. doi: https://doi.org/10.1155/2016/3057481
20. Feil, B., Abonyi, J. (2007). Geodesic Distance Based Fuzzy Clustering. Soft Computing in Industrial Applications, 50–59. doi: https://doi.org/10.1007/978-3-540-70706-6_5
21. Souvenir, R., Pless, R. (2005). Manifold clustering. Tenth IEEE International Conference on Computer Vision (ICCV'05) Vol. 1. doi: https://doi.org/10.1109/iccv.2005.149
22. Wu, Y., Chan, K. L. (2004). An extended Isomap algorithm for learning multi-class manifold. Proceedings of 2004 International Conference on Machine Learning and Cybernetics, 6, 3429–3433. doi: https://doi.org/10.1109/icmlc.2004.1380379
23. Yong, Q., Jie, Y. (2004). Modified Kernel Functions by Geodesic Distance. EURASIP Journal on Advances in Signal Processing, 2004 (16). doi: https://doi.org/10.1155/s111086570440314x