TELKOMNIKA, Vol. 11, No. 10, October 2013, pp. 6173 ~ 6178
ISSN: 2302-4046

Weighted K-Nearest Neighbor Classification Algorithm Based on Genetic Algorithm

Xuesong Yan[1], Wei Li[1], Wei Chen[1], Wenjing Luo[1], Can Zhang[1], Qinghua Wu[2,3*], Hammin Liu[4]
[1]School of Computer Science, China University of Geosciences, Wuhan, China
[2]Hubei Provincial Key Laboratory of Intelligent Robot, Wuhan Institute of Technology, Wuhan, China
[3]School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan, China
[4]Wuhan Institute of Shipbuilding Technology, Wuhan, China
*Corresponding author, e-mail: wuqinghua@sina.com

Abstract

K-Nearest Neighbor (KNN) is one of the most popular algorithms for data classification. Many researchers have found that the KNN algorithm accomplishes very good performance in their experiments on different datasets. The traditional KNN text classification algorithm has limitations: calculation complexity, the performance is solely dependent on the training set, and so on. To overcome these limitations, an improved version of KNN is proposed in this paper, we use genetic algorithm combined with weighted KNN to improve its classification performance, and the experiment results shown that our proposed algorithm outperforms the KNN with greater accuracy.

Keywords*: data mining, k-nearest neighbor, weighted k-nearest neighbor, genetic algorithm*

1. Introduction

Data mining (DM) is a rapidly evolving art and science of discovering and exploiting new, useful, and profitable relationships in data that is awaking great interest in topics such as decision making, information management, medical diagnosis, performance prediction, and many other applications [1]. Classification is a well-recognized DM task and it has been studied extensively in the fields of statistics, pattern recognition, decision theory, machine learning literature, neural networks, etc. Classification operation usually uses supervised learning methods that induce a classification model from a database. The objective is to predict the class that an example belongs to.

Nearest neighbor (NN) search is one of the most popular learning and classification techniques introduced by Fix and Hodges [2], which has been proved to be a simple and powerful recognition algorithm. Cover and Hart [3] showed that the decision rule performs well considering that no explicit knowledge of the data is available. A simple generalization of this method is called K-NN rule, in which a new pattern is classified into the class with the most members present among the K nearest neighbors, can be used to obtain good estimates of the Bayes error and its probability of error asymptotically approaches the Bayes error [4].

Genetic Algorithms (GAs) are stochastic search methods, which have been inspired by the process of biological evolution [5]. Because of GAs' robustness and their uniform approach to large number of different classes of problems, they have been used in many applications. Data mining is also one of the important application fields of GAs. In data mining, a GA can be used either to optimize parameters for other kinds of data mining algorithms or to discover knowledge by itself. In this latter task the rules that a GA finds are usually more general because of its global search nature. In contrast, most of the other data mining methods are based on the rule induction paradigm, where the algorithm usually performs a kind of local search. The advantages of GAs become more obvious when the search space of a task is large.

Received March 1, 2013; Revised July 14, 2013; Accepted July 24, 2013

2. K-Nearest Neighbor Algorithm

In pattern recognition field, KNN is one of the most important non-parameter algorithms [6] and it is a supervised learning algorithm. The classification rules are generated by the training samples themselves without any additional data. The KNN classification algorithm predicts the test sample's category according to the K training samples which are the nearest neighbors to the test sample, and judge it to that category which has the largest category probability. The process of KNN algorithm to classify sample X is [7]:

1. Suppose there are j training categories $C_1, C_2, ..., C_j$ and the sum of the training samples is N after feature reduction, they become m-dimension feature vector;

2. Make sample X to be the same feature vector of the form $(X_1, X_2, ..., X_m)$, as all training samples;

3. Calculate the similarities between all training samples and X. Taking the i^{th} sample d_i $(d_{i1}, d_{i2}, ..., d_{im})$ as an example, the similarity $SIM(X, d_i)$ is as following: $SIM(X, d_i) = \frac{\sum_{j=1}^{m} X_j . d_{ij}}{\sqrt{(\sum_{j=1}^{m} X_j)^2} \cdot \sqrt{(\sum_{j=1}^{m} d_{ij})^2}}$;

4. Choose k samples which are larger from N similarities of $SIM(X, d_i)$, $(i = 1, 2, ..., N)$, and treat them as a KNN collection of X. Then, calculate the probability of X belong to each category respectively with the following formula: $P(X, C_j) = \sum_{d} SIM(X, d_i).y(d_i, C_j)$, where $y(d_i, C_j)$ is a category attribute function, which satisfied $y(d_i, C_j) = \begin{cases} 1, d_i \in C_j \\ 0, d_i \notin C_j \end{cases}$;

5. Judge sample X to be the category which has the largest $P(X, C_j)$.

The traditional KNN text classification has three limitations [8]:

1. High calculation complexity: To find out the k nearest neighbor samples, all the similarities between the training samples must be calculated. When the number of training samples is less, the KNN classifier is no longer optimal, but if the training set contains a huge number of samples, the KNN classifier needs more time to calculate the similarities. This problem can be solved in three ways: reducing the dimensions of the feature space; using smaller data sets; using improved algorithm which can accelerate to [9];

2. Dependency on the training set: The classifier is generated only with the training samples and it does not use any additional data. This makes the algorithm to depend on the training set excessively; it needs recalculation even if there is a small change on training set;

3. No weight difference between samples: All the training samples are treated equally; there is no difference between the samples with small number of data and huge number of data. So it doesn't match the actual phenomenon where the samples have uneven distribution commonly.

3. Weighted K-Nearest Neighbor Algorithm

Conceptually, each feature x, called an instance, is represented as a vector of length $|F|$, the size of the record: $< w_1(x_1), w_2(x_2), ..., w_F(x_F) >$, where $w_j(x_j)$ is the weight of the j^{th} term. That weight may be set according to different criteria, such as: frequency, TFIDF or a score assigned to the feature for its capability to divide the examples into some set of classes. The simplest feature weighting function is to assign the same value to each term that occurs in the instance (let us say 1 for instance), and 0 to those that do not, which amount to a non-weighted features approach.

In traditional KNN algorithm [10], used distance as a basis to weight the contribution of each k neighbor in the class assignment process and define the confidence of having data d belonging to class c as : $Confidence(c, d) = \frac{\sum_{k_i' \in K | (Class(k_i')=c)} Sim(k_i', d)}{\sum_{k_i \in K} sim(k_i, d)}$, where Sim is the value

returned by the similarity function used to compare the instance with its neighbors. That is, for each neighbor in the neighbor set K (of size k) belonging to a particular class c, then sum up their similarities to data d and divide by the summation of all similarities of the k neighbors with regard to d.

To compare data d with instance i, it can choose the $CosSim$ function which is particularly simple using the binary term weight approach: $CosSim(i,d) = \frac{C}{\sqrt{A*B}}$, where C is the number of terms that i and d share, A is the number of term in i and B the number of terms in d. The neighbor set K of d thus comprises the k instances that rank the highest according to that measure.

4. Weighted KNN Algorithm Based Genetic Algorithm

4.1. Algorithm

The most popular evolutionary model used in the current research is Genetic Algorithms, originally developed by John Holland [11]. The GA reproduction operators, such as recombination and mutation, are considered analogous to the biological process of mutation and crossover respectively in population genetics. The recombination operator is traditionally used as the primary search operator in GA while the mutation operator is considered to be a background operator, which is applied with a small probability.

Traditionally, GA uses a binary string representation of chromosomes with concentration on the notion of 'schemata'. A schema is a template that allows exploring the similarity among chromosomes. Genetic Algorithms model evolution as a search for structures or building blocks that perform well in a given environment. Therefore, the recombination and mutation operators focus on an individual's structure, not the structure's interpretation. The results of applying reproduction operation in GA generate solutions that share structural similarities with their parents but may have significantly different interpretations. However, many recent applications of GA have used other representation such as graphs, Lisp expressions, ordered list, and red-valued vectors.

Genetic algorithm pseudo-code:

```
t=0;
initialize P(t);
evaluate structures in P(t);
repeat
t= t+1;
select- reproduction C(t)from: P(t-1);
combine and mutate structures in C(t)forming C'(t);
evaluate structures in C'(t);
select-replace P(t) from C '(t) and P(t+ 1);
Until ( termination condition satisfied ).
```

The above code gives the basic algorithmic steps for GA. After the initial population of individuals is generated (usually randomly) and individuals' structures are evaluated, the loop is entered. Then a selection buffer C(t) is created to accommodate the selected copies from P(t-l), "select-reproduction". In the Holland original GA, individuals are selected probabilistically by assigning each individual a probability proportional to its structural fitness. Thus, better individuals are given more opportunity to produce offspring. Next the variation operators (mutation and crossover) are applied to the individuals in C(t) buffer producing offspring C(t). After evaluating the structural fitness of C(t), the selection method is applied to select replacement for P(t) from C'(t) and P(t-1).

In general, genetic algorithms are usually used to solve problems with little or no domain knowledge, NP-complete problems, and problems for which near optimum solution is sufficient [12, 13]. The GA methods can be applied only if there exist a reasonable time and space for evolution to take place.

String Representation [14]- Here the chromosomes are encoded with real numbers; the number of genes in each chromosome represents the features in the training set. Each gene will

have 5 digits for vector index and feature number of genes. For example, if in the data, each record has 5 features, a sample chromosome may look as follows:

00100 10010 00256 01875 00098

Once the initial population is generated now we are ready to apply genetic operators. With these neighbors, the distance between each sample in the testing set is calculated and the accuracy is stored as the fitness values of this chromosome.

Reproduction (selection) - The selection process selects chromosomes from the mating pool directed by the survival of the fittest concept of natural genetic systems. In the proportional selection strategy adopted in this article, a chromosome is assigned a number of copies, which is proportional to its fitness in the population, that go into the mating pool for further genetic operations. Roulette wheel selection is one common technique that implements the proportional selection strategy.

The genetic operator: crossover and mutation, we use the traditional method. We use the single point crossover and random mutation. After the genetic operators are applied, the local maximum fitness value is calculated and compared with global maximum. If the local maximum is greater than the global maximum then the global maximum is assigned with the local maximum, and the next iteration is continued with the new population. The cluster points will be repositioned corresponding to the chromosome having global maximum. Otherwise, the next iteration is continued with the same old population. This process is repeated for N number of iterations.

The algorithm process step is given as Figure 1.

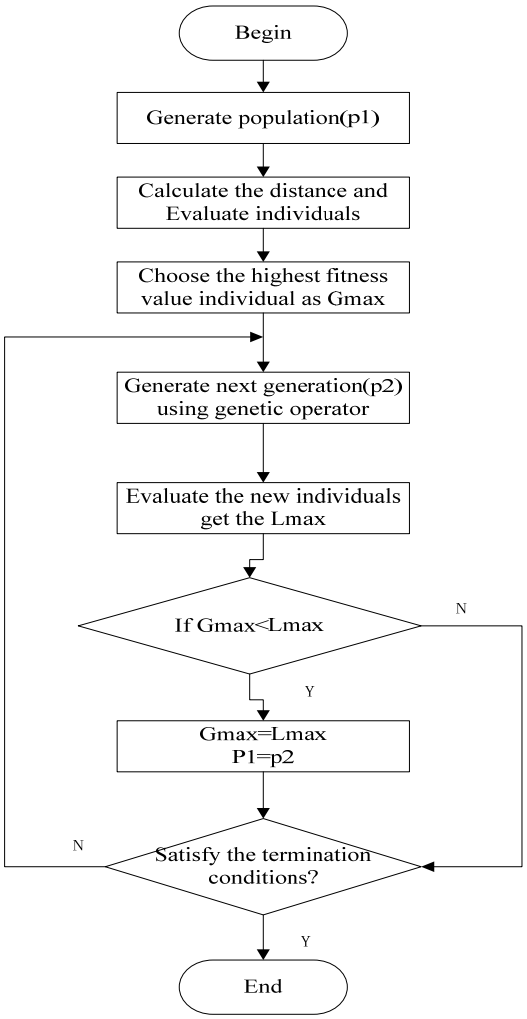

Figure 1. Algorithm framework

4.2. Experiment Results

The performance of the approaches discussed in this paper has been tested with 5 different datasets, downloaded from UCI machine learning data repository. All experiments are performed on Intel Core(TM)2 Duo CPU 2.26GHz/4G RAM Laptop. Each datasets run 10 times with the k=3, and has different value with population size. Table 1 shows the details about the datasets used in this paper.

Table 1. Experiment dataset

Dataset Name	Total No. of Instances	Total No. of Features
Balance	624	5
Breast	350	9
Diabetes	768	8
Glass	214	10
Waveform	5000	21

Table 2 depicts the performance accuracy of our proposed classifier compared with traditional KNN. From the results it is shown that our proposed method outperforms the traditional KNN method with higher accuracy.

Table 2. Experiment Results Comparison

Dataset	Genetic Algorithm			K-Nearest Neighbor Algorithm Accuracy Rate(%)
	Population Size	Generation Size	Accuracy Rate(%)	
Balance	10	200	88.63	84.47
	20	200	87.57	
	30	200	87.91	
Breast	10	200	98.83	96.94
	20	200	97.57	
	30	200	97.76	
Diabetes	10	200	72.46	69.70
	20	200	73.73	
	30	200	74.84	
Glass	10	200	67.48	64.57
	20	200	66.13	
	30	200	67.11	
Waveform	10	200	80.69	79.89
	20	200	81.58	
	30	200	81.37	

5. Conclusion

The KNN classification algorithm is one of the most popular neighborhood classifier in data classification. However, it has limitations such as: great calculation complexity, fully dependent on training set, and no weight difference between each class. To combat this, a novel method to improve the classification performance of KNN using Genetic Algorithm combine with weighted KNN algorithm is proposed in this paper. We use this new algorithm for data classification and the experiment results shown that our proposed algorithm outperforms the KNN with greater accuracy.

Acknowledgments

This paper is supported by Natural Science Foundation of China. (No.61272470 and No.61203307), National Civil Aerospace Pre-research Project of China, the Provincial Natural

Science Foundation of Hubei (No. 2012FFB04101) and the Fundamental Research Founds for National University, China University of Geosciences (Wuhan).

References

[1] J Han, M Kamber. Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers Academic Press. 2001.
[2] E Fix and J Hodges. *Discriminatory analysis. Nonparametric discrimination: Consistency properties*. Technical Report 4. USAF School of Aviation Medicine. Randolph Field, Texas. 1951.
[3] TM Cover and PE Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*. 1967; 13: 21-27.
[4] RO Duda and PE Hart. Pattern classification and scene analysis. New York: Wiley. 1973.
[5] Z Michalewicz. Genetic Algorithms+Data Structures=Evolution Programs. 3th Edition, Springer-Verlag. 1999.
[6] Belur V. Dasarathy. *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. Mc Graw-Hill, Computer Science Series. IEEE Computer Society Press. Las Alamitos, California, 1991: 217-224.
[7] Y Lihua, D Qi, and G Yanjun. Study on KNN Text Categorization Algorithm. *Micro Computer Information*, 2006; 21: 269-271.
[8] W Yu and W Zhengguo. *A fast KNN algorithm for text categorization*. Proceedings of the Sixth International Conference on Machine Learning and Cybernetics. Hong Kong. 2007: 3436-3441.
[9] W Yi, B Shi and W Zhang'ou. A Fast KNN Algorithm Applied to Web Text Categorization. *Journal of the China Society for Scientific and Technical Information*. 2007; 26(1): 60-64.
[10] Pascal Soucy, Guy W Mineau. *A Simple KNN Algorithm for Text Categorization*. Proceedings of International Conference on Date Mining. 2001: 647-648.
[11] J Holland. Adaptation in natural and artificial systems. University of Michigan press. 1975.
[12] Qinghua Wu, Hanmin Liu, Yuxin Sun, Fang Xie, Jin Zhang, Xuesong Yan. Research of Function Optimization Algorithm. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2012; 10 (4): 858-863.
[13] Xuesong Yan, Qinghua Wu, Can Zhang, Wei Li, Wei Chen, Wenjing Luo. An Improved Genetic Algorithm and Its Application. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2012; 10 (5): 1081-1086.
[14] N Suguna and K Thanushkodi. An Improved k-Nearest Neighbor Classification Using Genetic Algorithm. *International Journal of Computer Science Issues*. 2010; 7(4): 18-21.