A Fast KNN Algorithm Based on Simulated Annealing

Chuanyao Yang[1] Yuqin Li[1] Chenghong Zhang[2] Yunfa Hu[1]
1 Department of Computing & Information Technology, Fudan University
2 School of Management，Fudan University
No 220 Handan RD., Shanghai, P.R.China, 200433
Sunny.yangchy@gmail.com; li_yuqin@yahoo.com.cn;
chzhang@fudan.edu.cn; yfhu@fudan.edu.cn

Abstract--**K-Nearest Neighbor is used broadly in text classification, but it has one deficiency—computational efficiency. In this paper, we propose a heuristic search way to find out the k nearest neighbors quickly. Simulated annealing algorithm and inverted array are used to help find out the expected neighbors. Our experimental results demonstrate a significant improvement in classification computational efficiency in comparison with the conventional KNN.**

1. INTRODUCTION

K-Nearest Neighbor is one of the most popular algorithms for text categorization [1], especially in the area of case base reasoning. Many researchers have found that the KNN algorithm achieves very good performance in their experiments on different data sets [2] [3].

KNN was first proposed by Cover and Hart in 1968. The idea of k-Nearest Neighbor algorithm is simple and straightforward. To classify a new document, the system finds the k nearest neighbors among the training documents, and uses the categories of the k nearest neighbors to weight the category candidates [1]. One of the drawbacks of KNN algorithm is its efficiency. KNN is a lazy categorization, and instead of estimating the target function once for the entire instance, they delay processing until a new instance must be classified and it needs to compare a test instance or document with all samples in the training set. In addition, the performance of this algorithm greatly depends on two factors, that is, a suitable similarity function and an appropriate value for the parameter k.

Although K-nearest neighbor can be applied broadly, it has shortcoming mentioned above. Because of computational complexity, k-nearest neighbor is seldom used in the real-time scenario. In order to improve the efficiency of K-nearest neighbor, many methods have been proposed which can be divided into two categories.

One is reducing the number of training sets in circumstance of not losing the precision; Zhou proposes a fast KNN text classification approach based on pruning the training corpus [4];

The other is to adopt the fast algorithm with the proof of categorization function. A traditional way to do it is by representing the training set as a tree called kD-tree [5] which stores a set of points in k-dimensional space, k being the number of attributes. This is a binary tree that divides the input space with a hyper-plane and then splits each partition again, recursively. Then the computation is done within this kD-tree.

Genetic Algorithm (GA) is also used to help K-nearest neighbor do classification [6]. Anupam Kumar Nath [7] proposed an enhancement of kNNC where GA has been applied for effective selection and upgrade of attribute set to find out k-Nearest Neighbors.

In this paper, we adopt a heuristic search way—simulated annealing to choose out the k-nearest neighbors of test instances quickly, rather than calculating distance of all instances in training set. We do not prune the training corpus or upgrade the attribute set. We focus on the computation efficiency on the premise of precision.

II. K-NEAREST NEIGHBOR AND SIMULATED ANNEALING

A K-nearest neighbor learning

K-nearest neighbor algorithm assumes all instances correspond to points in the n-dimensional space R^n. The nearest neighbors of an instance are defined in terms of the

standard Euclidean or Cosine distance. More precisely, let an arbitrary instance x be described by the feature vector

$$(a_1(x), a_2(x), ...a_n(x))$$

where $a_r(x)$ denotes the value of the *r*th attribute of instance x. Then the distance between two instances x_i and x_j is defined to be $d(x_i, x_j)$, where

$$d(x_i, x_j) = \frac{\sum_{t=1}^{n} x_{it} \times x_{jt}}{\sqrt{\sum_{t=1}^{n} x_{it}^2 \sum_{t=1}^{n} x_{jt}^2}} \qquad (1)$$

Classification algorithm is defined as: Given a query instance x_q to be classified, let $x_1 \ldots x_k$ denotes the k instances from training set that are nearest to x_q.

$$f(x_q) \leftarrow \arg\max \sum_{i=1}^{k} y(x_j, c_k) \qquad (2)$$

where x_q is a test instance, x_j is one of the neighbors in the training set, $y(x_j, c_k) \in \{0,1\}$ indicates whether x_j belongs to class c_k. Equation (2) means that the predication will be the class that has the largest number of members in the k nearest neighbors.

The aim of k-nearest neighbors is to get the k highest values of all distances between the test instance and the other training instances. We can also define it more specifically and concretely.

$$Max_k d(x_i, x_j) = \frac{\sum_{t=1}^{n} x_{it} \times x_{jt}}{\sqrt{\sum_{t=1}^{n} x_{it}^2 \sum_{t=1}^{n} x_{jt}^2}} \qquad (3)$$

Of course we can change it into:

$$Min_k d(x_i, x_j) = -\frac{\sum_{t=1}^{n} x_{it} \times x_{jt}}{\sqrt{\sum_{t=1}^{n} x_{it}^2 \sum_{t=1}^{n} x_{jt}^2}} \qquad (4)$$

From the above formula（4）, we can see that in fact, this is a problem of combinatorial optimization and what we have to compute instantaneously is $\sum_{t=1}^{n} x_{it} \times x_{jt}$. Simulated annealing is a solution method in the field of combinatorial optimization based on analogy with the physical process of annealing, so we can borrow the idea of simulated annealing to improve the computational efficiency. Because the storage structure is greatly related with algorithm, we introduce it before we give the introduction of simulated annealing.

B storage structure

In this paper, application background is text classification, so instances are those of documents.

Inverted array is used to store the instances and compute the Cosine distance.

Documents are composed of characters or words, and the number of words is very large. Our intuitionistic thinking is to store a document in a vector. A good way to improve the efficiency is to change the vector into the matrix. But since the number of words is very large, the matrix is a sparse matrix and it will waste too much space. So the best way is to use the linked list and it can save space.

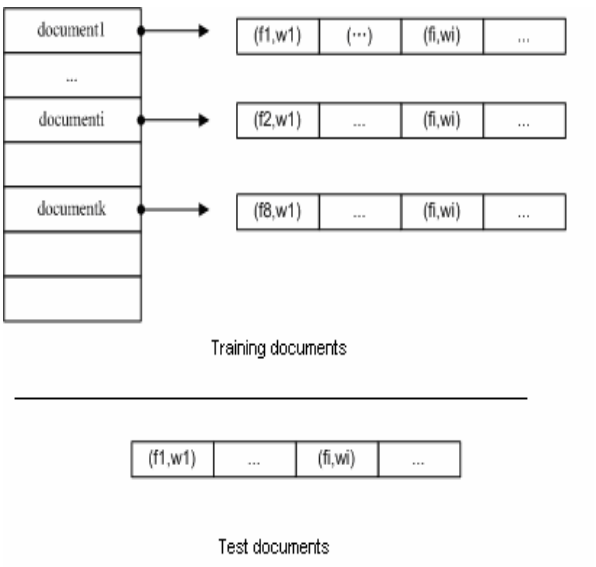

Figure 1 representation of the instances

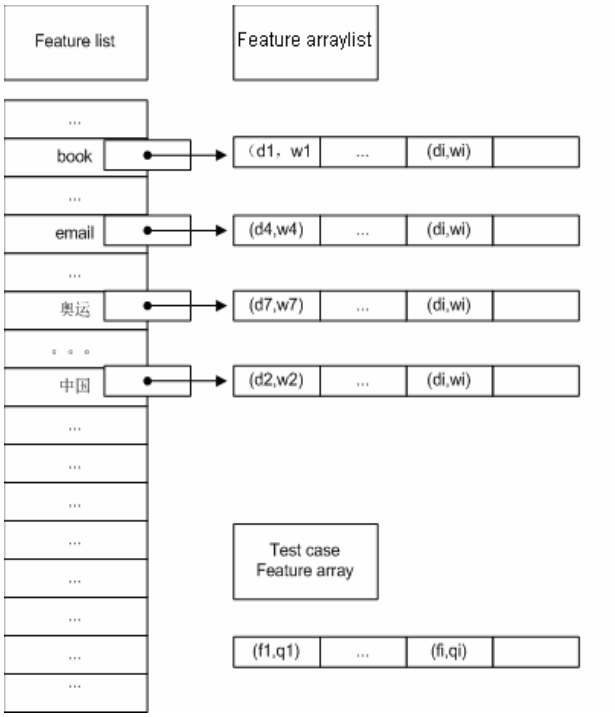

Figure 2 inverted array to represent instances

Data structure—Feature list stores the features of the instances including the English and Chinese characters and words. This list is indexed. In this array list there is a pointer which points to another data structure called feature arraylist which stores the details of feature. D_i is used to record the number of the document and w_n is used to record the weight of feature *n* in D_i.

C simulated annealing

Simulated annealing (SA) is a random-search technique to find a good solution to an optimization problem by trying random variations of the current solution. The concept is based on the manner in which liquids freeze or metals recrystalize in the process of annealing. The origins of the algorithm are in statistical mechanics (Metropolis algorithm) and it was first presented as a search algorithm for CO problems in [8] and [9]. The fundamental idea is to allow moves resulting in solutions of worse quality than the current solution (uphill moves) in order to escape from local minima. The probability of doing such a move is decreased during the search. SA forms the basis of an optimization technique for combinatorial and other problems. SA approaches the global maximization problem similarly to using a bouncing ball that can bounce over mountains from valley to valley. It begins at a high "temperature" which enables the ball to make very high bounces, which enables it to bounce over any mountain to access any valley, given enough bounces. As the temperature declines the ball cannot bounce so high and it can also settle to become trapped in relatively small ranges of valleys. A generating distribution generates possible valleys or states to be explored. An acceptance distribution is also defined, which depends on the difference between the function value of the present generated valley to be explored and the last saved lowest valley. The acceptance distribution decides probabilistically whether to stay in a new lower valley or to bounce out of it.

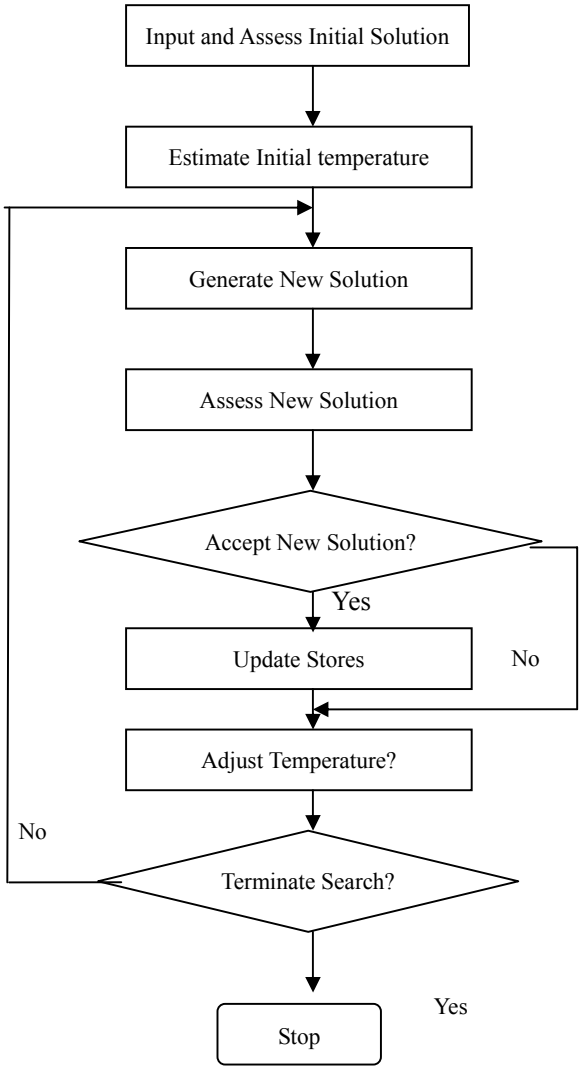

Figure 3 SA structure

(1) The method and structure of SA

SA's major advantage over other methods is an ability to avoid becoming trapped in local minima. The

implementation of the basic SA algorithm is straightforward. The above figure 3 shows its structure.

⑵ Four important elements of SA

The following elements must be provided:

1) A representation of possible solutions：to find the k nearest neighbors or a stable set of K nearest neighbors.

2) A generator of random changes in solutions: According to the sorted feature weight of test instance, to find the first *t* instances in the training set.

3) A means of evaluating the problem functions and: to find (or calculate) the stable state of candidate *k* nearest neighbors set.

4) An annealing schedule - an initial temperature and rules for lowering it as the search progresses. If unstable state is detected, continues to find the *t* nearest instances and calculate the result along the sorted feature weight of test instance.

⑶ Annealing schedule

Choosing an annealing schedule for practical purposes is something of an art. The standard implementation of the SA algorithm is one in which homogeneous Markov chains of finite length are generated at decreasing temperatures. The following parameters should therefore be specified:

1) An initial temperature T
2) A final temperature T or a stopping criterion
3) A length for the Markov chains and
4) A rule for decrementing the temperature.

Initial Temperature

As for KNN, the initial temperature is that： Firstly to find the heaviest feature of test instance, and then get the corresponding first K document labels from the inverted array, then compute the cosine distance of this test instance and all these K documents. This is the first initial temperature.

Final temperature or a stopping criterion is that when finding the next K or M document labels from the inverted array, compute the cosine distance, if these new cosine distances don't substitute the already existing ones in the candidate result set. Then we can conclude that this is the final temperature. In another word, the system has found the stable state.

Another important factor is the length for the Markov chain. Here we set length of Markov as K. the schedule is like that if in the testing state, system finds n constitute, then the energy to gain again is that f(n)*Markov. That means that the less stable the system is, the more test has to be done, which means that it has the opportunity to jump out of the trap.

⑷ The simulated annealing algorithm

1) Original optimization formulation

$$\text{Objective: } Max_k d(x_i, x_j) = \frac{\sum_{t=1}^{n} x_{it} \times x_{jt}}{\sqrt{\sum_{t=1}^{n} x_{it}^2 \sum_{t=1}^{n} x_{jt}^2}} \qquad (5)$$

2) Initializing:

(1) Figure out all the words or characters of the instances, store them in Feature list

(2) compute the feature' weight of each instance, store them in feature arraylist. D_i means the number of the instance which contains this feature, and w_i means the weight of this feature. And sort the feature weight and order them.

(3) Compute the feature' weight of test instance and store them in test instance feature array (see figure 1 and 2).

(4) Calculate $\sqrt{\sum_{t=1}^{n} x_i^2}$ of each training instance x, where *i* is denoted as each feature of the instance.

3) Computation:

1. Choose the feature with the highest feature weight from the test instance. Then choose out the first *k* instances directed by this feature from the inverted array. Compute the Cosine distance between the test instance and *k* instances and insert the computing result into the result candidate set.
2. Markov=k;
3. while (flag)
4. {
5. fetch the next highest feature *M* by its weight of the test instance;
6. fetch each first Markov document labels from the feature arraylist according to the feature *M*; and store them into the temp result set;
7. compute the cosine distance of testing instance and those documents which are not contained in candidate result set; store the computation result

in the temp result set

8. n=count the new cosine distances in temp result set which are larger than those in candidate result set.
9. substitute these n document labels and their cosine distances to the candidate result set; flag =true;
10. if n=0 then
11. flag=false;
12. else
13. Markov=(lg(n/k*10+0.1))*k;
14. }

III Experiments and Discussion

A Corpuses

The corpuses we used are the two most important Chinese Corpuses. One is from Natural Language Processing School of Peking University and the other is from Sohu, the most important Chinese web portal.

The first one contains 19,892 Chinese web pages which have been resolved by deleting the Html tag. There are 10 top-categories in the corpus. We evenly divided the corpus into three parts: one for training and the other two for test at random. In the training set, there are 10000 documents and in the testing set there are about 9892 documents.

The second one we used is downloaded from http://www.sogou.com/labs/dl/c.html. Sogou language corpus is now a most important open Chinese corpus for classification and clustering. What we downloaded is SogouC.reduced.20061127.zip about of 17910 documents of 48.2M.

In our experiments, a document is represented by a space vector, the dimensions of which correspond to Chinese words. We used the term weighting scheme named by information gain. The cosine function is used to compute the similarity (or distance) between two documents.

The experiment environment is: Windows 2003 OS, with Intel Pentium 1.4GHz and 256M memory. The application is coded under the environment of Visual studio 6.0.

B Measuring Performance

To evaluate the effectiveness of category assignments by classifiers to documents, the standard precision and recall are used here. Precision is defined to be the ratio of correct assignments by the system divided by the total number of the system's assignments. Recall is the ratio of correct assignments by the system divided by the total number of correct assignments.

C Results and Discussion

Table 1 and 2 gives the experimental results of two KNN algorithms with different k and Markov values. KNN-TR represents the traditional one, and KNN-SA denotes our modified version based on simulated annealing. KNN-TR achieved its best performance when k is around 10-25, while KNN-SA performed well with the largest Markov. On average, the performance of KNN-SA is at least ten times faster than that of KNN-TR algorithm.

From these two tables, we can also see that the different corpus has different classification result. Peking corpus has better performance than that of Sogou. The reason may lye in that Peking corpus has been elaborately chosen by experts than Sogou corpus.

Table 1 Result of Peking corpus (cpu second)

traditional KNN			KNN based on simulated annealing		
K	Precision (%)	CPU time	Markov	Precision (%)	CPU time
100	0.856	62	100	0.84	13
50	0.873	62	50	0.842	5.8
35	0.876	62	35	0.81	4.1
20	0.88	62	20	0.74	2.9
10	0.904	62	10	0.65	1.2
5	0.899	62	5	0.46	0.7

Table 2 Result of Sogou corpus

traditional KNN			KNN based on simulated annealing		
K	Precision	CPU time	Markov	precision	CPU time
100	0.67	143	100	0.68	16
50	0.70	143	50	0.69	6.5
35	0.677	143	35	0.72	5.6
25	0.7297	143	25	0.65	4.6
15	0.69	143	15	0.64	2.3
5	0.734	143	5	0.53	0.2

Figure 4 and 5 show that the different category of documents has different distinguishing function. IT, sports and politics has the higher ability, but more common categories has lower ability, such as culture. We have

distributed the different category evenly, so the number of instance is not the key factor influencing the distinguishing function. The reason may lie in that features chosen from IT, sports and military have the better distinguishing ability.

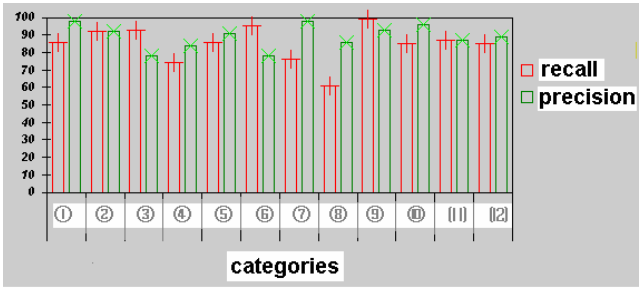

Figure 4 recall and precision of Peking corpus when k is 10.

(From left to right, categories are: computer, artist, economy, environment, education, politics, medicine, military, sports, and communication. And 11 is Micro Average; 12 is Macro Average)

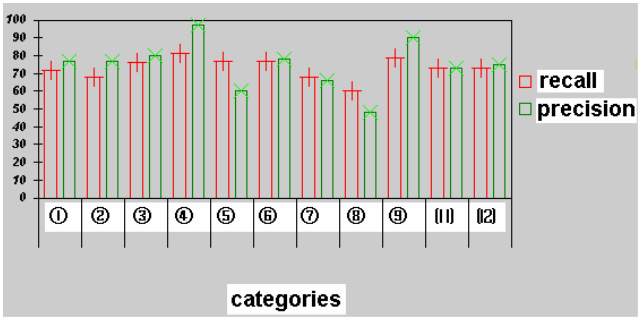

Figure 5 recall and precision of Sogou corpus when Markov is 35.

(From left to right, categories are: It, health, economy, sports, tourism, education, recruitment, culture, and military. And 11 is Micro Average; 12 is Macro Average)

IV Conclusion

In this paper we have proposed a heuristic k Nearest Neighbor classification method by using Simulated Annealing Algorithm. We also have implemented our newly proposed methods of KNN-SA together with the conventional one of KNN-TR on real life data set. Considering the percentage of correct prediction our proposed method has greatly outperformed the conventional one in computational efficiency.

But precision of both KNN-TR and KNN-SA is not high. From the different result of two separate corpuses, we know that features extraction is a key factor in classification. As for Chinese text classification, segmentation is another key factor.

Even though we have tested the method only on Chinese text, the method should be universally applicable to classification problems for data in other languages.

Acknowledgments: This paper is supported by the National Natural Science Foundation of China (No. 70471011) and the Shanghai Natural Science Foundation (No. 05ZR14019).

Reference

[1] Manning C.D. and Schutze H., 1999 Foundations of Statistical Natural Language Processing [M] Cambridge: Mit Press.

[2] Yang Y. and Liu X. 1999 A Re-examination of Text Categorization Methods [A]. In: Proceedings of 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval [C]. 42-49

[3] Li Baoli, Chen Yuzhou, and Yu Shiwen, 2002. A Comparative Study on Automatic Categorization Methods for Chinese Search Engine [A] In: Proceedings of the Eighth Joint International Computer Conference [C]. Hangzhou: Zhejiang University Press, 117-120.

[4] Zhou Shuigeng, Ling Tokwang, Guan Jihong. Fast text classification: a training corpus pruning based approach. Proceedings of Data-base Systems for Advanced Applications Kyoto, Japan: IEEE Computer Society. 2003:127-136

[5] Ian H. Witten, Eibe Frank. Data Mining: Practical Machine Learning Tools and Techniques (Second Edition). Elsevier 2005:129-135

[6] D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison Wesley, Boston, Massachusetts, 1989.

[7] Anupam Kumar Nath, Syed M. Rahman, Akram Salah. An Enhancement of k-Nearest Neighbor Classification Using Genetic Algorithm MICS 2005 held in North Dakota

[8] Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P., Optimization by Simulated Annealing, Science, Volume 220, Number 4598, 13 May 1983, pp. 671-680.

[9] Cerny, V., Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm, J. Opt. Theory Appl., 45, 1, 41-51, 1985 .