Using Clustering Ensemble in Classification Problems

Hosein Alizadeh
Iran University of Science and Technology,
Narmak, Tehran, Iran
P.O. Box 16765-163
Tel: +981213220161
ho_alizadeh@comp.iust.ac.ir

Saeid K. Amirgholipour
University of Isfahan,
Isfahan, Iran
Tel: +989111173182
s.kasmani@ui.ac.ir

Naghi R. Seyedaghaee
Islamic Azad University,
Branch of Mashad, Iran
Tel: +989111182478
sn_seyedaghaee@yahoo.com

Behrouz Minaei-Bidgoli
Iran University of Science and Technology,
Narmak, Tehran, Iran
Tel: +989125360306
b_minaei@iust.ac.ir

ABSTRACT
In this paper, a new classification method for enhancing the performance of K-Nearest Neighbor is proposed which uses clustering ensemble method. This new combinational method is called Nearest Cluster Algorithm, NCA. Inspiring the traditional K-NN algorithm, the main idea is classifying the test samples according to their neighbor tags. First, the train set is clustered into a large number of partitions, so that any cluster expectedly includes a small number of samples. Then, the labels of cluster centers are determined using applying the majority vote between the class labels of individual members in the cluster. After that, the class label of a new test sample is determined according to the class label of the nearest cluster center. Finally, the best classifier is selected according to the evaluation set. Computationally, the NCA is faster then KNN. The proposed method is evaluated on two different data sets. Empirical studies show the excellent improvement both in accuracy and time complexity in comparison with KNN method.

Keywords
Nearest Cluster Algorithm, K-Nearest Neighbor, Clustering Ensemble.

1. INTRODUCTION
One of the most fundamental and simple classification methods is *K*-Nearest Neighbor (KNN) classification and it should be one of the first choices for a classification study when there is little or no prior knowledge about the distribution of the data. This rule classifies *x* by assigning it the label most frequently represented among the *K* nearest samples; in other words, a decision is made by examining the labels on the *K*-nearest neighbors and taking a vote. KNN classification was developed from the need to perform discriminant analysis when reliable parametric estimates of probability densities are unknown or difficult to determine. In 1951, Fix and Hodges introduced a non-parametric method for pattern classification that has since become known the *K*-nearest neighbor rule [1]. Later in 1967, some of the formal properties of the *K*-nearest neighbor rule have been worked out; for instance it was shown that for *K*=1 and $n \to \infty$ the KNN classification error is bounded above by twice the Bayes error rate [2]. Once such formal properties of KNN classification were established, a long line of investigation ensued including new rejection approaches [3], refinements with respect to Bayes error rate [4], distance weighted approaches [5,6], soft computing [7] methods and fuzzy methods [8,9].

Some advantages of KNN are: Simple to use, Robust to noisy training data (especially if we use inverse square of weighted distance as the "distance") and Effective if the training data is large. Although KNN has this advantages, it has some disadvantages such as: a) Computation cost is quite high because it needs to compute distance of each query instance to all training samples; b) The large memory to implement in proportion with size of training set; c) Low accuracy rate in multidimensional data sets; d) Need to determine value of parameter *K* (number of nearest neighbors); e) Distance based learning is not clear which type of distance to use; and f) which attributes are better to use producing the best results. Shall we use all attributes or certain attributes only? [10].

The computational complexity of the nearest neighbor algorithm, both in space (storage of prototypes) and time (distance computation) has received a great deal of analysis. Suppose we have *N* labeled training samples in *d* dimensions, and seek to find the closest to a test point *x* (*K* = 1). In the most naive approaches we inspect each stored point in turn, calculate its Euclidean distance to x, retaining the identity only of the current closest one. Each distance calculation is O(*d*), and thus this search is O(dN^2)[11].

Many efforts have been already done to reduce the computational complexity of the KNN algorithm. Some of methods that do this are Parallelism, Partial Distance, Pre-structuring and Editing, Pruning or Condensing [11]. Although these methods improve the KNN algorithm, the computational problem exists yet. The performance of a KNN classifier is primarily determined by the choice of *K* as well as the distance metric applied [14]. However, it has been shown in [15] that when the points are not uniformly distributed, predetermining the value of *K* becomes difficult. Generally, larger values of *K* are more immune to the noise presented and make boundaries smoother between classes. As a result, choosing the same (optimal) *K* becomes almost impossible for different applications. Since it is well known that using prior knowledge such as the distribution of the data and feature selection, KNN classifiers can significantly improve their performance; researchers have attempted to propose new approaches to augmenting the performance of KNN method such as: Adaptive Metric NN (ADAMENN) [15], Discriminant Adaptive NN (DANN) [16], Weight Adjusted KNN (WAKNN) [17] and Large Margin NN (LMNN) [18].

Despite the success and rationale of these methods, most have several constraints in practice. Such as the effort to tune numerous parameters: DANN introduces two new parameters, K_M and £; ADAMENN has six input parameters in total which could potentially cause over fitting; the required knowledge in other research fields: LMNN applies semi definite programming for the optimization problem; the dependency on specific applications: WAKNN is designed specifically for text categorization; and so on. Additionally, in spite of all the aforementioned constraints, choosing the proper value of K is still a crucial task for most KNN extensions, making the problem further compounded.

Two novel and effective yet easy to implement extensions of KNN method are proposed in [14] whose performances are relatively insensitive to the change of parameters. Both of their methods are inspired by the idea of informativeness. Generally, a point (object) is treated to be informative if it is close to the query point and far away from the points with different class labels. Thanh et al. in [12] introduced KNN-kernel density based clustering for high dimensional multivariate data. Their method is based on the combination of nonparametric KNN and kernel density estimation to overcome difficulties clustering high dimensional multivariate data. Alippi and Roveri in [13] used a new technique to reduce computational complexity in KNN based Adaptive Classifiers. They showed that using adaptive classifiers can reduce the computational complexity and the memory requirements of KNN classifiers by including condensing editing techniques. Jin et al. in [19] proposed a novel technique, called NNH ("Nearest Neighbor Histograms"), which uses specific histogram structures to improve the performance of NN search algorithms. A primary feature of their proposal is that such histogram structures can co-exist in conjunction with a plethora of NN search algorithms without the need to substantially modify them. The main idea behind it is choosing a small number of pivot objects in the space, and pre-calculates the distances to their nearest neighbors. They provided a complete specification of such histogram structures and showed how to use the information they provide towards more effective searching. They showed that nearest neighbor histograms can be efficiently constructed and maintained, and when used in conjunction with a variety of algorithms for NN search, they can improve the performance dramatically.

ITQON et al. in [20] proposed a classifier, TFkNN, aiming at upgrading of distinction performance of KNN classifier and combining plural KNNs using testing characteristics. Their method not only upgrades distinction performance of the KNN but also brings an effect stabilizing variation of recognition ratio; and on recognition time, even when plural KNNs are performed in parallel, by devising its distance calculation it can be done not so as to extremely increase on comparison with that in single KNN. Yaokai and Akifumi in [21] proposed a new method, called Batch-Incremental Nearest Neighbor search algorithm, denoted B-INN search algorithm, which can be used to process the INN query efficiently. The B-INN search algorithm is different from the INN search algorithms in that it does not employ the priority query that is used in the existing INN search algorithms and is very CPU and memory intensive for large databases in high-dimensional spaces. Bao et al. in [22] improved the performance of KNN classification by introducing the tolerant rough set. They related the tolerant rough relation with object similarity. In their method, two objects are called similar if and only if these two objects satisfy the requirements of the tolerant rough relation. Hence, the tolerant rough set is used to select objects from the training data and constructing the similarity function. A genetic algorithm is used for seeking optimal similarity metrics. They showed that their algorithm can improve the performance of the KNN classification, and achieve a higher accuracy compared with the C4.5 system.

Recently, some works have been done which uses clustering to improve the performance of classification problem. Parvin et al. in [23] have proposed a classification ensemble method which uses clustering of classes. Also, Mohammadi et al. in [24] have used clustering ensemble technique to improve the performance of neural network ensembles. Here we use the clustering technique to enhance the performance of K-Nearest Neighbor.

Organization of the rest of the paper is as follows: Section 2 describes the proposed Nearest Cluster Algorithm. Demonstrating the experimental results is cited in section 3. Finally, section 4 concludes.

2. NEAREST CLUSTER ALGORITHM

Beforehand, the data set is divided into three partitions, Train, Evaluation and Test sets. The main idea of the presented method is assigning the data to the nearest cluster who is naturally consisted the neighbor points. To implement this idea, first, the train samples are clustered using a clustering ensemble method. Then, the label of cluster centers which are indicator of cluster members is determined using simple majority vote method.

For *i* = 1 to *Maxiteration*

// Clustering Ensemble
1. Partitioning the Train set into *k* cluster

// Majority Vote between labels of cluster members
2. Determining class label of cluster centers

// Assigning the label of nearest cluster center to any pseudo test samples in Evaluation set
3. Evaluating the quality of cluster centers

End

// Final classifier
4. Selecting the best clustering ensemble result

Figure 1. Pseudo-code of the Nearest Cluster Algorithm

In this method, a new test sample is assigned to the nearest cluster label. The quality of obtained clusters is evaluated applying the evaluation set. After determining the nearest cluster, its label is assigned to the sample. After that, in comparison with the ground truth label of data, the accuracy of the obtained classifier is derived. This procedure is iterated *Maxiteration* times. Finally, the best classifier is according to the evaluation set is selected as the final cluster centers. The pseudo code of the nearest cluster algorithm is shown in figure 1. Until here, the training of the NC classifier is finished. After here, any test samples are classified using this trained classifier.

In the rest of this section the proposed method is described in detail, answering the questions, how to cluster the train set, how to determine the class label of cluster centers and how to find the best clustering results for classifying the test samples.

2.1 How to Apply the Cluster Ensemble?

Clustering [25] is a fundamental problem in data mining with innumerable applications spanning many fields. The objective of clustering is to partition a set of unlabelled patterns into homogeneous clusters. Data clustering is an ill-posed problem when the associated objective function is not well defined, leading to fundamental limitations of generic clustering algorithms [25, 26].

The exploratory nature of clustering tasks strongly request efficient methods that would benefit from combining the power of many individual clustering algorithms. This is the focus of research on clustering ensembles, seeking a combination of multiple partitions that provides improved overall clustering of the given data. Clustering ensembles can outperforms what is typically achieved by a single clustering algorithm in several respects [27]:

Robustness. Better average performance across the domains and datasets.

Novelty. Finding a combined solution unattainable by any single clustering algorithm.

Stability and confidence estimation. Clustering solutions with lower sensitivity to noise, outliers or sampling variations. Clustering uncertainty can be assessed from ensemble distributions.

In this step, the goal is to cluster the train set into k individual partitions. Expectedly, K (capital) sample is existed in each cluster, if the number of clusters, k (minuscule) is chosen as the equation 1:

$$k = \frac{N}{K} \qquad (1)$$

Where N is the size of train set. If the number of custers, k, is derived from equation 1, in average, the NC algorithm would classify according to K nearest samples, like K-NN method.

Clustering ensembles can be formalized as follows. Let D be a data set of N data points in d-dimensional space. The input data can be represented as an $N \times d$ pattern matrix or $N \times N$ dissimilarity matrix, potentially in a non-metric space. The k-means algorithm with random initializations for selecting the k cluster centers is the common method for generating the component clusterings which is used here, too. Suppose that P_i is the clustering result of the i-th run of k-means; and $P_i(x)$ is the assigned cluster of sample x in i-th run.

Summarizing various clustering results in a co-association matrix is proposed by Fred in [28]. Co-association values represent the strength of association between objects by analyzing how often each pair of objects comes in the same cluster. The co-association similarity between two data points x and y is defined as the number of clusters shared by these points in the clusters of an ensemble. The equation 2 is applied for constructing the co-association matrix.

$$Co-association(x,y) = \frac{1}{H}\sum_{i=1}^{H} S(P_i(x), P_i(y)) \qquad (2)$$

Where the H is the number of iterations which the weak clustering is performed. The function S takes into account the similarity between the point x and y in one iteration of k-means. The equation 3, define this function.

$$S(a,b) = \begin{cases} 1 & a = b \\ 0 & a \neq b \end{cases} \qquad (3)$$

Usually, Hierarchical clustering algorithms are employed to find the final target partition from the co-association matrix. Some of the most common hierarchical consensus functions are: Single Linkage (SL), Average Linkage (AL) and Complete Linkage (CL) [25]. Here, the SL function is used for combining the weak clustering results.

Figure 2 shows the primary results of NCA in step 3 over evaluation data set applying clustering ensemble to obtain the cluster centers. The results are drawn after 50 individual runs of the steps 1,2 and 3, comparatively with KNN algorithm over the same data.

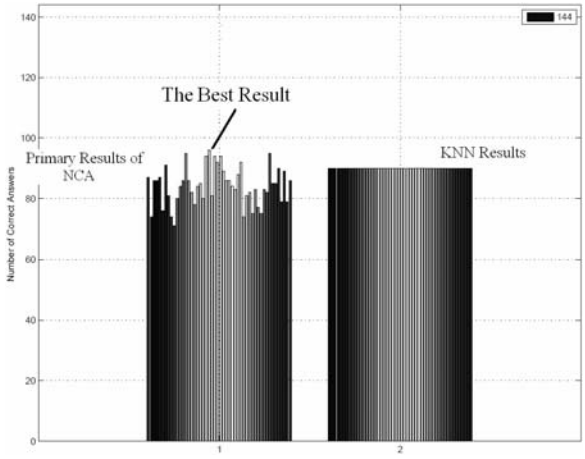

Figure 2. The raw results of clustering ensemble on MONK-2 data set, *K*=3, Mean=84.2, KNN=92, *Maxiteration*=50.

Note that, it is obvious the clustering ensemble is executed over train data without considering the class labels; however the of class tags are given. It is done to reveal the naturally neighborhoods in data feature space.

2.2 How to Determine the Label of Cluster Centers?

In this section, the class labels are used to specify the labels of the cluster centers which are explanatory points of the clusters. There are some combining methods for aggregating the class labels of the cluster members. When the individual votes of classifiers are crisp (not soft/fuzzy), the Simple Majority Vote is the common logical approaches which votes to class j if a little more than n/c of cluster members are assigned to class j [29]. Note that n and c stand for the number of cluster members and the number of classes, respectively. In this paper, the majority vote is used to assign a class label to cluster centers.

2.3 Evaluation of the Cluster Centers

There are some methods for evaluating the clustering result which use external, internal and relative indices [30]. External index needs further information to evaluate the clusters. In this paper, the evaluation set is used for measuring the performance of the different clusterings. It is a kind of using external index. First, the nearest cluster algorithm with the specific clustering is trained on the train set. Then, by executing the trained classifier on the evaluation data, the accuracy of this method is obtained using the true class labels of the evaluation data set.

As it is shown in figure 1, the steps 1, 2 and 3 are repeated *Maxiteration* times. In this method the best cluster centers according to the evaluation set are selected as a set of satisfactory good cluster centers among several times of performing the clustering techniques; however, the cluster centers obtained from any iteration can be considered as the solution. This method enhances both the accuracy and robustness of the KNN algorithm, significantly; however, it does not need to more both time and memory complexity in testing phase.

3. EXPERIMENTAL RESULT

This section discuses the experimental results and compares the new method with original KNN method.

3.1 Data sets

The proposed method is evaluated on two data sets, namely SA-Heart disease and MONK-2 [31]. None of the databases had missing values.

The SA-Heart data set which is obtained from www-stat.stanford.edu/ElemStatLearn is a retrospective sample of males in a heart-disease high-risk region of the Western Cape, South Africa. There are roughly two controls per case of CHD. Many of the CHD positive men have undergone blood pressure reduction treatment and other programs to reduce their risk factors after their CHD event. In some cases the measurements were made after these treatments. This data set has nine continuous features and two classes with the number of 463 instances. These data are taken from a larger dataset, described in [32].

The MONK's problem were the basis of a first international comparison of learning algorithms. There are three MONK's problems. The domains for all MONK's problems are the same. In this paper the MONK-2 is used for testing the performance of the NCE method. It has six features and two classes as well as the 432 instances.

These data sets have been used with many others for comparing various classifiers. In both data sets, the instances are divided into training, evaluation and test.

3.2 Experiments

All experiments are reported on Cross Validation procedure such as the data set is divided into three partitions. Then all six permutations of the partitions as train, validation and test set are executed. Finally, the average results of these examinations are reported.

The clustering ensemble applied has 50 individual members forming the co-association matrix. The simple *k*-means algorithm is the weak clustering used for this ensemble. The similarity between points (distance measure) is determined using Euclidian distance.

Table 1. Final results of NCA

	SA-Heart		MONK-2	
	Num. of Correct	St.dev.	Num. of Correct	St.dev.
KNN	94.67	±5.13	90.33	±4.63
NC Ensemble	**96.83**	**±4.22**	**91.00**	**±3.69**

Table 1 shows the results of the performance of classification using the presented method and traditional method comparatively. NCA is compared with original version of KNN. The nearest cluster algorithm both in, the number of test samples which are correctly classified and the standard deviation of the obtained performance outperforms the KNN method. In addition, because of the lower number of stored prototypes, these results are gained while the testing phase of the NCA has less computational burden both in the cases time and memory rather than the KNN algorithm.

4. CONCLUSION

In this paper, a new method for improving the performance of KNN classifier is proposed. The proposed method which is called NCA, standing for Nearest Cluster Algorithm, improves the KNN method both in time and memory burden. The NCA employs clustering technique to find the same groups of data in multi dimensional feature space. To gain more robust cluster centers, the clustering ensemble is used, effectively.

Despite of reducing training prototypes, the clustering technique can cause to finding the natural groups of data. In other hands, the natural neighborhoods can be successfully recognized by clustering ensemble technique. Moreover, unlike the KNN method which classifies any sample without considering the data distribution, based just on exactly *K* nearest neighbor, in the nearest cluster algorithm, the data is grouped into *k* clusters unequally, according to the data distribution and the position of data samples in feature space.

The NCA method is examined on two benchmark tasks: SA-Heart and MONK-2. Regarding to the obtained results, it can be concluded that the proposed algorithm is comparatively more accurate and robust than the KNN algorithm.

5. ACKNOWLEDGMENTS

This work is partially supported by Data and Text Mining Research group at Computer Research Center of Islamic Sciences (CRCIS), NOOR co. P.O. Box 37185-3857, Qom, Iran.

6. REFERENCES

[1] Fix, E., Hodges, J.L. Discriminatory analysis, nonparametric discrimination: Consistency properties. Technical Report 4,

USAF School of Aviation Medicine, Randolph Field, Texas, 1951.

[2] Cover, T.M., Hart, P.E. Nearest neighbor pattern classification. IEEE Trans. Inform. Theory, IT-13(1):21–27, 1967.

[3] Hellman, M.E. The nearest neighbor classification rule with a reject option. IEEE Trans. Syst. Man Cybern., 3:179–185, 1970.

[4] Fukunaga, K., Hostetler, L. k-nearest-neighbor bayes-risk estimation. IEEE Trans. Information Theory, 21(3), 285-293, 1975.

[5] Dudani, S.A. The distance-weighted k-nearest-neighbor rule. IEEE Trans. Syst. Man Cybern., SMC-6:325–327, 1976.

[6] Bailey, T., Jain, A. A note on distance-weighted k-nearest neighbor rules. IEEE Trans. Systems, Man, Cybernetics, Vol. 8, pp. 311-313, 1978.

[7] Bermejo, S. Cabestany, J. Adaptive soft k-nearest-neighbour classifiers. Pattern Recognition, Vol. 33, pp. 1999-2005, 2000.

[8] Jozwik, A. A learning scheme for a fuzzy k-nn rule. Pattern Recognition Letters, 1:287–289, 1983.

[9] Keller, J.M., Gray, M.R., Givens, J.A. A fuzzy k-nn neighbor algorithm. IEEE Trans. Syst. Man Cybern., SMC-15(4):580–585, 1985.

[10] www.hkjdhjbbscb.com

[11] R. O. Duda, P. E. Hart and D. G. Stork ,Pattern Classification , John Wiley & Sons, 2000.

[12] Thanh N. Tran, RonWehrens, Lutgarde M.C. Buydens , KNN-kernel density-based clustering for high-dimensional multivariate data , ELSEVIER 2005.

[13] Alippi, C.; Roveri, M. , Reducing Computational Complexity in k-NN based Adaptive Classifiers, IEEE International Conference on Volume , Issue , 27-29 June 2007 Page(s):68 – 71, 2007.

[14] Yang Song, Jian Huang, Ding Zhou, Hongyuan Zha, and C. Lee Giles , IKNN: Informative K-Nearest Neighbor Pattern Classification, PKDD 2007, LNAI 4702, pp. 248–264, 2007

[15] Domeniconi, C., Peng, J., Gunopulos, D.: Locally adaptive metric nearest-neighbor classification. IEEE Trans. Pattern Anal. Mach. Intell. 24(9), 1281–1285 (2002).

[16] Hastie, T., Tibshirani, R.: Discriminant adaptive nearest neighbor classification. IEEE Trans. Pattern Anal. Mach. Intell. 18(6), 607–616 (1996).

[17] Han, E.-H.S., Karypis, G., Kumar, V.: Text categorization using weight adjusted k -nearest neighbor classification. In: 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), pp. 53–65 (2001).

[18] Weinberger, K.Q., Blitzer, J., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. In: NIPS (2005).

[19] L Jin, N Koudas, C Li, NNH: Improving performance of nearest-neighbor searches using histograms, Springer Berlin / Heidelberg 2004.

[20] ITQON, KANEKO SHUN'ICHI, IGARASHI SATORU, Improving Performance of k-Nearest Neighbor Classifier by Test Features, springer Transactions of the Institute of Electronics, Information and Communication Engineers 2001.

[21] FENG Yaokai , MAKINOUCHI Akifumi **,** Batch-Incremental Nearest Neighbor Search Algorithm and Its Performance Evaluation(Databases), Batch-Incremental Nearest Neighbor Search Algorithm and Its Performance Evaluation(Databases), Vol.E86-D, No.9(20030901) pp. 1856-1867

[22] Yongguang Bao; Xiaoyong Du; Ishii, N. Improving performance of the k-nearest neighbor classifier bytolerant rough sets**,** Cooperative Database Systems for Advanced Applications, 2001. CODAS 2001. The Proceedings of the Third nternational Symposium on Volume , Issue , 2001 Page(s):167 – 171.

[23] H. Parvin, H. Alizadeh and B. Minaei-Bidgoli, Using Clustering for Generating Diversity in Classifier Ensemble, *International Journal of Digital Content: Technology and its Application, JDCTA*, ISSN: 1975-9339, 2009 (in press).

[24] Mohammadi M., Alizadeh H. and Minaei-Bidgoli B. (2008), "*Neural Network Ensembles using Clustering Ensemble and Genetic Algorithm*", 2008 Intl. Conf. on Convergence and hybrid Information Technology, ICCIT08, Nov. 11-13, IEEE CS, Korea.

[25] A.K. Jain, M. N. Murty, and P. Flynn, *Data clustering: A review*, ACM Computing Surveys, 31(3): 264–323, 1999.

[26] M. Law, A. Topchy and A. K. Jain, Clustering with Soft and Group Constraints, *Joint IAPR International Workshop on Syntactical and Structural Pattern Recognition and Statistical Pattern Recognition*, 2004.

[27] Clustering Ensembles, in Proc. *SIAM Intl. Conf. on Data Mining*, SDM 04, 379-390.

[28] A.L.N. Fred, *Finding Consistent Clusters in Data Partitions*, In Proc. 3d Int. Workshop on Multiple Classifier Systems. Eds. F. Roli, J. Kittler, LNCS 2364: 309-318, 2001.

[29] L. Lam and C.Y. Suen, Application of majority voting to pattern recognition: An analysis of its behavior and performance. *IEEE Transactions on Systems, Man, and Cybernetics*, 27(5):553–568, 1997.

[30] A.K. Jain, and R. C. Dubes; R. C. (1988). *Algorithms for Clustering Data.* Englewood Cliffs, NJ: Prentice-Hall.

[31] The Monk's Problems, Donor: Sebastian Thrun School of Computer Science Carnegie Mellon University Pittsburgh, PA 15213, USA, 1992.

[32] Rousseauw et al., South African Medical Journal, 1983.