AN OPTIMIZED ADABOOST ALGORITHM BASED ON K-MEANS CLUSTERING

Peng Zhang*
School of Information and Engineering, Tianjin Bohai Vocational Technical College
Tianjin, 300000, P.R.China
19396@163.com

Abstract
Classification plays an important role on data mining techniques. AdaBoost is a classic upgrade algorithm on data classification. The Optimization algorithm on AdaBoost emerged is endless. In this paper we make some improvement on the base of a nonlinear AdaBoost algorithm based on statistics for K-nearest neighbors. In the basic algorithm, the prediction accuracies were improved more or less while it took plenty of time to calculate the Euclidean distance between the instance and all the samples. So we raise an improved method to shorten the computing time with K-means Clustering algorithm, and the classification accuracies will be as accurate as the original. Experiment results show that improvement is an effective method while the number of samples is large. And the bigger the number of the samples is, the more time can be saving in the stage of classification for the instance.

1. Introduction
AdaBoost is a popular boosting classification algorithm in data mining. Since Freund and Schipare proposed the AdaBoost algorithm[1,2]. The improvement on AdaBoost mainly involves in two aspects: 1) adjust the weights of weak classifier in new ways, 2) improving training method to decrease the error rates of classifier or saving training time. The RADA algorithms[3], which suppress the weight of samples, have been misclassified. The FloatBoost algorithm[4,5] removed the weak classifiers with higher error rates. So the resulting stronger classifier has been consist of fewer weak classifier which has lower error rates. Fast Adaboost training algorithm[6] based on dynamic weight cut. They cut the sample with smaller weight while the training data set is large. Nonlinear AdaBoost algorithm based on statistics for K-nearest neighbors[7], which we will stick to KNN AdaBoost , is the algorithm we compared. It is mainly altering the way each weak classifier gets their weight. In our paper, we simplify the training data set about the KNN AdaBoost algorithm to keep error rates in the same level while reduce the classification time.

2. AdaBoost and Optimization Algorithm
AdaBoost is an iterative algorithm for classification, the core idea of the Adaboost is training the same dataset for weak classifiers, and then combine these weak classifiers to a combination classifier (strong classifier). The implementation of the algorithm is by changing the data distribution. The weight of each sample is determined by the weight of weak classifier at last and some parameters in equation 3. Then revised the weights for every sample in dataset and send to the next classifier to training. Finally we can get a final classifier when the weak classifier’s correct rates reach a limit value or the iteration number has achieved a limit value. Then we can integrate all the weak classifier as a final
decision classifier\[^8\].

KNN AdaBoost mainly choose the training samples according to the instance should be classified. Then the weights of weak classifiers are calculated by the selected samples and a strong classifier should be integrated. We can classify the instance by strong classifier.

2.1. The procedure of KNN AdaBoost

Data classification both have two stages, they are training and testing. In the stage of training we should achieved a stronger classifier, and then we can classify the samples according to the stronger classifier. Here we will make a description about the KNN AdaBoost.

2.1.1. Training

**Step 1.** Prepare the training dataset \( S \), \( \text{yet means its label.} \)

\[
S = \langle(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N) \rangle,
\]

\( x_i \in X \subseteq \mathbb{R}, y_i \in Y = \{-1, 1\} \) (1)

Then Select weak classification algorithm, such as Bayes, J48, and Random Forest. Then determine sample weight \( D \)

\[
D(w_{1,1}, w_{1,2}, \ldots, w_{1,N} ; w_{i,N}, w_i) = \frac{1}{N}, i \in \{1, 2, \ldots, N\}
\]

(2) Generally speaking the initial weight is \( 1/N \). The total iteration is \( T \).

**Step 2.** Call for weak classification algorithm, training the data set, and then get a weak classifier \( G_t(x) \), \( t \) means the current iterations.

**Step 3.** Calculate the error rate of the weak classifier using the equation (3):

\[
e_t = P(G_t(x_i) \neq y_i) = \sum_{i=1}^{N} w_i I(G_t(x_i) \neq y_i)
\]

(3)

**Step 4.** Calculate the weight of the classifier:

\[
\alpha_t = \frac{1}{2} \log \frac{1-e_t}{e_t}
\]

(4)

**Step 5.** Updated sample weight distribution \( D \):

\[
D(w_{t+1,1}, w_{t+1,2}, \ldots, w_{t+1,i}, \ldots, w_{t+1,N}),
\]

\[
w_{t+1,i} = \frac{w_t}{z} \exp(-\alpha_t y_i G_t(x_i))
\]

(5)

**Step 6.** Repeat steps 2 to 5, until the error rate is below a limit level or reach \( T \).

**Step 7.** The final strong classifier is obtained according to the weak classifier:

\[
G(x) = \sum_{t=1}^{T} \alpha_t G_t(x)
\]

(6)

2.1.2. Classification in KNN AdaBoost

Traditional AdaBoost have a strong classifier which has a stable weight of the weak classifiers. In the procedure of Testing, instance has been input to the strong classifier and returns a class which has the biggest weight. The KNN AdaBoost has a dynamic weight about the weak classifier. When the instance need to have a class, the Euclidean distance has been calculate from the instance to each of training firstly. Then we choice the K-nearest sample as the training data set. Training K samples and make a statistical about the amount of wrong classification times. \( e_t' \) is the weak classifier’s error rate.

\[
e_t' = \sum_{i=1}^{K} w_i I(G_i(x_i) \neq y_i) / K
\]

(7)
The weight of the weak classifier can be achieved through equation (8)

\[ \alpha_i = \frac{1}{2} \log \frac{1 - e_i}{e_i} \]  

(8)

KNN AdaBoost’s combined classifier is equation(9).

\[ G'(x) = \sum_{i=1}^{T} \alpha_i G_i(x) \]  

(9)

\( G'(X) \) is the terminal classifier for the instance, and we can get the class of the instance according to the equation (7) (8) (9)

2.2. K-means Clustering improved based on KNN AdaBoost

Due to clustering analysis has a simple, intelligent, the advantages of wide application range, has an important role in data mining, the main function is used to analysis and processing for large data sets. At present, the clustering algorithm has many types, which are widely used in: the algorithm based on hierarchy algorithm based on density method based on network, based on the classification of the algorithm. Clustering analysis has a wide range of application in many fields, such as used in image analysis, image search, artificial intelligence, pattern recognition.

As a kind of unsupervised clustering intrusion detection technology, the algorithm is based on two assumptions: first, assume that all the main activities in the process, the invading behavior relative to the body of the normal activity is very little; Second, intrusion behavior there are significant differences compared with the normal body activities. Based on these two assumptions, we can assume that a recorded the data set of main body activity, we mark a data in the data set as the initial clustering center, and then according to the data to the center distance, it is divided into a cluster, after the division, cancel a tag of the initial clustering center, according to cluster the data in the distance, calculate the cluster centers of new values. Get a tag clustering centers, and so again eliminate each cluster, according to the clustering center again divided into clusters, cluster and the choice of a new center value, marked as a clustering center, so repeated division and computing center value of the operation, until the data convergence.

Although clustering method based on classification exist many weakness is indisputable fact that, as it is sensitive to initial value abnormal in the algorithm, and easily influenced by data on various aspects; Easy to fall in with the optimum, and so on, in addition to the simple, fast and efficient, but the algorithm is especially suitable for used in the detection of large-scale data. So I used the improvement algorithm applied in intrusion detection.

K-means algorithm is a classical clustering algorithm based on partition of a. Correlation measure of this algorithm using Euclidean distance, said applied to seek the corresponding clustering center vector optimization classification problem, and makes the final optimization goal optimal. K-means algorithm is a kind of indirect clustering method based on similarity measure between samples, belongs to unsupervised learning method. The algorithm for k parameter, n object is divided into k clusters, in order to make within the cluster have high similarity, and the similarity between the clusters is low. Similarity calculation based on a bunch of objects in the average. This algorithm first randomly selected k objects, each object represents a cluster of center of mass. For the rest of the each object, according to the object with the clustering of the distance between the center of mass, it is assigned to the most similar with the clustering. The new then calculate every clustering center of mass. Repeat the process until the convergence criterion function. K-means algorithm is a kind of typical point modified iterative dynamic clustering algorithm, the point is based on the error sum of squares criterion function. Point change class center: a pixel samples, according to a certain rule belong to a particular set of classes, after will recalculate the average set of classes, and a new mean value as the center of condensed the next like element clustering; Batch-by-batch modify class center: in all samples of pixel classification by a group of class center, after calculation modifying all kinds of average again, as the next classification center of condensation.
As mentioned above, the algorithm in literature[7] change the weight of weak classifier into a series of dynamic ones which is decided by the statistics of the K- nearest neighbors of the instance. In this way, the weight of each basic classifier is more effective and the classification result is more accurate. While it will sacrifice a lot of time to calculate the Euclidean distance between instance and all the training samples. In this paper we insert a k-means clustering procedure before the stage of classifying the instance.

We know $X$ is space vector from equation(1). Now we need to divide all the samples into $M$ clusters and get $C=\{c_1,c_2,\cdots,c_M\}$. To be sure that equation(10) has a minimum value. $\tau_{nm}$ is has a label of 1 when $x_n$ is the member of $c_m$, or the label will be 0.

$$J = \sum_{n=1}^{N} \sum_{m=1}^{M} \tau_{nm} \| x_n - \mu_m \|^2$$

(10)

When $J$ has a minimum value, $u_k$ has a value of equation(11)

$$\mu_k = \frac{\sum_{n} \tau_{nk} x_n}{\sum_{n} \tau_{nk}}$$

(11)

In the stage of clustering we can get $M$ classes according to the full data set, then we choose $m (m<M)$ classes with whose centers are more closer to the instance. Then we treat the samples in the $m$ classes as the training data set. The KNN AdaBoost improved the weight of weak classifier. Fig1 is a schematic diagram of clustering results. The samples have been divided into 4 classes, and the circle point is the center of each class.

Fig1 the result of k-means clustering

Here is the concrete procedures about the method we arised in this paper.

| D, the training dateSet has been labeled by the class attribute |
|---------------------------------------------------------------|
| T, the max iteration                                          |

/\ Training

(1) Initialize the weight of element in $D$ as $D_1 = 1/N$
(2) For $t=1$ to $T$ do
(3) Training the $D_t$ to get the weak classifier $G_t(x)$
(4) Computing the errors rate of $G_t(x)$ as $e_t$
(5) If $e_t > 1/N$
(6) Go back (3)
(7) End if
(8) Computing the weight of $G_t(x)$ as $\alpha_t$
Update the weight of as $D_{t+1}$

End for

//Prediction

Input: $X$, the instance need to make a classification

Output: the class of $X$

1. Divide $D$ into $M$ classes with k-means clustering
2. Computing the Euclidean distance between the instance and the center of $M$ classes,
3. Choice the closer $m$ classes
4. Computing the Euclidean distance between the instance and the element in $m$
5. for $t=1$ to $T$
6. Computing the errors rate $e_t$, then get the new weight of the weak classifier
7. Endfor
8. Bring the instance into the strong classifier return the class which has the biggest weight

3. Experiment and result analysis

3.1. Experimental data and design

Experimental data come from the UCI. Table 1 point out the data set name, the number of samples, the number of attributes and the number of classes.

We choice NBTree, Random Forest, Bayes and other weak classification algorithm as the basis classification algorithm In this experiment, Then statistic the error times and time consumption about traditional AdaBoost, KNN AdaBoost $t$ and Clustering optimization based on KNN AdaBoost.

| Data set     | Samples | Features | Classes |
|--------------|---------|----------|---------|
| glass        | 214     | 10       | 7       |
| breast-cancer| 286     | 10       | 2       |
| house-votes-84| 435   | 17       | 2       |
| arrhythmia   | 452     | 279      | 16      |
| eucalyptus   | 736     | 20       | 5       |
| anneal       | 898     | 38       | 6       |
| hypothryid   | 3772    | 30       | 4       |
| satimage     | 6430    | 36       | 6       |
| adult        | 48842   | 15       | 2       |
| connect-4    | 67557   | 42       | 3       |

3.2. Experimental results and analysis

Table 2 is the results of experiment. Traditional Adaboost means the results of traditional AdaBoost. KNN means the result of the KNN AdaBoost, Clustering Algorithm means the optimized algorithm we proposed in this paper.

| Data Set/Weak classifier/Number | Clustering Algorithm | KNN Algorithm | Traditional Adaboost |
|--------------------------------|----------------------|---------------|---------------------|
|                                | E.N | P.T  | E.N | P.T  | E.N | P.T  |
| eucalyptus/NBTree/1            | 282 | 19.23| 280 | 20.05| 302 | 0.58 |
| glass/NBTree/2                 | 41  | 4.13 | 44  | 5.22 | 52  | 0.25 |
From the testing results we can get fig2. Conclude from fig1 that clustering algorithm and KNN AdaBoost algorithm is basically the same about their errors rates, and both lower than the traditional AdaBoost algorithm. Thus we can understand that the clustering optimized algorithm can guarantee the accuracy of KNN AdaBoost algorithm.

Fig3 is the contrast between KNN AdaBoost algorithm and the k-means clustering optimized algorithm. It shows the time they cost in making prediction. Compared with table1 we can find that when the training samples’ amount is small, clustering algorithm has not have obvious effect, while there is a large amount of training samples, it is obviously to save time. Why does this happen? This is because when the number of samples is small, in order to guarantee the accuracy, we need choice most or all the samples to be the training data set. So it is an extra time or the time saved in computing the Euclidean distance cannot counteract the clustering time. Thus the total time has increased.

However, facing large amounts of training samples, clustering algorithm eliminating unnecessary calculations and can save plenty of time, large may reach 70% to 80%.
4. Conclusion
In this paper we proposed an improved algorithm based on k-means clustering algorithm. The algorithm we refer was the nonlinear AdaBoost based on statistics for $K$-nearest neighbors. The algorithm we present is unstable when the training samples’ amount is small, and sometimes may consume more time because we save less time than the time needed for clustering. While the amount of training samples is large, the algorithm can save time significantly. So it is a good choice when it is a large amount of training data set.

References
[1] Freund Y, Schipare RE. A decision- -theoretic generalization of on-line learning and an application to boosting. *Computational Learning Theory: Second European Conference*, 1995, 23-37.
[2]  Freund Y, Schipare RE. Experiments with a new boosting algorithm. *International Conference on Machine Learning*. 1996: 148-156.
[3]  Cid HA, Salas R, Allende H, Nanculef R. Robust alternating AdaBoost. Progress in pattern recognition, *image analysis and applications*, 2007.
[4]  Li SZ, Zhang ZQ, Shum HY, Zhang HJ. FloatBoost Learning for Classification. *Proc. Neural Information Processing Systems*. Dec. 2002.
[5]  Li SZ, Zhang ZQ. FloatBoost learning and statistical face detection. *Pattern Analysis and Machine Intelligence*, 2004.
[6]  JIA Hui Xing, ZHANG YuJin. Fast Adaboost Training Algorithm by Dynamic Weight Trimming. *CHINESE JOURNAL L OF COMPUTERS*, 2009(02): 336-341.
[7]  GOU Fu, ZHENG Kai. Nonlinear AdaBoost algorithm based on statistics for $K$-nearest neighbors. *Journal of Computer Applications*, 2015, 35(9): 2579-2583.
[8]  Peter Harrington, Machine Learning in Action, 2013.6
[9]  HAN J W, KAMBER M. Data mining: concepts and techniques
[10] FAN M, MENG X, translated. 3rd ed. *Beijing: China Machine Press*, 2007: 263-266.