WCCI 2010 IEEE World Congress on Computational Intelligence
July, 18-23, 2010 - CCIB, Barcelona, Spain IJCNN

Negative Correlation Learning for Classification Ensembles

Shuo Wang and Huanhuan Chen and Xin Yao

***Abstract*— This paper proposes a new negative correlation learning (NCL) algorithm, called AdaBoost.NC, which uses an ambiguity term derived theoretically for classification ensembles to introduce diversity explicitly. All existing NCL algorithms, such as CELS [1] and NCCD [2], and their theoretical backgrounds were studied in the regression context. We focus on classification problems in this paper. First, we study the ambiguity decomposition with the 0-1 error function, which is different from the one proposed by Krogh et al. [3]. It is applicable to both binary-class and multi-class problems. Then, to overcome the identified drawbacks of the existing algorithms, AdaBoost.NC is proposed by exploiting the ambiguity term in the decomposition to improve diversity. Comprehensive experiments are performed on a collection of benchmark data sets. The results show AdaBoost.NC is a promising algorithm to solve classification problems, which gives better performance than the standard AdaBoost and NCCD, and consumes much less computation time than CELS.**

I. Introduction

Negative correlation learning (NCL) is a successful ensemble learning technique, since the role of diversity has been recognized [4] [5]. In addition to the bias and variance of each individual learner, the generalization error of an ensemble also depends on the covariance among the individuals. Hence, several NCL algorithms have been proposed to negatively correlate the errors made by each other explicitly based on neural networks. Cooperative ensemble learning system (CELS) proposed by Liu and Yao [1] is a representative algorithm. It has achieved empirical success in both regression and classification problems. Unlike Bagging [6] and Boosting [7], the idea of CELS is to encourage the individual networks to learn different aspects of a given data set cooperatively and simultaneously. It emphasizes diversity, the degree of disagreement, among the individual networks explicitly by introducing a penalty term that provides the missing gradient component (variance and covariance terms) within the ensemble MSE [8].

Although CELS has adequate theoretical support in the regression context and has been applied to solve classification problems, a theoretical gap in the classification context still needs to be filled. An ambiguity term [3] derived for regression ensembles is used as the penalty in CELS, but it is difficult to obtain for classification ensembles. Very recently, Chen [9] gave an ambiguity decomposition for binary classification problems with the 0-1 error function. In this paper, we will explain this ambiguity decomposition, and extend it to multi-class cases.

Shuo Wang and Xin Yao are with the School of Computer Science, University of Birmingham, Birmingham, B15 2TT, UK (email: {s.wang, x.yao}@cs.bham.ac.uk).
Huanhuan Chen is with the School of Computing, University of Leeds, Leeds, LS2 9JT, UK (email: {H.H.Chen}@leeds.ac.uk).

In addition to the lack of some theoretical explanations for classification, CELS suffers some other drawbacks. So far, it is only applicable to neural networks. Other base learners are not suitable to its training procedure. The pattern-by-pattern weight-updating strategy results in very high computation cost. The training can take a long time before it converges to the expected error threshold for a very large data set. To speed up the learning process, another NCL algorithm, negative correlation learning via correlation-corrected data (NCCD), was proposed by Chan and Kasabov [2]. The idea is to incorporate error correlation information into the training data instead of every network's error function. It reduces the updating times of information exchange, and makes parallel model implementation possible. However, it still requires that the base learners are capable of processing real-valued outputs, and our experiments will show that its performance is very sensitive to the parameter settings, involving a scaling coefficient λ and the update interval.

Considering the above problems, this paper proposes a new NCL algorithm for classification ensembles, AdaBoost.NC. As the name suggests, we exploit AdaBoost's flexibility to overcome the above disadvantages of the existing NCL algorithms, and attempt to solve the reported overfitting problem of AdaBoost by introducing diversity [10] [11]. Similar to AdaBoost, AdaBoost.NC is independent of the choice of base learners. It builds classifiers sequentially and samples the data space automatically, which provides us with a chance to incorporate the information of the current diversity level explicitly. Low diversity will be penalized by using the ambiguity term derived in this paper. Different from AdaBoost, CELS and NCCD, AdaBoost.NC introduces the error correlation information into the weights of the training data, when the updating happens, or in other words, after each classifier is constructed. Thus, information exchange is much easier and there is no need to modify the original training examples. The computation time can be shortened significantly. Only one main parameter, the scaling coefficient of the penalty term λ, needs to be decided. We expect AdaBoost.NC can produce better accuracy than AdaBoost and needs shorter computation time than CELS and NCCD.

In this paper, we concentrate on classification problems. First, we will explain and extend Chen's ambiguity decomposition [9]. Then, we will introduce our new NCL algorithm, AdaBoost.NC, which is tested on both binary-class and multi-class tasks. The experimental results are quite positive. It has better generalization performance than the standard AdaBoost and NCCD in most of the cases, and gives competitive results with CELS. Its computation time is significantly shorter than CELS and NCCD, and nearly the same as the standard AdaBoost.

The rest of this paper is organized as follows: Section II gives some background descriptions about AdaBoost and existing NCL algorithms. Section III derives the ambiguity term for classification ensembles and describes AdaBoost.NC algorithm. Section IV presents the experimental analysis and comparing results. Finally, section V makes the conclusions.

II. Background

Ensemble learning is to construct multiple estimators for the same learning tasks and then aggregate their predictions when an unknown example comes. We begin this section by introducing AdaBoost [7], one of the most popular ensemble algorithms. Following this, we conclude some recent classification ensemble methods briefly. Since the role of diversity of an ensemble was recognized [4] [5], a few NCL methods and their theoretical support have been proposed in the regression context. Then, two representative NCL algorithms, CELS [1] and NCCD [2], will be introduced.

A. AdaBoost and Classification Ensembles

AdaBoost is a successful ensemble technique in ensemble learning area, proposed by Freund and Schapire [7]. It builds base learners sequentially and emphasizes the hardest examples. It is achieved by a weight updating rule. A set of weights is maintained over the training data. Misclassified examples by the previous learner get their weights increased for the next iteration. Consequently, harder examples possess larger weights that have higher possibility to be selected for the current training.

Comprehensive empirical and theoretical studies have proved AdaBoost can significantly reduce the error of any weak learner in terms of both bias and variance, and it is more effective at bias reduction than variance reduction [12]. However, degradation of the generalization performance is observed in some cases [10] [11]. The weight vectors can become very skewed, which may lead to undesirable bias towards some limited groups of data. Freund and Schapire attribute it to overfitting [13]. They restrict training epoch T to keep final hypothesis simple and achieve lower generalization error. Some related studies also find that AdaBoost can produce diverse ensemble at the first few training epochs, but diversity drops as more classifiers are added in. So, they suggest that it could be beneficial to stop training progress early [14].

Other ensemble algorithms, such as Bagging [6], Stacking [15], and Mixtures of Experts [16], and a number of variations of them, have been proposed to solve binary classification problems. Recent extension to multi-class problems includes SAMME [17], GAMBLE [18], and MSmoothBoost [19]. Another large group of ensemble algorithms is to solve class imbalance problems, where a data domain has skewed class distribution and it causes classification difficulty. Due to the flexibility an ensemble method can have, many ensemble solutions have appeared by re-balancing the uneven situation from either the data level or the algorithm level, for instance, SMOTEBoost [20], EasyEnsemble and BalanceCascade [21]. In short, ensemble techniques, especially AdaBoost-related algorithms, have become a popular tool in machine learning.

B. Negative Correlation Learning: CELS and NCCD

It has been commonly agreed that diversity is a successful factor of ensemble algorithms. Different opinions from multiple classifiers are expected to reduce the generalization error. Both Bagging and Boosting try to introduce diversity in an implicit way. Different from them, NCL encourages diversity explicitly by adding a correlation penalty term to each network's error function. Its theoretical support comes from the bias-variance-covariance decomposition [5] and ambiguity decomposition [3] in the regression context. Although empirical successes have been achieved to solve classification problems, there is still a lack of convincing proof to link diversity and the generalization performance. Various diversity measures are therefore brought forward [22], and related to the ensemble margin [23] and overall accuracy under some assumptions [24]. Brown et al. [25] gave a comprehensive discussion for both regression and classification cases.

Cooperative ensemble learning system (CELS) [1], proposed by Liu and Yao, is a successful NCL algorithm. CELS trains and combines individual networks simultaneously, aiming at the best result for the whole ensemble instead of every single one. The individuals are forced to learn different parts of training data. It is achieved by the unsupervised penalty term p_t. It provides the missing gradient components that correspond to the average variance and covariance terms within the ensemble MSE [8]. The individuals tend to be negatively correlated. In more detail, the error function e_t for network h_t is defined by,

$$e_t = \frac{1}{2}\left(h_t - y\right)^2 + \lambda p_t, t = 1, \dots T, \tag{1}$$

where y is the expected value of training example x. λ is the scaling coefficient to adjust the strength of the penalty in range $[0, 1]$, and the ensemble size is T. The penalty term p_t is expressed as

$$p_t = \left(h_t - H\right) \sum_{k \neq t} \left(h_k - H\right), \tag{2}$$

where H is the final output by combining the decisions from the individuals.

However, this training mechanism assumes the base learners are neural networks with back propagation strategy. Besides, the pattern-by-pattern weight updating method makes its training procedure much slower than other ensemble techniques, especially when the problem domain or the feature space is very large. Those hinder this algorithm from being more widely used.

To overcome the problem of long running time, Chan and Kasabov [2] proposed another NCL approach, Learning via Correlation-Corrected Data (NCCD) algorithm. NCCD embeds penalty values to every training example instead of the error function. After certain epochs of training, the expected outputs of training data are updated, named correlation-corrected data (C-C data). C-C data will join the next training period. Thus, NCCD significantly reduces the network communication caused by the exchange of correlation information and shortens the training time. C-C data have new targets y_{update} updated by

$$y_{update} = \frac{y - 2\lambda H}{1 - 2\lambda}, \tag{3}$$

where λ ranges in $[0, 0.5)$. The individuals can be implemented either synchronously or asynchronously. Base learners are not restricted to back propagation any more, but the capability of processing data with real-valued outputs is still necessary. Their experiments showed that NCCD has comparable generalization performance to CELS, and NCCD is much simpler and more efficient.

Besides, Chen and Yao proposed a regularized negative correlation learning algorithm, and made use of Bayesian inference to infer the explicit regularization parameters for large noisy data [26]. They formulated the regularized negative correlation learning as a multi-objective evolutionary learning problem [27]. A multi-objective evolutionary algorithm is used to search effectively the best trade-off among these objectives without searching for the combination parameters to weigh these objectives. Excellent performance has been obtained in their both implementations.

III. Negative Correlation Learning for Classification Ensembles

In this section, we explore the ambiguity decomposition for classification ensembles, including both two-class and multi-class problems. To our best knowledge, it is still an ambiguous issue in all existing papers, although the one for regression has been fully discussed. Then, we describe our NCL algorithm, called AdaBoost.NC.

A. Ambiguity Decomposition

Taking a closer look of the penalty term in CELS, Eq.(2) can be rearranged as

$$p_t = -\left(h_t - H\right)^2, \tag{4}$$

which is in fact the ambiguity term proposed by Krogh and Vedelsby [3]. Their ambiguity decomposition is achieved for regression tasks. We follow the notation in the previous section. For a single data input, the weighted average quadratic error of the ensemble is

$$\left(H - y\right)^2 = \sum_t \alpha_t \left(h_t - y\right)^2 - \sum_t \alpha_t \left(h_t - H\right)^2. \tag{5}$$

The second term is referred to as the "ambiguity" (denoted as "amb" in this paper), the variance of the weighted ensemble around the mean. It measures the disagreement among the individuals. Thus, CELS is maximizing this ambiguity term to balance the tradeoff between the accuracy and diversity directly during training. If a uniform weight is assumed,

$$amb = \frac{1}{T}\sum_{t=1}^{T}\left(h_t - H\right)^2. \tag{6}$$

The difficulty of such error decomposition for classification ensembles leads to various diversity measures and a number of work about the relationship between diversity and the overall accuracy [28] [29]. Those measures have been proved to exhibit strong relationships among themselves and similar correlation with the overall performance [22]. Very recently, Chen [9] first proposed an ambiguity decomposition based on the 0-1 error function for binary classification problems and introduced an ambiguity term as a diversity measure. They showed that the ambiguity measure has the highest correlation with the generalization error. For a data domain with two class labels $\{+1, -1\}$ (i.e. positive and negative classes), their ambiguity term is formulated as [9]

$$amb = \frac{1 - s}{2} y \cdot H, \tag{7}$$

$$s = \frac{1}{T}\left|\sum_{t=1}^{T} h_t\right|, \tag{8}$$

where h_t and $H \in \{+1, -1\}$. Term s measures the difference between the numbers of positive and negative votes. Smaller s implies higher diversity degree.

To make it applicable to all classification tasks including multi-class data domains, we redefine the ambiguity decomposition with the correct/incorrect decision. The practical output $h_t(x)$ is 1 if x is labeled correctly by h_t, and -1 otherwise. It is worth noting that $H \neq sign\left(\sum_{t=1}^{T} \alpha_t h_t\right)$ any more when majority voting is applied, since there are more than two classes. If the error function follows the definition in [9],

$$error(h) = \frac{1}{2}\left(1 - h\right), h \in \{h_1, \dots h_T, H\}, \tag{9}$$

then the ambiguity term will be

$$amb = error(H) - \sum_{t=1}^{T} \alpha_t error(h_t) = \frac{1}{2}\sum_{t=1}^{T} \alpha_t \left(H - h_t\right). \tag{10}$$

If the individual classifiers are uniformly weighted, amb can be re-expressed as

$$amb = \frac{1}{2T}\sum_{t=1}^{T}\left(H - h_t\right). \tag{11}$$

The ambiguity term is related to the difference between the numbers of correct and incorrect votes. Comparing Eq.(6) and Eq.(11), they have a similar role in the generalization error, but an obvious difference exists. amb in regression ensembles is an unsupervised term that measures the magnitude of the difference, whereas amb in classification ensembles depends on the sign of the term $(H - h_t)$ that measures the difference between an individual and the ensemble.

To encourage diversity for classification ensembles, a new NCL algorithm AdaBoost.NC is developed. Analogous to CELS, the ambiguity term in Eq.(11) is used as the penalty. However, the distinction between the two ambiguity terms makes us hesitate about the form of the penalty term, using amb or $|amb|$? If we follow the NCL idea strictly, the original amb should be applied to enlarge the ambiguity as much as possible and minimize the generalization error; if considering the meaning of Eq.(6) in CELS, the sign of amb should be ignored and use the absolute value $|amb|$. With some preliminary experiments, $|amb|$ actually works better than amb. The magnitude is thus used to compute the penalty.

B. AdaBoost.NC

So far, amb for making the classifiers negatively correlated has been derived for classification ensembles. Now, the problem is how to use it. To overcome the limitations of CELS and NCCD, AdaBoost.NC is proposed by utilizing the nice properties of AdaBoost [7], which can be regarded as a kind of cost-sensitive Boosting. The current diversity degree will be the cost, obtained directly from the ensemble.

AdaBoost.NC exchanges correlation information during the sequential training procedure. After each single classifier is built, the difference among the current classifiers is measured by the penalty term and combined into the weights of training examples with the misclassification information together. Then, they are used to build the next classifier. The penalty is introduced into the weight-updating rule. Therefore, both classification errors and the low diversity degree will be punished by rising weights. Table I presents the pseudo-code of AdaBoost.NC.

TABLE I
ADABOOST.NC ALGORITHM

```
Given training data set ${(x_1, y_1), \dots, (x_i, y_i), \dots, (x_m, y_m)}$,
initialize data weights $D_1(x_i) = 1/m$; penalty term $p_1(x_i) = 1$.

For training epoch $t = 1, 2, \dots, T$:
Step 1. Train weak classifier $h_t$ using distribution $D_t$.
Step 2. Get weak classifier $h_t$: $X \rightarrow Y$.
Step 3. Calculate the penalty value for every example $x_i$: $p_t(x_i)$.
Step 4. Calculate $h_t$'s weight $\alpha_t$ by error and penalty.
Step 5. Update data weights $D_t$ and obtain new weights $D_{t+1}$
by error and penalty.

Output the final ensemble:
$H(x) = \arg\max_{\mathbf{y}} \sum_{t=1}^{T} \alpha_t \|h_t(x) = y\|$.
(Define $\|\pi\|$ to be 1 if $\pi$ holds and 0 otherwise.)
```

From Table I, we can see that AdaBoost.NC is independent of the selection of base learning algorithms, because the weights are maintained separately from the algorithm. There is no need to modify the training examples themselves either. Besides, the weight updating happens at the ensemble level, rather than at the algorithm level or at the data level. Therefore, the total updating times with diversity information only depends on the size of the ensemble, which is much simpler than CELS and NCCD. A lot of training time will be saved. AdaBoost.NC could be a good way to make negative correlation learning more flexible and faster.

The updates of α_t and D_t in step 4 and 5 are decided by the misclassification error and penalty values with the information of current diversity degree at the same time. In the standard AdaBoost [7], the weight updating rule is

$$D_{t+1}(x_i) = \frac{D_t(x_i)\, exp\left(-\alpha_t h_t(x_i)\, y_i\right)}{Z_t}, \qquad (12)$$

where Z_t is the normalization factor and

$$\alpha_t = \frac{1}{2}\log\left(\frac{\sum_{i, y_i = h_t(x_i)} D_t(x_i)}{\sum_{i, y_i \neq h_t(x_i)} D_t(x_i)}\right), \qquad (13)$$

in which way the ensemble's training error can be bounded.

In AdaBoost.NC, there are two possible ways to introduce penalty term p_t into the rule: inside and outside the exponent [30]. If it is placed inside the exponent, the penalty term will make a negative effect, which is not our intention. The correctly classified examples that the classifiers have the least disagreement on get the largest weight decrease, but in fact they are supposed to be emphasized. Therefore, we choose to put p_t outside the exponent. AdaBoost.NC defines the penalty term and weight updating rule as,

$$p_t = 1 - |amb|$$

$$D_{t+1}(x_i) = \frac{\left(p_t(x_i)\right)^{\lambda} D_t(x_i)\, exp\left(-\alpha_t h_t(x_i)\, y_i\right)}{Z_t}$$

λ is to control the strength of the penalty term. To bound the overall training error and minimize Z_t [30], the weight of each classifier in Step 4 is determined by

$$\alpha_t = \frac{1}{2}\log\frac{\sum_{i, y_i = h_t(x_i)} \left(p_t(x_i)\right)^{\lambda} D_t(x_i)}{\sum_{i, y_i \neq h_t(x_i)} \left(p_t(x_i)\right)^{\lambda} D_t(x_i)}; \qquad (14)$$

The penalty term considers the magnitude of ambiguity, and ranges in $[1/2, 1]$. When $|amb|$ is small, which means the individuals tend to have the same opinion on one example, p_t will become large and this example is likely to be boosted in the next round. Simply to say, the misclassified examples that receive more same votes from individual classifiers get a larger weight *increase*; The correctly classified examples that gain more disagreement opinions get a larger weight *decrease*.

IV. EXPERIMENTAL STUDIES

In this section, we compare AdaBoost.NC to some state-of-art methods, including the standard AdaBoost, CELS and NCCD. Comprehensive experiments are designed and discussed on 10 two-class data domains and 4 multi-class data domains.

A. Experimental Design

In the experiments, MLP network and C4.5 decision tree are chosen as the base learners. As mentioned earlier, CELS and NCCD are designed for and tested on neural networks. It is necessary to do the comparison by using MLP. Each network is trained by 250 epochs. The updating epoch of NCCD is set to 20, which is a rough estimate, because the optimal update interval is problem-dependent. Furthermore, decision tree is recognized as the "ideal" learner with AdaBoost, since boosting decision trees improves their accuracy dramatically and tree-based algorithms come closest to an off-the-shelf method [31]. From this point, we also examine the performance of AdaBoost.NC with C4.5 as the base learner. For the consideration of running time, every ensemble consists of 9 individual classifiers in this paper, where the odd number can avoid even voting happening. More discussion about the ensemble size will be included in our future work.

The scaling coefficient λ requires the range $[0, 1]$ in CELS and $[0, 0.5)$ in NCCD. In our experiments, CELS sets λ to $\{0.25, 0.5, 1\}$ and NCCD sets λ to $\{0.25, 0.4\}$. Both of them use "winner-take-all" to combine the outputs as demanded by the original papers. The range of λ in AdaBoost.NC, however, is not limited, because the penalizing mechanism decides the performance is not very sensitive to the change of λ. It ranges from 0.25(conservatively) to 12(aggressively) in our experiments. The best results are chosen for the comparisons.

TABLE II
SUMMARY OF CLASSIFICATION DATA SETS.

Name	Train	Test	Attributes	Classes
promoter	84	22	57	2
sonar	166	42	60	2
ionosphere	280	71	34	2
house-votes-84	348	87	15	2
crx	489	201	15	2
breast-w	559	140	9	2
pima	614	154	8	2
german	800	200	20	2
hypothyroid	2530	633	25	2
insurance	5822	4000	85	2
soybean-large	307	376	35	19
vowel	528	462	10	11
segmentation	1848	462	19	7
satimage	4435	2000	36	6

The experiments are conducted on a collection of UCI classification tasks [32], including 10 two-class data sets and 4 multi-class data sets. A summary is given in Table II, ordered by the size of the problems. The last four have more than two classes. Some data sets are provided with a test set. For the others, we randomly sample 80% of the data as the training set. The rest are for the test. A cross-validation method may be more appropriate for some data problems, which will be applied in our future work. We reran each algorithm 10 times on every data set, and averages computed. Statistical T-test with 95% confidence level is applied to check the significance of the difference among those algorithms.

B. Results and Analysis

AdaBoost.NC is compared to the standard AdaBoost, CELS and NCCD. We output the test error rate and computation time from each algorithm. AUC values are further examined on the two-class data sets for AdaBoost.NC algorithm and the standard AdaBoost. The computation environment is windows Xp with Intel Core 2 Duo 2.13GHz and 1G RAM. All the algorithms are implemented in Java. The results are shown in Tables III - V.

1) Generalization error: Now, we examine the generalization error. Table III presents the performance of these algorithms on the 10 two-class benchmarks. When MLP is the base learner, AdaBoost.NC always gives better results than the standard AdaBoost on the 10 data sets, where 5 wins are significant. Comparing to CELS, AdaBoost.NC produces better performance on 5 out of 10 data sets, where 2 wins are significant; CELS wins once significantly and the other four are ties. Comparing to NCCD, AdaBoost.NC wins on 7 data sets significantly, and the other three are ties. When C4.5 is applied as the base learner, similar to the results of MLP, our algorithm gives quite encouraging outputs. It significantly outperforms the standard AdaBoost in the last 6 data sets with larger sizes, and 4 ties for the others.

In the two-class cases, generally speaking, AdaBoost.NC is a promising algorithm. It improves the standard AdaBoost and outperforms NCCD significantly in more than half of the data sets. NC and CELS are quite competitive. In NC, the penalty term is introduced into the weight updating rule outside the exponent, and the penalty is computed by using $1-|amb|$. It is analogous to the penalty form of CELS, which does not depend on the sign of the ambiguity. In addition, we observe from the experiments that AdaBoost.NC is not very sensitive to or even benefits from the large λ values. In some cases, it achieves the best result when λ is set to 12. λ in CELS and NCCD is bounded in a specific range, which makes them more sensitive to the parameter setting than AdaBoost.NC. Especially for NCCD, a slight change of λ and the update interval can make it much worse than the optimal setting. We conjecture the reason could be NCCD adjusting the training data directly, which may induce very inaccurate classifiers. The experimental results of different parameter settings are not presented in this paper for the space consideration.

To explain why AdaBoost.NC performs better than the others, we can understand the algorithm from another point of view. As we have described in Section III, by applying AdaBoost.NC strategy, the misclassified examples with low disagreement level from the ensemble get the largest weight increase and the correctly classified examples with high disagreement level get the largest weight decrease. In other words, the more "difficult" examples within the misclassified part and the "easier" examples within the correctly classified part are emphasized. Different from the traditional AdaBoost that only focuses on the misclassified ones, "easier" examples

TABLE III

PERFORMANCE OF ADABOOST.NC (ABBR. NC), STANDARD ADABOOST (ABBR. ADA), CELS AND NCCD ON THE TWO-CLASS DATA SETS WITH MLP AND C4.5 AS BASE LEARNERS. MEAN AND STANDARD VARIATION OF THE TEST ERROR (ABBR. ERR %), AND MEAN COMPUTATION TIME (IN SECONDS) ARE COMPARED. THE METHOD WITH THE LOWEST TEST ERROR UNDER EACH LEARNING ALGORITHM IS IN BOLDFACE.

Name		MLP				C4.5	
		NC	Ada	CELS	NCCD	NC	Ada
promoter	Err	17.727±4.520	24.091±4.312	**12.727±1.917**	15.000±9.103	**10.454±4.815**	10.909±6.843
	Time	19.6	19.5	**1666.4**	267.3	**0.014**	0.01
sonar	Err	10.952±2.300	12.381±3.214	**7.619±1.004**	22.857±15.681	21.667±5.318	**21.190±5.435**
	Time	43.2	43.1	**267.1**	271.4	0.212	**0.206**
ionosphere	Err	4.647±0.950	5.211±0.950	4.366±0.445	**4.084±2.523**	**1.831±0.680**	2.253±0.984
	Time	26.5	26.5	174.6	**25.0**	**0.225**	0.205
house-votes-84	Err	**2.528±0.484**	3.218±0.726	3.333±0.363	4.023±1.817	3.793±1.217	**3.563±1.006**
	Time	**9.9**	10.0	72.3	9.8	0.059	**0.051**
crx	Err	**15.671±0.853**	19.055±1.502	16.667±0.586	18.358±1.454	**16.069±1.050**	18.109±0.943
	Time	**12.1**	12.0	426.9	69.7	**0.111**	0.127
breast-w	Err	**1.071±0.505**	1.286±0.451	3.214±1.398	33.071±28.081	**1.357±0.710**	2.143±0.588
	Time	**7.2**	7.1	1939.6	298.5	**0.035**	0.028
pima	Err	24.025±1.960	25.130±1.437	**23.246±1.396**	49.221±17.987	**20.649±1.779**	23.961±2.340
	Time	7.5	7.4	**60.1**	8.4	**0.20**	0.25
german	Err	**24.900±1.776**	26.900±1.744	25.250±1.585	31.850±6.200	**25.400±1.926**	27.600±2.389
	Time	**31.7**	31.7	1351.2	209.1	**0.21**	0.26
hypothyroid	Err	**2.227±0.157**	2.275±0.260	2.432±0.388	2.796±0.372	**0.553±0.083**	0.948±0.235
	Time	**137.7**	137.6	957.6	185.0	**0.81**	1.43
insurance	Err	6.450±0.177	8.080±0.171	**6.293±0.228**	6.407±0.271	**6.225±0.079**	8.120±0.225
	Time	2874.4	2852.7	**17930.0**	3742.2	**52.3**	52.5

in our algorithm are selected to help the classification on the more "difficult" examples.

To make our experiments more convincing, we further compute the AUC values [33] produced by the standard AdaBoost and AdaBoost.NC with C4.5 as the base learner over the 10 two-class data sets. AUC has been showed to be a statistically consistent and more discriminating measure than the generalization error [34]. The comparing results are presented in Table IV. Every case with significant difference is marked by '*'. We can see that AUC is more "strict" than the test error rate. For the first 9 data sets, the number of cases, where AdaBoost.NC outperforms the standard AdaBoost significantly, reduces to three data sets. The other six are ties. An interesting phenomenon occurs in data set "insurance". Although AdaBoost.NC produces a significantly lower error rate than the standard AdaBoost from Table III, it has a significantly worse AUC value. We conjecture that it is related to the very imbalanced class distribution of this domain. The reason will be investigated as a part of our future work.

TABLE IV

AUC COMPARISON OF ADABOOST.NC (ABBR. NC) AND STANDARD ADABOOST (ABBR. ADA) ON THE TWO-CLASS DATA SETS WITH C4.5 AS BASE LEARNERS. MEAN AND STANDARD VARIATION ARE PRESENTED. SIGNIFICANT DIFFERENCE IS MARKED BY '*'.

Name	NC	Ada	Significance
promoter	0.975±0.019	0.951±0.038	
sonar	0.881±0.051	0.860±0.072	
ionosphere	0.990±0.010	0.993±0.009	
house-votes-84	0.995±0.001	0.986±0.009	
crx	0.902±0.011	0.879±0.014	*
breast-w	0.993±0.003	0.993±0.003	
pima	0.865±0.014	0.819±0.026	*
german	0.778±0.022	0.733±0.025	*
hypothyroid	0.983±0.005	0.985±0.012	
insurance	0.616±0.012	0.634±0.012	*

Since the ambiguity decomposition is extended theoretically to the multi-class tasks in this paper, we also examine AdaBoost.NC on some multi-class data sets (Table V). Similar to the two-class cases, it performs better than the standard AdaBoost, where one is significant in the MLP settings and three in C4.5. Except the soybean-large data set, where NCCD beats the other three methods, NC wins CELS and NCCD significantly on two of them (vowel and segmentation). Besides, we notice that CELS doesn't show any advantage over these data sets. In fact, its ability of solving multi-class problems hasn't been fully explored by previous studies. More experiments need to be done in multi-class cases with different settings, and it is necessary to compare with other existing multi-class solutions. It is also interesting to know what kind of data sets our algorithm is the most effective on.

2) Computation time: According to Tables III and V, AdaBoost.NC is the winner for the running time among all the negative correlation learning solutions with no doubt. It doesn't bring extra computation cost comparing to the standard AdaBoost. It is much faster to build a model than CELS and NCCD. It doesn't need any complicated parameter settings, such as update interval, to decide when to exchange diversity information. Running time is only decided by the size of an ensemble, which is necessary to every ensemble algorithm. Because it allows choosing base learning algorithms freely, the training time can be further shortened by applying C4.5. Therefore, we conclude that

TABLE V

PERFORMANCE OF ADABOOST.NC (ABBR. NC), STANDARD ADABOOST (ABBR. ADA), CELS AND NCCD ON THE MULTI-CLASS DATA SETS WITH MLP AND C4.5 AS BASE LEARNERS. MEAN AND STANDARD VARIATION OF THE TEST ERROR (ABBR. ERR %), AND MEAN COMPUTATION TIME (IN SECONDS) ARE COMPARED. THE METHOD WITH THE LOWEST TEST ERROR UNDER EACH LEARNING ALGORITHM IS IN BOLDFACE.

Name		MLP				C4.5	
		NC	Ada	CELS	NCCD	NC	Ada
soybean-large	Err	10.080±1.124	10.665±1.281	10.505±0.258	**7.952±0.714**	**7.553±1.457**	8.244±1.795
	Time	71.2	68.8	1251.5	**226.3**	**0.17**	0.15
vowel	Err	**44.826±1.834**	46.147±1.679	47.792±2.456	49.329±1.846	**51.428±2.217**	53.030±2.105
	Time	**23.2**	23.1	150.1	45.7	**0.40**	0.38
segmentation	Err	2.575±0.387	**2.489±0.371**	3.463±0.478	3.376±0.423	**2.099±0.323**	2.316±0.250
	Time	107.8	**107.8**	717.8	374.2	**1.3**	1.2
satimage	Err	**9.735±0.382**	10.175±0.408	9.850±0.309	9.785±0.233	**10.315±0.395**	10.700±0.297
	Time	**607.4**	606.5	3702.5	2225.4	**9.7**	9.6

AdaBoost.NC inherits the advantages of NCL methods and overcomes their problems with low computation cost.

V. CONCLUSIONS

This paper proposes a new NCL algorithm for classification ensembles, AdaBoost.NC. It encourages diversity explicitly by building classifiers sequentially and introducing the correlation information into the weights of training examples. It is more flexible and simpler than other NCL algorithms, and applicable to both two-class and multi-class problems. Different from the penalty form in CELS, the ambiguity term used to measure the current diversity degree in AdaBoost.NC is derived theoretically for classification ensembles, which is another main work in this paper. The algorithm can be regarded as a "cost-sensitive" solution, where the diversity degree is the "cost".

AdaBoost.NC shows better generalization ability over ten two-class and four multi-class tasks comparing to the standard AdaBoost, CELS and NCCD. After each iteration, the easier examples are chosen to help the classification over the more difficult examples. It outperforms the standard AdaBoost and NCCD in more than half of the cases with a lower generalization error, and has competitive results with CELS. Besides, it seems CELS is less effective on multi-class cases than on two-class ones. As to the computation cost, our algorithm is much faster than CELS and NCCD, and relaxes the restriction on using neural network as the base learner. Therefore, AdaBoost.NC is a promising algorithm.

There are some remaining issues for future discussion. The ensemble size and the scaling coefficient λ are predefined at the moment. It is useful to know how our algorithm performs when an ensemble becomes very large or very small, and discuss the range of λ. Another interesting issue is to explore what kind of problems AdaBoost.NC is more effective on, and compare with recent classification ensembles by employing other metrics in addition to the misclassification error. Finally, more experiments need to be done for multi-class problems.

ACKNOWLEDGMENT

This work was partially supported by an Overseas Research Student Award (ORSAS) and a scholarship from the School of Computer Science, University of Birmingham, UK.

REFERENCES

[1] Y. Liu and X. Yao, "Simultaneous training of negatively correlated neural networks in an ensemble," *IEEE Transactions on Systems, Man and Cybernetics, PartB: Cybernetics*, vol. 29, no. 6, pp. 716–725, 1999.

[2] Z. S. H. Chan and N. Kasabov, "A preliminary study on negative correlation learning via correlation-corrected data," *Neural Processing Letters*, vol. 21, no. 3, pp. 207–214, 2005.

[3] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation, and active learning," in *Advances in Neural Information Processing Systems*, vol. 7, 1995, pp. 231–238.

[4] M. P. Perrone and L. N. Cooper, "When networks disagree: Ensemble methods for hybrid neural networks," *Artificial Neural Networks for Speech and Vision*, pp. 126–142, 1993.

[5] N. Ueda and R. Nakano, "Generalization error of ensemble estimators," *IEEE International Conference on Neural Networks*, vol. 1, pp. 90–95, 1996.

[6] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[7] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. of the 13th. Int. Conf. on Machine Learning*, 1996, pp. 148–156.

[8] G. Brown, J. L. Wyatt, and P. Tino, "Managing diversity in regression ensembles," *Journal of Machine Learning Research*, vol. 6, pp. 1621 – 1650, 2005.

[9] H. Chen, "Diversity and regularization in neural network ensembles (chapter 3)," Ph.D. dissertation, School of Computer Science, University of Birmingham, 2008.

[10] J. Quinlan, "Bagging, boosting, and c4.5," in *The 1996 13th National Conference on Artificial Intelligence, AAAI 96*, 1996, pp. 725–730.

[11] D. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *Journal of Artificial Intelligence Research*, vol. 11, pp. 169–198, 1999.

[12] Z. Zheng, "Multistrategy ensemble learning: Reducing error by combining ensemble learning techniques," *IEEE Trans. on Knowl. and Data Eng.*, vol. 16, no. 8, pp. 980–991, 2004.

[13] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, pp. 119–139, 1997.

[14] C. A. Shipp and L. I. Kuncheva, "An investigation into how adaboost affects classier diversity," in *IPMU*, 2002, pp. 203–208.

[15] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241 – 259, 1992.

[16] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.

[17] J. Zhu, S. Rosset, H. Zou, and T. Hastie, "Multi-class adaboost," 2006.

[18] J. Huang, S. Ertekin, Y. Song, H. Zha, and C. L. Giles, "Efficient multiclass boosting classification with active learning," in *The 7th SIAM International Conference on Data Mining*, 2007.

[19] R. Jin and J. Zhang, "Multi-class learning by smoothed boosting," *Machine Learning*, vol. 67, no. 3, pp. 207–227, 2007.

[20] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "Smote-boost: Improving prediction of the minority class in boosting," in *Knowledge Discovery in Databases: PKDD 2003*, vol. 2838/2003, 2003, pp. 107–119.

[21] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class imbalance learning," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 39, no. 2, pp. 539–550, 2009.

[22] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Machine Learning*, vol. 51, pp. 181–207, 2003.

[23] E. K. Tang, P. N. Suganthan, and X.Yao, "An analysis of diversity measures," *Machine Learning*, vol. 65, pp. 247–271, 2006.

[24] L. Kuncheva, C. Whitaker, C. Shipp, and R. Duin, "Limits on the majority vote accuracy in classifier fusion," *Pattern Analysis and Applications*, vol. 6, no. 1, pp. 22–31, 2003.

[25] G. Brown, J. Wyatt, R. Harris, and X. Yao, "Diversity creation methods: a survey and categorisation," *Information Fusion*, vol. 6, no. 1, pp. 5–20, 2004.

[26] H. Chen and X. Yao, "Regularized negative correlation learning for neural network ensembles," *IEEE Transactions on Neural Networks*, vol. 20, no. 12, pp. 1962–1979, 2009.

[27] ——, "Multi-objective neural network ensembles based on regularized negative correlation learning," *IEEE Transactions on Knowledge and Data Engineering*, 2010.

[28] L. Kuncheva and C. Whitaker, "Ten measures of diversity in classifier ensembles: limits for two classifiers," in *A DERA/IEE Workshop on Intelligent Sensor Processing (Ref. No. 2001/050)*, 2001, pp. 1–10.

[29] L. I. Kuncheva, "That elusive diversity in classifier ensembles," *Pattern Recognition and Image Analysis*, vol. 2652, pp. 1126–1138, 2003.

[30] Y. Sun, M. S. Kamel, A. K. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognition*, vol. 40, no. 12, pp. 3358–3378, 2007.

[31] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning: data mining, inference and prediction*. Springer Verlag, 2001.

[32] A. Asuncion and D. Newman, "UCI machine learning repository," 2007.

[33] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[34] C. X. Ling, J. Huang, and H. Zhang, "Auc: a statistically consistent and more discriminating measure than accuracy," in *In Proceedings of 18th International Conference on Artificial Intelligence (IJCAI2003)*, 2003, pp. 329–341.