Using Multiple Measures to Predict Confidence in Instance Classification

Kristine Monteith Tony Martinez

Abstract— **Selecting an effective method for combining the votes of classifiers in an ensemble can have a significant impact on the ensemble's overall classification accuracy. Some methods cannot even achieve as high a classification accuracy as the most accurate individual classifying component. To address this issue, we present the strategy of Aggregate Confidence Ensembles, which uses multiple measures to estimate a classifier's confidence in its predictions on an instance-by-instance basis. Using these confidence estimators to weight the votes in an ensemble results in an overall average increase in classification accuracy compared to the most accurate classifier in the ensemble. These aggregate measures result in higher classification accuracy than using a collection of single confidence estimates. Aggregate Confidence Ensembles outperform three baseline ensemble creation strategies, as well as the methods of Modified Stacking and Arbitration, both in terms of average classification accuracy and algorithm-by-algorithm comparisons in accuracy over 36 data sets.**

I. INTRODUCTION

Much attention has been directed towards determining how classifiers should be selected for use in an ensemble. One approach is to emphasize different portions of the data set during training, the strategy employed in Bagging [1]. Another strategy is to generate diversity not from the training sets but from the classifiers themselves. A number of techniques have been proposed to measure the diversity between classifiers [2], [3], and researchers have discussed correlations between classifier diversity in an ensemble and the accuracy that the ensemble is able to achieve [4], [5]. Efforts have been directed towards developing search strategies to dynamically discover a set of classifiers for use with a given task [6], [7]. Some research has even focused on discovering which classifiers and sets of classifiers are most accurate over the set of all "interesting" classification tasks. For example, in a large scale empirical study, [8] look at the behavior of a number of individual classifiers and ensembles on tens of thousands of data sets to determine which demonstrate the best overall performance.

However, even if an optimal set of classifiers could be identified for a given task, there remains the question of how to combine the information provided by these individual classifiers. Ideally, component classifiers specialize in different areas of the feature space, and the effectiveness of an ensemble can be enhanced if the votes are combined in a way that allows the overall ensemble to leverage these areas of expertise. This work focuses on optimizing the method of combining the votes of these classifiers to increase the overall accuracy of the ensemble.

The simplest method of combining the information presented in an ensemble is to allow each classifier to have one vote toward the final classification of an instance. A number of ensemble techniques such as Bagging [1] employ this strategy. With Boosting [9], [10], training data is weighted differently for each classifier, and votes are weighted by the accuracy that a given classifier achieves on the data set on which it was trained. More complicated ensemble-combining strategies include the Adaptive Mixture of Local Experts strategy [11], where a gating network determines the probability of selecting the output of one of the base level classifiers, and Stacking [12] which makes use of a meta-level learning algorithm that discovers the best way to combine outputs from the base level classifiers. Arbitration [13] creates a "referee" to determine the confidence that a learning model has in its classification of the various subdomains of a given problem. Information about both the misclassification of instances and the classifiers themselves are used in the development of the meta-learner referees.

These ensemble construction methods take advantage of the fact that individual classifiers generally perform better on some portions of the feature space than others; higher ensemble accuracy can often be obtained by taking areas of specialization into account when weighting ensemble votes. For example, Delegating [14] is an approach where a classifier assigns a class label to a given instance only if it has high confidence in that particular class. If it is less confident, the instance is delegated to another learner. With a technique called Dynamic Selection [15], information is collected on how well learners perform on instances in the training set. These learners are then used to classify test set instances, and their collective predictions are used to determine similarity to different instances in the training set. The learner that achieves the best performance on that area of the training set is then used to classify the test set instance.

Bayesian averaging is a technique that weights the predictions of each classifier in an ensemble by the probability that each model accurately represents the data [16]. In theory, this strategy could avoid pitfalls of overfit and achieve optimal classification accuracy. However, Domingos [17] found that Bayesian averaging was actually more prone to overfit than other more ad hoc methods because in practice, it so heavily favors the maximum likelihood option.

Dzeroski and Zenko [18] found that the accuracy of an

Kristine Monteith is with the Department of Computer Science, Brigham Young University, Provo, UT 84602, United States (phone: 801-422-1660; email: kristinemonteith@gmail.com).

Tony Martinez is with the Department of Computer Science, Brigham Young University, Provo, UT 84602, United States (phone: 801-422-6464; email: martinez@cs.byu.edu).

ensemble over a data set is often no better than the accuracy of one of the classifiers contributing to the ensemble. In order to justify the overhead of creating an ensemble, the ensemble should meet the criterion of having a higher overall classification accuracy than any of its component classifiers. In the algorithms Dzeroski and Zenko explore, only their Modified Stacking strategy was able to consistently achieve this level of accuracy.

This work presents the new strategy of Aggregate Confidence Ensembles. With this technique, the votes of classifiers are weighted by confidence as determined on an instance-by-instance basis. Each classifier is trained using a different algorithm on the same training set data. Then each instance in the test set is assigned a class value and an overall confidence rating for that classification by each of n classifiers. Multiple factors are taken into consideration when determining this overall confidence rating. For example, six different confidence estimators are used to calculate confidence in the prediction of a decision tree classifier. A given instance would receive six confidence ratings, reflecting properties such as the purity of the leaf node in which it was classified and the number of instances classified at that leaf. These six numbers are then aggregated to produce an overall confidence rating for the decision tree's classification of this particular instance. A similar method is used to calculate an overall confidence rating for each of the classifiers. The class label assigned to the instance is then calculated by summing the weights for each possible label and selecting the class label with the maximum total.

The technique of Aggregate Confidence Ensembles is shown to achieve higher average classification accuracy over 36 data sets than the standard combination strategies employed by Bagging and Boosting as well as the SelectBest strategy of allowing the most accurate classifier in the ensemble to make all the classifications. It also outperforms Arbitration [13] and the Stacking algorithm presented by Dzeroski and Zenko [18] both in terms of average classification accuracy and win/loss ratios.

Section two of this work gives an overview of the Aggregate Confidence Ensembles algorithm. Section three presents confidence estimators for five common classification algorithms. Section four provides results comparing Aggregate Confidence Ensembles with standard voting, voting by accuracy, the SelectBest strategy, Arbitration, and Modified Stacking. Section five outlines conclusions and suggests options for further research.

II. Aggregate Confidence Ensembles

Let $C_1 ... C_n$ be classifiers constructed using instances from a training set A. Each classifier C_i has m pre-defined confidence estimators. For a given instance k in the training set, a vector $\mathbf{h}_i^k$ of m confidence values is calculated, with h_{ij}^k representing the confidence that estimator j has in in classifier C_i's classification of instance k.

For each classifier C_i, let $\hat{\mathbf{y}}_i$ be the predictions of C_i over the training set A, with $\hat{y}_i^k$ being the class label assigned by classifier C_i when instance k appeared in a test fold during cross-validation on the training set. Let $\mathbf{y}$ be a vector of the target values for the training instances, and $\mathbf{z}_i$ be a vector describing $\hat{\mathbf{y}}_i = \mathbf{y}$. In other words, if $\hat{y}_i^k = y^k$, then z_i^k = 1; if not, z_i^k = 0. For each classifier C_i, let $\mathbf{r}_i$ be a vector of correlation values, where r_{ij} is the correlation between $\mathbf{z}_i$ and $\mathbf{h}_{ij}$. The values in $\mathbf{r}_i$ are then scaled to sum to one.

For each unlabeled instance x, let $\hat{y}_i^x$ be the class label assigned to instance x by classifier C_i. Let $\mathbf{h}_i^x$ be a vector of confidence values calculated for the classification of instance x by classifier C_i. These values will be used in determining how much weight the overall ensemble should assign to the classification $\hat{y}_i^x$. In order to make the values assigned by the various estimators more uniform among the classifiers, h_{ij}^x is normalized using the maximum and minimum values from the vector $\mathbf{h}_{ij}$ of values calculated for training set instances.

Let w_i^x be the dot product of $\mathbf{h}_i^x$ and $\mathbf{r}_i$. This aggregate measure is then used to weight $\hat{y}_i^x$. The class label assigned to $\mathbf{x}$ by the overall ensemble is calculated by summing the weights for each possible label and selecting the class label with the maximum total. The strategy of Aggregate Confidence Ensembles is outlined in Figure 1.

```
1.Train each of $n$ classifiers $C_1 ... C_n$ using training set $A$.
   A. Determine the following for each classifier $C_i$:
      1. For each instance $k$ in $A$:
         a. Calculate vector $\mathbf{h}_i^k$ of confidence values
            using $m$ estimators specific to $C_i$
         b. Calculate $\hat{y}_i^k$, the prediction of class label
            by $C_i$ when $k$ appeared in a test fold
            during cross-validation on $A$
         c. Identify $y^k$, the target value for instance $k$
      2. Define $\mathbf{z}_i$ to be a vector describing $\hat{\mathbf{y}}_i = \mathbf{y}$
      3. Calculate vector $\mathbf{r}_i$ of correlation values
         where $r_{ij}$ is the correlation between $\mathbf{z}_{ij}$ and $\mathbf{h}_{ij}$
2. For an unlabeled instance $x$:
   A. For each classifier $C_i$:
      1. Determine $\hat{y}_i^x$, the class value of $\mathbf{x}$ as predicted
         by $C_i$
      2. Create vector $\mathbf{h}_i^x$ of confidence values using
         $m$ estimators specific to $C_i$
      3. $w_i^x$ = $\mathbf{h}_i^x \cdot \mathbf{r}_i$
   B. Class value for $\mathbf{x}$ = argmax$_{y \in Y}(\Sigma_{i=1}^n \delta(y, \hat{y}_i^x) w_i^x)$
      $\delta(y, \hat{y}_i) = \begin{cases} 1 \text{ when } \hat{y}_i = y \\ 0 \text{ otherwise} \end{cases}$
```

Fig. 1. Aggregate Confidence Ensembles

III. Multiple Confidence Estimators

This section contains the information about the confidence estimators used to predict confidence in classifications for each of five different algorithms. The five algorithms used in this work were selected because they are representative of standard classes of algorithms used in machine learning.

Many of the confidence estimators presented here could be adapted for use with similar machine learning algorithms. The algorithms used in this work are implemented using Weka open source code [19]. Default settings are used for each of the algorithms.

While we have tried to select diverse models to represent the spectrum of machine learning algorithms, the technique of Aggregate Confidence Ensembles could be applied to ensembles with any number and type of base-level classifiers. The ensembles of the five base-level classifiers discussed here are designed simply to present the concept. One classifier of each type is used for parsimony and to avoid skewing the ensemble in favor of any particular classifier.

Efficacy of the various confidence estimators is evaluated using 36 data sets taken from the UCI Repository [20]. Table I provides information about these data sets. Data sets were selected so as to achieve variety in number of instances, attributes, attribute types, and output classes. The data sets range from 90 to 2310 instances, 5 to 70 attributes, and 2 to 24 output classes. Roughly a third of the data sets feature discrete attributes, another third have real-valued attributes, and the remaining data sets have a mixture of real-valued and discrete attributes. Ten of the data sets contain missing values. In the case of discrete attributes, unknown values were replaced by the most common value for the given attribute. For data sets with real-valued attributes, unknown values were replaced with the average value for the attribute.

TABLE I
LIST OF DATA SETS

anneal	audiology	autos	balance-scale
bupa	cancer	car	cmc
colic	credit-a	credit-g	dermatology
diabetes	ecoli-c	glass	haberman
heart	heart-cleveland	heart-statlog	hepatitis
ionosphere	iris	lymph	monks
post-op	primary-tumor	sonar	spect
tae	tic-tac-toe	vehicle	vote
wine	yeast	yugoslavia-cancer	zoo

Each subsection contains information about the algorithm to be addressed and the confidence estimators specific to that algorithm. The subsections also contain graphs providing information about the behavior of each confidence estimator on the data sets shown in Table I. Each of the classifiers was evaluated over each data sets using ten-fold cross validation, and instances were marked as correctly or incorrectly classified based on the classifier's ability to classify the instance when it appeared in the test set. This correctness of classification is then compared to the confidence measure assigned to each instance by each of the confidence estimators.

As an example, Figure 2 shows a graph constructed for the confidence estimator measuring purity of classification at the leaf node of a decision tree. The graph shows the number of instances receiving a given confidence value that were correctly and incorrectly classified. While the purity confidence estimator provides real values, for clarity in graphing, confidences are grouped in discrete bins (e.g. confidence values from 0.5 to 0.59 are all graphed as 0.5). The real-valued, unbinned confidence estimates are used in the actual classification experiments. The bar on the left for each bin represents the number of instances receiving this confidence value that were correctly classified. The bar on the right represents the number of instances that were incorrectly classified. For example, the far right-hand bin in Figure 2 shows that, out of all 36 data sets, 5128 instances receiving a confidence value of 1.0 from this confidence estimator were correctly classified, and 863 instances receiving this confidence value were incorrectly classified.

For each of the estimators studied, higher confidence values generally corresponded with a higher percentage of correctly classified instances. More specifically, instances assigned the highest confidence measure were more likely to be correctly classified than instances assigned the lowest confidence measure for each of the estimators studied. However, in each case, the aggregate confidence estimator was significantly more correlated with correctness of classification than each of the individual estimators. Each subsection contains a table outlining the correlation between the various confidence estimators and their correlation with correctness in test set classification.

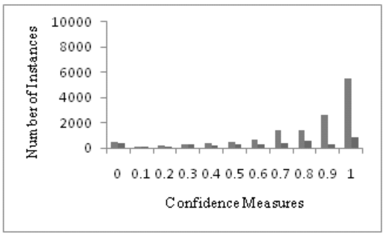
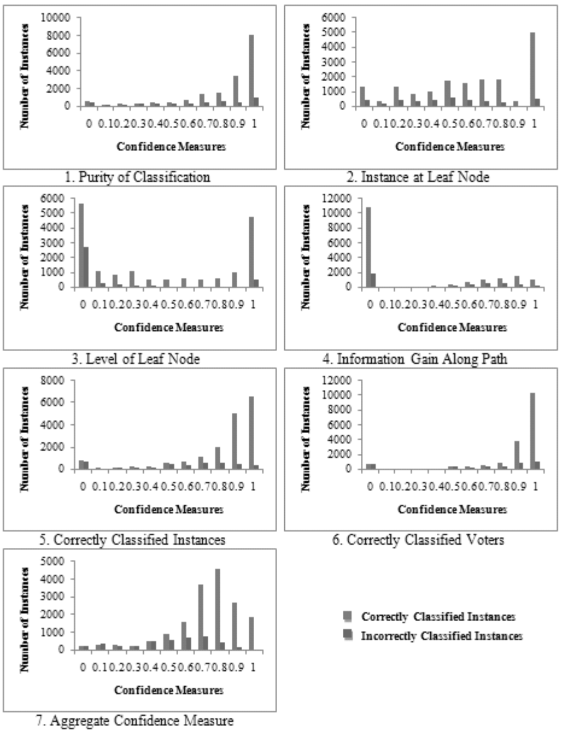


Fig. 2. Confidence Estimator Treatment of Correct and Incorrect Instances

A. *Decision Tree - J48*

The J48 algorithm is the Weka implementation of the C4.5 algorithm [21], an extension of the ID3 decision tree [22]. Six different confidence estimators are used to predict confidence in this algorithm's classification of a given instance. These include the following: 1) The number of instances with the majority classification at the leaf node when the given instance is classified (the purity of classification at that node); 2) The number of instances at the leaf node; 3) The level of the tree at which the given instance is classified; 4) The average of the information gain statistics along the classification path (normalized by maximum possible information gain for a given data set); 5) The percentage of instances at the leaf node that were correctly classified in hold-one-out cross-validation experiments on the training set; and 6) The percentage of instances at the leaf nodes with the majority classification for that node that are correctly classified in hold-one-out cross validation on the training set. Please note that in each case where cross validation is conducted on a training set to determine confidence, the training set is considered to be the training set for a given fold of the overall cross-validation experiments used to predict accuracy of the algorithm.[1]



1. Purity of Classification

2. Instance at Leaf Node

3. Level of Leaf Node

4. Information Gain Along Path

5. Correctly Classified Instances

6. Correctly Classified Voters

7. Aggregate Confidence Measure

Fig. 3. Decision Tree Confidence Estimators Treatment of Correct and Incorrect Instances

TABLE II
DECISION TREE CONFIDENCE ESTIMATORS AND CORRELATION WITH CORRECTLY CLASSIFIED INSTANCES

Confidence Estimator	Correlation
1. Purity of Classification	0.219
2. Instances at Leaf Node	0.167
3. Level of Leaf Node	0.199
4. Information Gain Along Path	-0.072
5. Correctly Classified Instances	0.280
6. Correctly Classified Voters	0.248
Aggregate Confidence Estimator	**0.292**

The first confidence estimator is a standard method for predicting confidence in the classification of a decision tree [19]. The second and third provide an effective complement to the first by providing information about the amount of overfit and thus how much the first should be trusted. The fourth confidence estimator provides information about how effectively a given attribute is able to split the data at each level of the decision tree, assuming that strong attributes will lead to more confident classifications. The fifth identifies how effective the classifier is at classifying the instances in this particular section of the data. The sixth confidence estimator provides information about how effectively the classifier was able to classify the instances specifically contributing to the classification of the given instance. Figure 3 shows the behavior of these confidence measures on instances in the thirty-six data sets studied. Table II shows how values assigned by these confidence measures correlate with correctness of classification.

B. Rule-Based Classifier - Decision Table

These experiments use one of Weka's rule-based classifiers called a Decision Table [23]. This algorithm selects a set of attributes to be used in determining classification, and produces a classification for each combination of observed values for these attributes. The following attributes are taken into consideration when trying to predict confidence in this algorithm's classification of a given instance: 1) The number of instances with the majority classification covered by the rule; 2) The number of antecedents in the rule; 3) The number of instances covered by the rule; 4) The percentage of instances covered by the rule that were correctly classified in hold-one-out cross-validation experiments on the training set; 5) The percentage of instances covered by the rule with the majority classification for that rule that were correctly classified in hold-one-out cross-validation on the training set; and 6) Whether or not this instance is covered by a rule.

The rationale for these confidence estimators is similar to the rationale for the decision tree confidence estimators. The first is a standard measure of confidence. The second and third assess the probability of overfit or underfit. The fourth and fifth measure the effectiveness and strength of classification. They indicate how effectively the decision table was able to classify instances that would end up in this region and how effectively the most pertinent instances in this region can be classified. The sixth confidence estimator indicates whether or not a rule was found in the table that covered the given instance to be classified. Figure 4 shows the behavior of these confidence measures on data set instances. Table III shows how values assigned by these confidence measures correlate with correctness of classification.

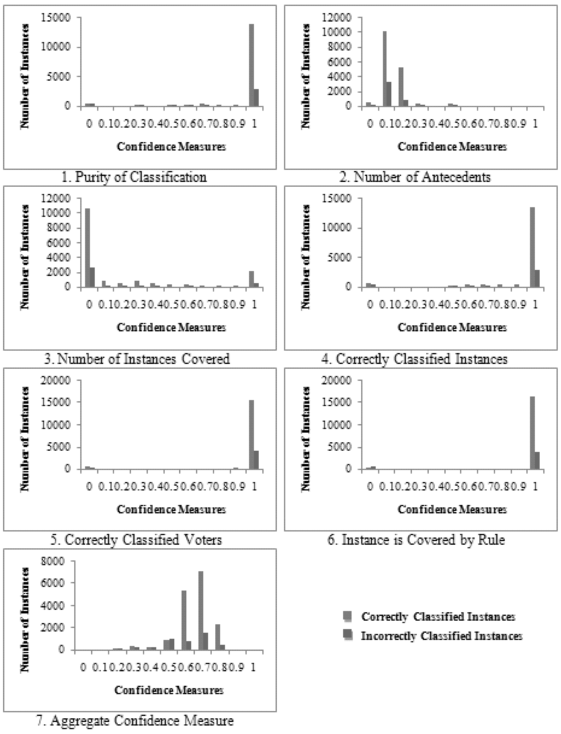
TABLE III
RULE-BASED CLASSIFIER CONFIDENCE ESTIMATORS AND CORRELATION WITH CORRECTLY CLASSIFIED INSTANCES

Confidence Estimator	Correlation
1. Purity of Classification	0.147
2. Number of Antecedents	-0.004
3. Number of Instances Covered	0.110
4. Correctly Classified Instances	0.139
5. Correctly Classified Voters	0.102
6. Instance is Covered by Rule	0.217
Aggregate Confidence Estimator	**0.240**

C. Instance-Based Classifier

With the instance-based k-nearest-neighbor algorithm, an instance is classified based on the classifications of the k

[1] For a more rigorous presentation of these and all other confidence estimators mentioned in this paper, please see http://axon.cs.byu.edu/ACE.

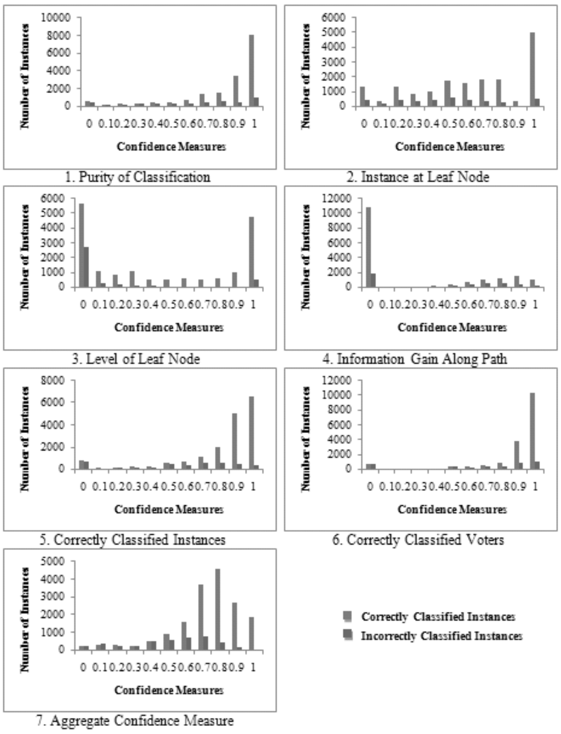


1. Purity of Classification
2. Number of Antecedents
3. Number of Instances Covered
4. Correctly Classified Instances
5. Correctly Classified Voters
6. Instance is Covered by Rule
7. Aggregate Confidence Measure

Fig. 4. Rule-Based Classifier Confidence Estimators Treatment of Correct and Incorrect Instances

instances nearest that instance [24]. These experiments use the 5-nearest-neighbor version of the algorithm. Six different options are used to predict confidence in this algorithm's classification of a given instance: 1) The percentage of the first five neighbors that have the same classification as the majority classification for those five neighbors; 2) The difference between the distance-weighted vote of the predicted class and the distance-weighted vote of the next highest class; 3) The average distance from this instance to its first five neighbors (normalized and subtracted from one); 4) The percentage of the first five neighbors that were correctly classified in hold-one-out cross-validation on the training set; 5) The percentage of neighbors with the majority classification that were correctly classified in hold-one-out cross-validation on the training set; and 6) A comparison of 3-NN, 5-NN and 7-NN classifications of a given instance.

The first and second confidence estimators indicate the general confidence in a classification, and how confident that classification is relative to other possible classifications. The third measures how close the neighbors are to the individual instance, making the assumption that a point closer to other points is more likely to be correctly classified. The fourth and fifth confidence estimators measure the classification accuracy of instances in this region and the accuracy on instances contributing to the classification of the instance in question. The last confidence estimator indicates the effectiveness of using this particular number of neighbors to classify the given instance. Figure 5 shows the behavior of these confidence measures on data set instances. Table IV reports correlation between values assigned by these measures and correctness of classification.



1. Purity of Classification
2. Instance at Leaf Node
3. Level of Leaf Node
4. Information Gain Along Path
5. Correctly Classified Instances
6. Correctly Classified Voters
7. Aggregate Confidence Measure

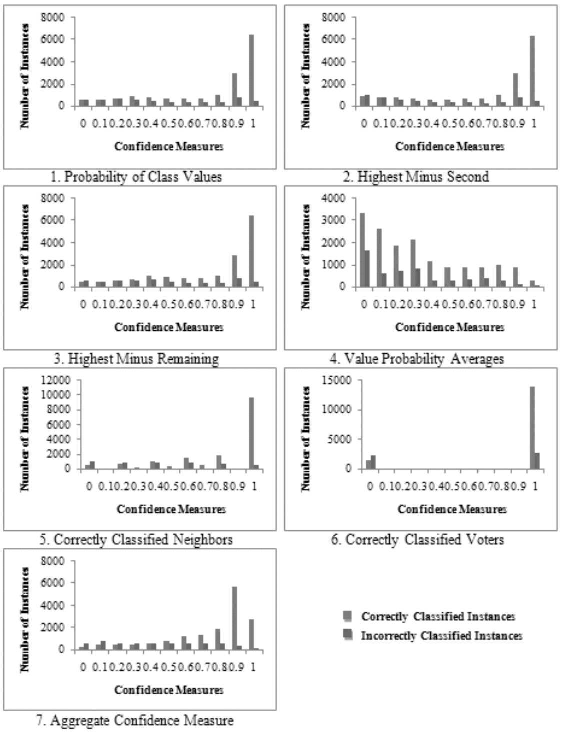
Fig. 5. Instance-Based Classifier Confidence Estimators Treatment of Correct and Incorrect Instances

TABLE IV
INSTANCE-BASED CLASSIFIER CONFIDENCE ESTIMATORS AND CORRELATION WITH CORRECTLY CLASSIFIED INSTANCES

Confidence Estimator	Correlation
1. Neighbors in Agreement	0.325
2. Highest Minus Second	0.343
3. Average Distance to Neighbors	0.114
4. Correctly Classified Neighbors	0.276
5. Correctly Classified Voters	0.198
6. 3-NN vs. 5-NN vs. 7-NN	0.242
Aggregate Confidence Estimator	**0.358**

D. Naïve Bayes Classifier

The Naïve Bayes classifier uses Bayesian logic to predict class values for each instance based on the probabilities of the attribute values for that instance [25] [26]. The following are considered when trying to predict confidence in classification of a given instance by the Naïve Bayes classifier: 1) Probability of the class value predicted by the Naïve Bayes classifier; 2) The distance between the predicted probability and the probability of the second most likely class



1. Probability of Class Values

2. Highest Minus Second

3. Highest Minus Remaining

4. Value Probability Averages

5. Correctly Classified Neighbors

6. Correctly Classified Voters

7. Aggregate Confidence Measure

Fig. 6. Naïve Bayes Classifier Confidence Estimators Treatment of Correct and Incorrect Instances

TABLE V
NAÏVE BAYES CLASSIFIER CONFIDENCE ESTIMATORS AND CORRELATION WITH CORRECTLY CLASSIFIED INSTANCES

Confidence Estimator	Correlation
1. Probability of Class Value	0.303
2. Highest Minus Second	0.298
3. Highest Minus Remaining	0.303
4. Value Probability Averages	0.075
5. Correctly Classified Neighbors	0.371
6. Correctly Classified Voters	0.306
Aggregate Confidence Estimator	**0.394**

value for the instance; 3) The distance between the predicted probability and the sum of the probabilities for the remaining class values; 4) The average probability across the data set of each attribute value in the instance; 5) The percentage of the five neighbors nearest in probability that were correctly classified in hold-one-out cross-validation on the training set; and 6) The percentage of the nearest five neighbors with the same class value as this instance that were correctly classified in hold-one-out cross-validation on the training set.

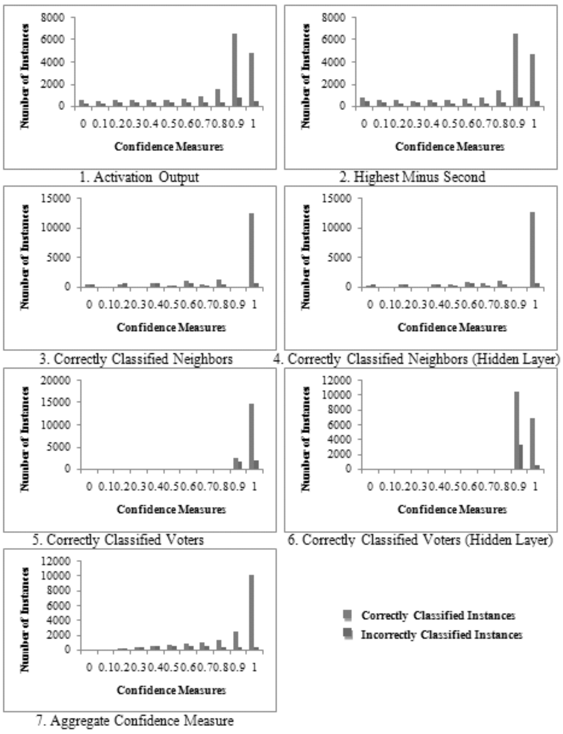
The first confidence estimator is used because it is the standard way of predicting the confidence of a Naïve Bayes classifier. The second and third confidence estimators are attempts to gain more information about how confident the classifier is in its ordering. The fourth confidence estimator addresses the fact that attribute values with lower representation in a data set may be less effective at contributing to a correct classification. The fifth confidence estimator is aimed at determining how confident the classifier is in this region of the input space. With this confidence estimator, the output probabilities of all the instances in the training data are taken into consideration. The five instances with output probabilities closest to those of the instance in question are then located, and the confidence estimate is calculated by observing the percentage of these five instances that were correctly classified in hold-one-out cross-validation on the training set. The sixth confidence estimator focuses specifically on neighbors with the same classification as the given instance. Figure 6 shows the confidence measures assigned to correct and incorrect instances. Table V shows how values assigned by these confidence measures correlate with correctness of classification.

E. Multilayer Perceptron trained with Backpropagation

One of the most common methods of training a multilayer perceptron, backpropagation incrementally changes the weights between nodes when these weights are responsible for the misclassification of instances during training [27]. The following are considered in trying to predict confidence in classification by the Multilayer Perceptron: 1) The activation output for the selected classification; 2) The difference between the highest and second highest activation outputs; 3) The percentage of the five neighbors nearest in activation output that were correctly classified in hold-one-out cross-validation on the training set; 4) The percentage of the five neighbors nearest in activation output of the hidden layer that were correctly classified in hold-one-out cross-validation on the training set; 5) The average difference in activation output between the selected classification and its five nearest neighbors compared to the average of this statistic computed for all instances; and 6) The average difference in hidden-layer activation output between the selected classification and its five nearest neighbors compared to the average of this statistic computed for all instances.

The first and second confidence estimators provide information about the confidence of a given classification and confidence relative to other possible classifications. The third and fourth provide information about how confident the learner is on this region of the input space. These confidence estimators are calculated in a similar manner to the fifth confidence estimator used for the Naïve Bayes algorithm: all the instances in the training set are considered, and the five with output vector most similar to the instance in question are then used to calculate the confidence estimator. The third confidence estimator uses the outputs from the standard output nodes to identify the nearest neighbors. The fourth confidence estimator uses the outputs from the hidden nodes. The fifth and sixth confidence estimators provide information about how similar a given instance is to previously seen instances, based on the assumption that the classifier will be more effective at predicting a class value for an instance similar to one that it has seen before. Figure 7 shows the behavior of these confidence measures on data set instances.

Table VI reports how values assigned by these confidence measures correlate with correctness of classification.



1. Activation Output

2. Highest Minus Second

3. Correctly Classified Neighbors

4. Correctly Classified Neighbors (Hidden Layer)

5. Correctly Classified Voters

6. Correctly Classified Voters (Hidden Layer)

7. Aggregate Confidence Measure

Fig. 7. Multilayer Perceptron Confidence Estimators Treatment of Correct and Incorrect Instances

TABLE VI

MULTILAYER PERCEPTRON CONFIDENCE ESTIMATORS AND CORRELATION WITH CORRECTLY CLASSIFIED INSTANCES

Confidence Estimator	Correlation
1. Activation Output	0.053
2. Highest Minus Second	0.051
3. Correctly Classified Neighbors	0.295
4. Correctly Classified Neighbors (Hidden Layer)	0.266
5. Average Distance to Neighbors	0.239
6. Average Distance to Neighbors (Hidden Layer)	0.157
Aggregate Confidence Estimator	**0.310**

IV. RESULTS AND DISCUSSION

In this section, the strategy of Aggregate Confidence Ensembles is compared with a number of different ensemble combining strategies using average accuracy, win/loss/tie ratios, and overall wins in classification accuracy.

A. Results

The first baseline comparison method is a standard voting strategy where each classifier in an ensemble votes on the classification of an instance and the votes are weighted equally. The second baseline method weights the votes by the overall accuracy of the learner on the training set. The third baseline method, identified here as the SelectBest method, chooses the classifier in the ensemble that achieved the highest accuracy on the training data and uses that classifier alone on the test data.

The strategy of Aggregate Confidence Ensembles is also compared to the method of Stacking found to be most effective by Dzeroski and Zenko [18]. In this method, identified as Modified Stacking in the following analyses, the output probabilities of each of the component classifiers are given as input to a set of model trees. Each tree is designed to make a binary decision about a given possible output class, and the ensemble assigns a value to the instance according to which model tree has the highest positive confidence in its prediction. Table VII shows the results of these comparisons. Using the Wilcoxon signed-rank test to determine significance [28], Aggregate Confidence Ensembles achieved a significantly higher average accuracy than any of the other strategies at a confidence level of 99% (p-values<=0.001,0.003,0.014,0.003,0.008).

TABLE VII

ALGORITHM AVERAGE ACCURACY OVER 36 DATASETS

Standard Voting	83.65
Weight by Accuracy	83.76
SelectBest	83.48
Arbitration	83.58
Modified Stacking	83.20
Aggregate Confidence Ensembles	**84.32**

Aggregate Confidence Ensembles also performed well in an algorithm-by-algorithm comparison of accuracy on the thirty-six individual data sets. Table VIII shows an algorithm-by-algorithm comparison of each of the five strategies. Each box shows the number of wins, losses, and ties in accuracy on each of the 36 data sets when comparing the algorithm in the given row with the algorithm in the given column. For example, when compared to the standard voting strategy, Aggregate Confidence Ensembles achieved a higher classification accuracy on twenty-four of the data sets, a lower classification accuracy on nine of the data sets, and the same classification accuracy on three data sets. Aggregate Confidence Ensembles achieved higher classification accuracy on a majority of the data sets studied when compared to Standard Voting, Weighting by Accuracy, the SelectBest method, Arbitration, and Modified Stacking.

B. Discussion

Additional experiments were conducted to investigate the accuracy of ensembles using single confidence estimators. Many of the single estimator ensembles were able to achieve a higher average classification accuracy than a baseline strategy of standard voting. However, the use of these single values was not sufficient to create an ensemble that produced a higher average predictive accuracy on a level that was statistically significant, so the use of additional confidence

TABLE VIII

Algorithm Comparision using Win/Loss/Tie Ratios

	Standard Voting	Weight by Accuracy	SelectBest	Arbitration	Modified Stacking
W. Accuracy	9/3/24				
SelectBest	16/17/3	15/19/2			
Arbitration	14/15/7	12/17/7	16/18/2		
M. Stacking	15/19/2	14/21/1	18/17/1	15/19/2	
ACE	**24/9/3**	**22/8/6**	**23/10/3**	**27/8/1**	**24/9/3**

estimators was warranted.[2]

The higher average accuracy of Aggregate Confidence Ensembles does come with a higher cost of computation, but for two-thirds of the heuristics the increase in computational complexity is only linear in regard to the size of the data set. The other one-third of the heuristics requires cross-validation on the training set. The computational complexity for these heuristics could be reduced by reducing the number of folds used in these training set calculations.

V. Conclusion and Future Work

This work presents a new method of combining the votes in an ensemble using multiple estimators to predict confidence in the classification of a given instance. A number of estimators designed for this task are proposed for each of five different types of classifiers. Combinations of these measures are shown to be more highly correlated with correctness of classification than any of the individual measures. The strategy of Aggregate Confidence Ensembles, which employs all of the confidence estimators presented, is shown to achieve a higher average classification accuracy over 36 data sets than five alternate ensemble strategies. It also compares favorably in an algorithm-by-algorithm comparison of wins and losses in accuracy over the data sets.

The confidence estimators presented in this work explore some of the strengths and weaknesses of a given classifier on a given data set. This information could result in the development of new algorithms. For example, a new instance-based classifier might be developed in which only instances that were correctly classified in hold-one-out cross-validation would be allowed to vote on the classification of an unseen instance. The probabilities output by a Naïve Bayes classifier might be altered slightly based on information gained through confidence estimators like the ones presented here. Insights gained by observing the behavior of the confidence estimators on various data sets may help target areas of data sets on which individual classifiers can improve their performance.

References

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

[2] R. Kohavi and D. Wolpert, "Bias plus variance decomposition for zero-one loss functions," *Proceedings of the Thirteenth International Conference on Machine Learning*, pp. 275–283, 1996.

[3] A. H. Peterson and T. R. Martinez, "Estimating the potential for combining learning models," *Proceedings of the ICML Workshop on Meta-Learning*, pp. 68–75, 2005.

[4] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles," *Machine Learning*, vol. 51, pp. 181–207, 2003.

[5] D. Ruta and B. Gabrys, "Analysis of the correlation between majority voting error and the diversity measures in multiple classifier systems," *Proceedings of the Fourth International Symposium on Soft Computing*, 2001.

[6] ——, "Classifier selection for majority voting," *Information Fusion*, vol. 6, no. 1, pp. 63–81, 2005.

[7] G. Giacinto and F. Roli, "A theoretical framework for dynamic classifier selection," *Proceedings of the Fifteenth International Conference on Pattern Recognition*, 2000.

[8] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," *Proceedings of the Twenty-third International Conference on Machine Learning*, pp. 161–168, 2006.

[9] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," *Proceedings of the Thirteenth International Conference on Machine Learning*, 1996.

[10] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pp. 80–91, 1998.

[11] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, pp. 79–87, 1991.

[12] D. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–260, 1992.

[13] J. Ortega, M. Koppel, and S. Argamon, "Arbitrating among competing classifiers using learned referees," *Knowledge and Information Systems Journal*, vol. 3, no. 4, pp. 470–490, 2001.

[14] C. Ferri, P. Flach, and J. Hernandez-Orallo, "Delegating classifiers," *Proceedings of the Twenty-first International Conference on Machine Learning*, pp. 289–296, 2004.

[15] C. J. Merz, "Dynamic learning bias selection," *Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics*, pp. 386–395, 1995.

[16] J. Hoeting, D. Madigan, A. Raftery, and C. Volinsky, "Bayesian model averaging: A tutorial," *Statistical Science*, vol. 14, no. 4, pp. 382–417, 1999.

[17] P. Domingos, "Bayesian averaging of classifiers and the overfitting problem," *Proceedings of the Seventeenth International Conference on Machine Learning*, pp. 223–230, 2000.

[18] S. Dzeroski and B. Zenko, "Is combining classifiers with stacking better than selecting the best one?" *Machine Learning*, vol. 54, pp. 255–273, 2004.

[19] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. San Francisco: Morgan Kaufmann, 2005.

[20] S. Hettich, C. L. Blake, and C. J. Merz, "Uci repository of machine learning databases," Irvine, CA: University of California, Department of Information and Computer Science, 1998. [Online]. Available: http://www.ics.uci.edu/ mlearn/MLRepository.html

[21] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufman, 1993.

[22] ——, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[23] R. Kohavi, "The power of decision tables," *Proceedings of the Eighth European Conference of Machine Learning*, 1995.

[24] T. M. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, pp. 21–27, 1967.

[25] K. Lang, "Newsweeder: Learning to filter netnews," *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 331–339, 1995.

[26] T. M. Mitchell, *Machine Learning*. WCB/McGraw-Hill, 1997.

[27] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. Vol 1: Foundations, pp. 318–362, 1986.

[28] J. Demsar, "Statistical comparison of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, p. 130, 2006.

[2] Please see http://axon.cs.byu.edu/ACE for more details on these experiments.