

Experiments on Ensembles with Missing and Noisy Data

Prem Melville, Nishit Shah, Lilyana Mihalkova, Raymond J. Mooney

Department of Computer Sciences
University of Texas at Austin, Austin TX 78712, USA

Abstract. One of the potential advantages of multiple classifier systems is an increased robustness to noise and other imperfections in data. Previous experiments on classification noise have shown that bagging is fairly robust but that boosting is quite sensitive. DECORATE is a recently introduced ensemble method that constructs diverse committees using artificial data. It has been shown to generally outperform both boosting and bagging when training data is limited. This paper compares the sensitivity of bagging, boosting, and DECORATE to three types of imperfect data: missing features, classification noise, and feature noise. For missing data, DECORATE is the most robust. For classification noise, bagging and DECORATE are both robust, with bagging being slightly better than DECORATE, while boosting is quite sensitive. For feature noise, all of the ensemble methods increase the resilience of the base classifier.

1 Introduction

In addition to their many other advantages, multiple-classifier systems hold the promise of developing learning methods that are robust in the presence of imperfections in the data; in terms of missing features, and noise in both the class labels and the features. Noisy training data tends to increase the variance in the results produced by a given classifier; however, by learning a committee of hypotheses and combining their decisions, this variance can be reduced. In particular, variance-reducing methods such as *Bagging* [2] have been shown to be robust in the presence of fairly high levels of noise, and can even *benefit* from low levels of noise [3].

Bagging is a fairly simple ensemble method which is generally outperformed by more sophisticated techniques such as ADABOOST [4,13]. However, ADABOOST has a tendency to overfit when there is significant noise in the training data, preventing it from learning an effective ensemble [3]. Therefore, there is a need for a general *ensemble meta-learner*¹ that is at least as accurate as ADABOOST when there is little or no noise, but is more robust to higher levels of random error in the training data.

DECORATE [9,10] is a recently introduced ensemble meta-learner that directly constructs diverse committees by employing specially-constructed artificial

¹ An ensemble meta-learner, like Bagging and ADABOOST, takes an arbitrary *base learner* and uses it to build a more effective committee of hypotheses [17].

training examples. Extensive experiments have demonstrated that DECORATE constructs more accurate diverse ensembles than ADABOOST and Bagging when training data is limited, and does at least as well as ADABOOST when the training set is relatively large. By using artificial training data to construct diverse committees and prevent over-fitting, DECORATE has been shown to be a very effective ensemble meta-learner on a wide variety of data sets.

This paper explores the resilience of DECORATE to the various forms of imperfections in data. In our experiments, the training data is corrupted with missing features, and random errors in the values of both the category and the features. Results on a variety of UCI data demonstrate that, in general, DECORATE continues to improve on the accuracy of the base learner, despite the presence of each of the three forms of imperfections. Furthermore, DECORATE is clearly more robust to missing features than the other ensemble methods.

2 The DECORATE Algorithm

This section summarizes the DECORATE algorithm; for further details see [9, 10]. The approach is motivated by the fact that combining the outputs of multiple classifiers is only useful if they disagree on some inputs [6]. We refer to the amount of disagreement as the *diversity* of the ensemble, which we measure as the probability that a random ensemble member’s prediction on a random example will disagree with the prediction of the complete ensemble.

DECORATE was designed to use additional artificially-generated training data in order to generate highly diverse ensembles. An ensemble is generated iteratively, learning one new classifier at each iteration and adding it to the current ensemble. The ensemble is initialized with the classifier trained on the given data. The classifiers in each successive iteration are trained on the original data and also on some artificial data. In each iteration, a specified number of artificial training examples are generated based on a simple model of the data distribution. The category labels for these artificially generated training examples are chosen so as to differ maximally from the current ensemble’s predictions. We refer to this artificial training set as the *diversity data*. We train a new classifier on the union of the original training data and the diversity data. If adding this new classifier to the current ensemble increases the ensemble training error, then this classifier is rejected, else it is added to the current ensemble. This process is repeated until the desired committee size is reached or a maximum number of iterations is exceeded.

The artificial data is constructed by randomly generating examples using an approximation of the training data distribution. For numeric attributes, a Gaussian distribution is determined by estimating the mean and standard deviation of the training set. For nominal attributes, the probability of occurrence of each distinct value is determined using Laplace estimates from the training data. Examples are then generated by randomly picking values for each feature based on these distributions, assuming attribute independence. In each iteration, the artificially generated examples are labeled based on the current ensemble. Given an

example, we compute the class membership probabilities predicted by the current ensemble, replacing zero probabilities with a small ϵ for smoothing. Labels are then sampled from this distribution, such that the probability of selecting a label is inversely proportional to the current ensemble’s predictions.

3 Experimental Evaluation

3.1 Methodology

Three sets of experiments were conducted in order to compare the performance of ADABOOST, Bagging, DECORATE, and the base classifier, J48 ², under varying amounts of three types of imperfections in the data:

1. **Missing features:** To introduce $N\%$ missing features to a data set of D instances, each of which has F features (excluding the class label), we select randomly with replacement $\frac{N \cdot D \cdot F}{100}$ instances and for each of them delete the value of a randomly chosen feature. Missing features were introduced to both the training and testing sets.
2. **Classification noise:** To introduce $N\%$ classification noise to a data set of D instances, we randomly select $\frac{N \cdot D}{100}$ instances with replacement and change their class labels to one of the *other* values chosen randomly with equal probability. Classification noise was introduced only to the training set and not to the test set.
3. **Feature noise:** To introduce $N\%$ feature noise to a data set of D instances, each of which has F features (excluding the class label), we randomly select with replacement $\frac{N \cdot D \cdot F}{100}$ instances and for each of them we change the value of a randomly selected feature. For nominal features, the new value is chosen randomly with equal probability from the set of *all* possible values. For numeric features, the new value is generated from a Normal distribution defined by the mean and the standard deviation of the given feature, which are estimated from the data set. Feature noise was introduced to both the training and testing sets.

In each set of experiments, ADABOOST, Bagging, DECORATE, and J48 were compared on 11 UCI data sets using the Weka implementations of these methods [17]. Table 1 presents some statistics about the data sets. The target ensemble size of the first three methods was set to 15. In the case of DECORATE, this size is only an upper bound on the size of the ensemble, and the algorithm may terminate with a smaller ensemble if the number of iterations exceeds the maximum limit. As in [9], this maximum limit was set to 50 iterations, and the number of artificially generated examples was equal to the training set size.

To ascertain that no ensemble method was being disadvantaged by the small ensemble size, we ran additional experiments on some datasets with the ensemble size set to 100. The trends of the results are similar to those with ensembles of

² J48 is a Java implementation of C4.5 [12] introduced in [17].

size 15. Details of these experiments are omitted here, but can be found in the extended version of this paper [11].

For each set of experiments, the performance of each of the learners was evaluated at increasing noise levels from 0% to 40% at 5% intervals using 10 complete 10-fold cross validations. In each 10-fold cross validation the data is partitioned into 10 subsets of equal size and the results are averaged over 10 runs. In each run, a distinct subset is used for testing, while the remaining instances are provided as training data.

To compare two learning algorithms across all domains we employ the statistics used in [16], namely the significant win/draw/loss record and the geometric mean error ratio. The win/draw/loss record presents three values, the number of data sets for which algorithm A obtained better, equal, or worse performance than algorithm B with respect to classification accuracy. A win or loss is only counted if the difference in accuracy is determined to be significant at the 0.05 level by a paired t -test.

The geometric mean (GM) error ratio is defined as $\sqrt[n]{\prod_{i=1}^n \frac{E_A}{E_B}}$, where E_A and E_B are the mean errors of algorithm A and B on the same domain. If the geometric mean error ratio is less than one it implies that algorithm A performs better than B , and vice versa. We compute error ratios to capture the degree to which algorithms out-perform each other in win or loss outcomes.

Table 1. Summary of Data Sets

Name	Cases	Classes	Attributes	
			Numeric	Nominal
autos	205	6	15	10
balance-scale	625	3	4	–
breast-w	699	2	9	–
colic	368	2	10	12
credit-a	690	2	6	9
glass	214	6	9	–
heart-c	303	2	8	5
hepatitis	155	2	6	13
iris	150	3	4	–
labor	57	2	8	8
lymph	148	4	–	18

3.2 Results

In this section, we only present the statistics summarized over all 11 datasets. For detailed tables of results, see the extended version of this paper [11].

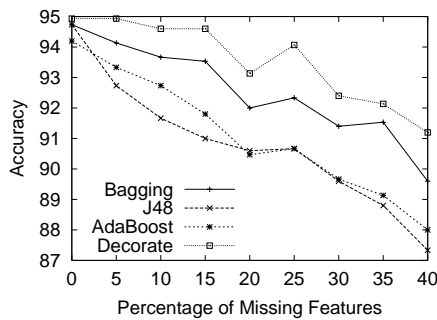
Missing Features: The results of running the algorithms when missing features are introduced, are presented in Tables 2–4. Each table compares the accuracy of DECORATE versus another algorithm for increasing percentages of missing features.

These results demonstrate that DECORATE is fairly robust to missing features, consistently beating the base learner, J48, at all noise levels (Table 2). In fact, when the amount of missing features is 20% or higher, DECORATE produces statistically significant wins over J48 on all datasets. The amount of error reduction produced by using DECORATE is also considerable, as is shown by the mean error ratios.

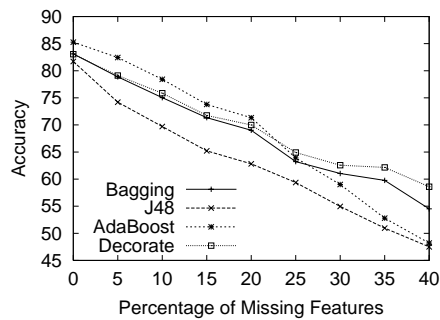
For this kind of imperfection in the data, in general, all of the ensemble methods produce some increase in accuracy over the base learner. However, the improvements brought about by using DECORATE are higher than those caused by both Bagging and ADABOOST. The amount of error reduction achieved by DECORATE also increases with greater amounts of missing features; as is clearly demonstrated by the GM error ratios.

Figure 1(a) shows the results on a dataset that clearly demonstrates DECORATE’s superior performance at all levels of missing features. In Figure 1(b), we see a dataset on which ADABOOST has the best performance when there are no missing features; but with increasing amounts of missing features, both Bagging and DECORATE outperform it.

The superior performance of DECORATE could be attributed to the fact that it adds artificial examples to the training set. These artificial examples do not contain any missing features, and are generated based on the distributions of features estimated over the visible (non-missing) values. A thorough analysis of how using artificial examples can increase robustness to missing features is an important subject for future research.



(a) Iris



(b) Autos

Fig. 1. Missing Features

Table 2. Missing Features: DECORATE vs J48

Noise Level %	0	5	10	15	20	25	30	35	40
<i>Sig. W/D/L</i>	8/3/0	10/1/0	10/1/0	10/1/0	11/0/0	11/0/0	11/0/0	11/0/0	11/0/0
<i>GM Error Ratio</i>	0.8286	0.7882	0.7877	0.7815	0.7921	0.8039	0.8004	0.8095	0.8047

Table 3. Missing Features: DECORATE vs Bagging

Noise Level %	0	5	10	15	20	25	30	35	40
<i>Sig. W/D/L</i>	2/7/2	2/8/1	4/7/0	3/7/1	5/5/1	4/7/0	4/7/0	5/5/1	8/3/0
<i>GM Error Ratio</i>	0.9520	0.9298	0.9201	0.9177	0.9041	0.9083	0.9085	0.9150	0.8882

Table 4. Missing Features: DECORATE vs ADABOOST

Noise Level %	0	5	10	15	20	25	30	35	40
<i>Sig. W/D/L</i>	4/4/3	5/4/2	6/4/1	4/6/1	4/7/0	6/5/0	8/3/0	6/5/0	8/3/0
<i>GM Error Ratio</i>	0.9534	0.9382	0.9197	0.9024	0.9109	0.8982	0.8827	0.8968	0.8876

Classification Noise: The comparison of each ensemble method with the base learner, in the presence of classification noise are summarized in Tables 5-7. The tables provide summary statistics, as described above, for each of the noise levels considered.

The win/draw/loss records indicate that, both Bagging and DECORATE consistently outperform the base learner on most of the datasets at almost all noise levels; demonstrating that both are quite robust to classification noise. In the range of 10-35% of classification noise, Bagging performs a little better than DECORATE, as is seen from the error ratios. This is because, occasionally, the addition of noise helps Bagging, as was also observed in [3].

Unlike, Bagging and DECORATE, ADABOOST is very sensitive to noise in classifications. Though ADABOOST significantly outperforms J48 on 7 of the 11 datasets in the absence of noise, its performance degrades rapidly at noise levels as low as 10%. With 35-40% noise, ADABOOST performs significantly worse than the base learner on 7 of the datasets. Our results on the performance of ADABOOST agree with previously published studies [3, 1, 7]. As pointed out in these studies, ADABOOST degrades in performance because it tends to place a lot of weight on the noisy examples.

Figure 2(a) shows a dataset on which DECORATE has a clear advantage over other methods, at all levels of noise. Figure 2(b) presents a dataset on which Bagging outperforms the other methods at most noise levels. This figure also clearly demonstrates how rapidly the accuracy of ADABOOST can drop below that of the base learner. These results confirm that, in domains with noise in classifications, it is beneficial to use DECORATE or Bagging, but detrimental to apply ADABOOST.

Table 5. Class Noise: DECORATE vs J48

Noise Level %	0	5	10	15	20	25	30	35	40
<i>Sig. W/D/L</i>	8/3/0	7/4/0	8/3/0	8/1/2	8/1/2	6/3/2	6/3/2	7/2/2	8/1/2
<i>GM Error Ratio</i>	0.8286	0.8398	0.8633	0.8734	0.8809	0.8960	0.9121	0.9229	0.8995

Table 6. Class Noise: Bagging vs J48

Noise Level %	0	5	10	15	20	25	30	35	40
<i>Sig. W/D/L</i>	7/4/0	9/2/0	9/2/0	9/2/0	8/3/0	7/4/0	8/2/1	7/3/1	7/3/1
<i>GM Error Ratio</i>	0.8704	0.8687	0.8526	0.8508	0.8443	0.8719	0.8867	0.8972	0.8995

Table 7. Class Noise: ADABOOST vs J48

Noise Level %	0	5	10	15	20	25	30	35	40
<i>Sig. W/D/L</i>	7/3/1	6/1/4	2/4/5	1/5/5	1/4/6	2/2/7	1/4/6	1/3/7	1/3/7
<i>GM Error Ratio</i>	0.8691	0.9930	1.0984	1.1604	1.2322	1.2242	1.2431	1.2120	1.1989

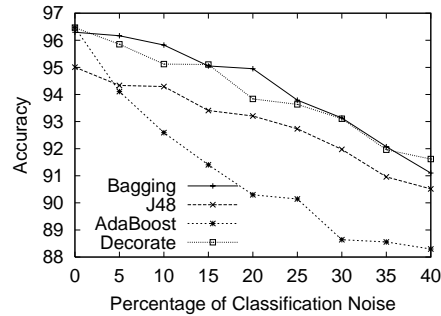
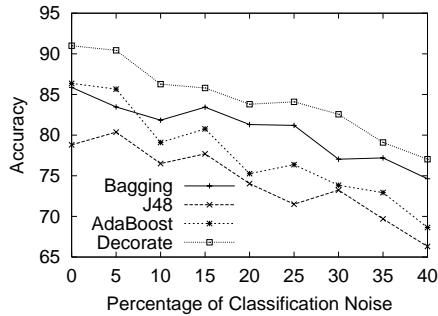
Feature Noise: The results of running the algorithms with noise in the features are presented in Tables 8–10. Each table compares the accuracy of each ensemble method versus J48 for increasing amounts of feature noise.

In most cases, all ensemble methods improve on the accuracy of the base learner, at all levels of feature noise. Bagging performs a little better than the other methods, in terms of significant wins according to the win/draw/loss record. In general, all systems degrade in performance with added feature noise. The drop in accuracy of the ensemble methods seems to mirror that of the base learner, as can be seen in Figure 3. The performance of the ensemble methods seems to be tied to how well the base learner deals with feature noise.

4 Related Work

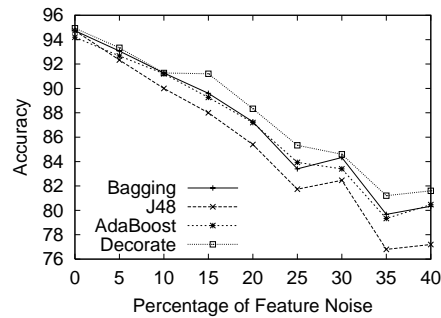
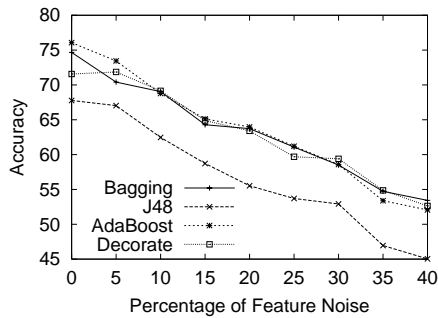
Several previous studies have focused on exploring the performance of various ensemble methods in the presence of noise. A thorough comparison of Bagging, ADABOOST, and Randomization (a method for building a committee of decision trees, which randomly determine the split at each internal tree node) is presented in [3]. This study concludes that while ADABOOST outperforms Bagging and Randomization in settings where there is no noise, it performs significantly worse when classification noise is introduced.

Other studies have reached similar conclusions about AdaBoost [1, 7], and several variations of AdaBoost have been developed to address this issue. For example, Kalai and Servedio [5] present a new boosting algorithm and prove that it can attain arbitrary accuracy when classification noise is present. Another algorithm, Smooth Boosting, that is proven to tolerate a combination of classification and feature noise is presented in [14]. McDonald et al. [8] compare



(a) Labor

(b) Breast-W

Fig. 2. Classification Noise

(a) Glass

(b) Iris

Fig. 3. Feature Noise

ADABOOST to two other boosting algorithms—LogitBoost and BrownBoost—and conclude that BrownBoost is quite robust to noise. In an earlier study an extension to BrownBoost for multi-class problems was presented and shown empirically to outperform ADABOOST on noisy data [7]. However, BrownBoost’s drawback is that it requires a time-out parameter to be set, which can be done only if the user can estimate the level of noise.

5 Future Work

Noise in training data chiefly contributes to an increase in the error due to variance of the base learner; and hence, variance-reduction techniques would be ideal to combat such noise. Bagging is a very effective variance reduction method; whereas ADABOOST is primarily a bias reduction technique, though empirically it has shows to produce some reduction in variance as well [16]. In general, DECORATE also produces significantly more accurate classifiers than

Table 8. Feature Noise: DECORATE vs J48

Noise Level %	0	5	10	15	20	25	30	35	40
<i>Sig. W/D/L</i>	8/3/0	7/4/0	7/4/0	8/3/0	7/3/1	7/4/0	6/5/0	7/4/0	6/5/0
<i>GM Error Ratio</i>	0.8286	0.8335	0.8434	0.8329	0.8593	0.8554	0.8690	0.8723	0.8782

Table 9. Feature Noise: Bagging vs. J48

Noise Level %	0	5	10	15	20	25	30	35	40
<i>Sig. W/D/L</i>	7/4/0	10/1/0	10/1/0	10/1/0	10/1/0	8/3/0	10/1/0	10/1/0	10/1/0
<i>GM Error Ratio</i>	0.8704	0.8586	0.8450	0.8496	0.8473	0.8627	0.8634	0.8614	0.8661

Table 10. Feature Noise: ADABOOST vs. J48

Noise Level %	0	5	10	15	20	25	30	35	40
<i>Sig. W/D/L</i>	7/3/1	7/2/2	8/2/1	7/3/1	8/1/2	8/1/2	8/2/1	7/4/0	8/2/1
<i>GM Error Ratio</i>	0.8691	0.8449	0.8575	0.8455	0.8463	0.8564	0.8830	0.8900	0.8750

the base learner. We are currently investigating whether this improvement in accuracy is mainly due to a reduction in bias or variance. This should lend some more insight into DECORATE’s resilience to imperfections in data.

In our study, all the ensemble methods were used to generate committees of size 15. It may be beneficial to generate larger ensembles, so that the difference in performance between the systems is more pronounced.

An interesting avenue for future work would be to compare the performance of DECORATE and Bagging with the new boosting algorithms mentioned in Section 4. Another interesting subject for future experimentation is testing how the ensemble methods discussed in this study compare to noise elimination techniques such as the ones presented in [15].

6 Conclusion

This paper evaluates the performance of three ensemble methods, Bagging, ADABOOST and DECORATE, in the presence of different kinds of imperfections in the data. Experiments using J48 as the base learner, show that in the case of missing features DECORATE significantly outperforms the other approaches. In the case of classification noise, both DECORATE and Bagging are effective at decreasing the error of the base learner; whereas ADABOOST degrades rapidly in performance, often performing worse than J48. In general, Bagging performs the best at combating high amounts of classification noise. In the presence of noise in the features, all ensemble methods produce consistent improvements over the base learner.

Acknowledgments

Raymond J. Mooney and Prem Melville were supported by DARPA grants F30602-01-2-0571 and HR0011-04-1-007. Lilyana Mihalkova was supported by the MCD Fellowship from the University of Texas at Austin.

References

1. E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting and variants. *Machine Learning*, 36, 1999.
2. Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
3. Thomas G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157, 2000.
4. Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In Lorenza Saitta, editor, *Proc. of 13th Intl. Conf. on Machine Learning (ICML-96)*. Morgan Kaufmann, July 1996.
5. Adam Kalai and Rocco A. Servedio. Boosting in the presence of noise. In *Thirty-Fifth Annual ACM Symposium on Theory of Computing*, 2003.
6. A. Krogh and J. Vedelsby. Neural network ensembles, cross validation and active learning. In *Advances in Neural Information Processing Systems 7*, 1995.
7. Ross A. McDonald, Idris A. Eckley, and David J. Hand. A multi-class extension to the brownboost algorithm. Technical Report TR-03-14, Imperial College, London, 2003.
8. Ross A. McDonald, David J. Hand, and Idris A. Eckley. An empirical comparison of three boosting algorithms on real data sets with artificial class noise. In *Fourth International Workshop on Multiple Classifier Systems*, pages 35–44. Springer, 2003.
9. Prem Melville and Ray Mooney. Constructing diverse classifier ensembles using artificial training examples. In *Proc. of 18th Intl. Joint Conf. on Artificial Intelligence*, pages 505–510, Acapulco, Mexico, August 2003.
10. Prem Melville and Raymond J. Mooney. Creating diversity in ensembles using artificial data. *Information Fusion: Special Issue on Diversity in Multiclassifier Systems*, 2004.
11. Prem Melville, Nishit Shah, Lilyana Mihalkova, and Raymond J. Mooney. Experiments on ensembles with missing and noisy data. Technical Report UT-AI-TR-04-310, University of Texas at Austin, 2004.
12. J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.
13. J. Ross Quinlan. Bagging, boosting, and C4.5. In *Proc. of 13th Natl. Conf. on Artificial Intelligence (AAAI-96)*, pages 725–730, Portland, OR, August 1996.
14. Rocco A. Servedio. Smooth boosting and learning with malicious noise. *The Journal of Machine Learning Research*, 4:633–648, 2003.
15. Sofie Verbaeten and Anneleen Van Assche. Ensemble methods for noise elimination in classification problems. In *Fourth International Workshop on Multiple Classifier Systems*, pages 317–325. Springer, 2003.
16. G. Webb. Multiboosting: A technique for combining boosting and wagging. *Machine Learning*, 40, 2000.
17. Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, San Francisco, 1999.