Removing Bias from Diverse Data Clusters for Ensemble Classification

Sam Fletcher(✉) and Brijesh Verma

Centre for Intelligent Systems, School of Engineering and Technology,
Central Queensland University, Brisbane, QLD 4000, Australia
{s.fletcher,b.verma}@cqu.edu.au

Abstract. Diversity plays an important role in successful ensemble classification. One way to diversify the base-classifiers in an ensemble classifier is to diversify the data they are trained on. Sampling techniques such as bagging have been used for this task in the past, however we argue that since they maintain the global distribution, they do not engender diversity. We instead make a principled argument for the use of k-Means clustering to create diversity. When creating multiple clusterings with multiple k values, there is a risk of different clusterings discovering the same clusters, which would then train the same base-classifiers. This would bias the ensemble voting process. We propose a new approach that uses the Jaccard Index to detect and remove similar clusters before training the base-classifiers, reducing classification error by removing repeated votes. We demonstrate the effectiveness of our proposed approach by comparing it to three state-of-the-art ensemble algorithms on eight UCI datasets.

Keywords: Ensemble · Classification · Clustering · Diversity · Bias · Voting

1 Introduction

By combining the predictions of multiple diverse classifiers, an ensemble of classifiers can perform better than any one classifier can. We propose a novel ensemble classifier that can more accurately classify unseen data than the current state-of-the-art. The proposed algorithm f takes some $n \times m$ data X as input, where the rows $x \in X$ represent independently and identically distributed samples from some universe $\mathcal{X}$, each with a label y that represents the classification category that x belongs to from some output space Y. $f(X)$ is trained using X, learning the underlying patterns in the data that determine what category $y \in Y$ any particular datum x will belong to. The aim of ensemble classification is then to use $f(X)$ to predict the label y of unseen data $z \in Z$, which comes from the same universe $\mathcal{X}$ as X [9]. The columns of X represent features A that describe the properties of each row (i.e., record) x, and it is these features that a classification algorithm uses to learn how to classify unseen data [9]. Examples of

D. Liu et al. (Eds.): ICONIP 2017, Part IV, LNCS 10637, pp. 140–149, 2017.
https://doi.org/10.1007/978-3-319-70093-9_15

classification algorithms include support vector machines [28] and decision trees [21]. By training multiple classifiers, and then combining the predictions made by each classifier into a final overall prediction, an ensemble of classifiers can outperform any individual classifier [8,25].

1.1 Contributions

We propose a generalized ensemble classification algorithm that uses data diversity and base-classifier diversity to build a decision model with low classification error. Our novel contributions are:

- We maximize data diversity by creating subsets of data with large differences in distribution, using k-Means clustering for a large range of k values, $k = 1, \ldots, K$. We ensure that each cluster has sufficient data for training the base-classifiers by bounding K such that if there are k clusters, there are at least an average of k^2 records in each cluster.
- During the incremental clustering process, we compare the new clusters to all previous clusters, and remove (i.e., prune) the new cluster if it is sufficiently similar to a previous cluster. This prevents the prediction voting process from being biased by repetitious votes.
- When classifying a new record z, rather than inputting it into all base-classifiers, we only input the record into the classifiers built using the K clusters that are closest to the new record. This is done regardless of which clusterings the K clusters came from.

After presenting related work in Sect. 2, we introduce the proposed approach in Sect. 3, exploring each component in separate subsections. We then empirically test the proposed algorithm in Sect. 4, before concluding the paper in Sect. 5.

2 Related Work

The notion of "diversity" when building ensemble classifiers has been researched extensively in the past [1,6,14,24]. Despite being difficult to define precisely [6], the overall concept is straight-forward: if all the base-classifiers in an ensemble make the same predictions, they are also making the same mistakes, and if they are all making the same mistakes, there is no advantage in having more than one of them. By diversifying the predictions that the base-classifiers make, the ensemble can perform better than the sum of its parts. There are several types of diversity that an ensemble algorithm can achieve.

Data diversity is achieved by sampling subsets of data from an original dataset, in a way that causes the predictions made by classifiers trained on the subsets to differ from one another. This can be achieved by selecting either records or features (or both) from the original dataset. Duplicating records or features across the new sets of data is viable (such as bagging [2] or random feature subspaces [29]), as is creating mutually-exclusive sets (such as clustering

[22,30,32]). The manipulation of data has successfully diversified the data if the end result is that a diversity of predictions is outputted [14].

Classifier diversity (or "structural diversity" [24] or "heterogeneous ensembling" [18]) has a similar goal of diversifying the predictions that the base-classifiers output. Classifier diversity is achieved by using different classifier algorithms that learn from the data in different ways. By using a variety of classifiers, each with their own advantages and disadvantages, the outputs of the classifiers are diverse [4].

As an example, the Random Forest algorithm [3] uses bagging [2] and random feature subspaces [10,29] to achieve data diversity. It only builds an ensemble of decision trees though, and thus does not target classifier diversity. In this paper, we use both types of diversity in our proposed approach. These are explored below in Sect. 3.

3 Proposed Approach

We first provide an overview of the proposed approach in Sect. 3.1. We then investigate each novel component of the approach one-by-one in the proceeding subsections.

3.1 Overview

The approach can be summarized in the following steps:

- **Step 1:** Calculate the largest number of clusters K we can partition the training data X into without reducing the average number of records in each cluster below the square of the number of clusters.
- **Step 2a:** For $k = 1, \ldots, K$ partition the data X using k-Means clustering.
- **Step 2b:** For each new cluster created, compare its similarity (in terms of the records it contains) to all previously created clusters, using the Jaccard Index [11]. Remove a new cluster if it is very similar to a previous cluster.
- **Step 3a:** For each remaining cluster, check if all the records in the cluster have the same class label y (i.e., the cluster is homogeneous). If so, skip 3b, and future records that are filtered to this cluster will be predicted to have the same label that all the training records had. In effect, the cluster will output v votes for the label y, rather than training v base-classifiers from the homogeneous data.
- **Step 3b:** For each cluster not addressed by 3a, build v base-classifiers using the data in the cluster. Examples of base-classifiers include a decision tree [21], a support vector machine [28], a naive Bayes model [20], a discriminant analysis model [19], a k-nearest neighbors model [31] and a randomly under-sampled boosted model (RUSBoost) [27]. Finding the optimal set of base-classifiers is part of future work.
- **Step 4:** Predict the label of new records z by filtering them into the K closest clusters, using the base-classifiers built from those clusters to each vote on a label, and then using the majority vote as the final prediction.

The specifics of each of these steps are explored below in the following subsections.

3.2 Achieving Data Diversity

To produce a diverse set of base-classifiers, we diversify the training data. Previous research supports using bagging to achieve this [5,16,26], however we argue that the benefits of bagging are in reducing the variance of the models [2], not in promoting diversity. Because bagging maintains the distribution of the underlying data with increasing detail as the sample size increases, it does not provide a diverse range of distributions to the base-classifiers. This problem can be avoided by clustering the data instead, finding regions of data with many attributes in common, and few attributes in common with other regions of data. We achieve this using k-Means clustering [12].

We *could* find a single optimal value for k, and limit our ensemble of base-classifiers to a single clustering. Instead though, we propose using a range of values for k, and building a much larger ensemble. By using values of k ranging from 1 to some upper bound K, we increase diversity further by finding clusters of different sizes (and different distributions) in the training data.

3.3 Choosing K

We need an appropriate K value that gives us a large set of clusters to build many diverse classifiers from, but also provides enough data in each cluster to meaningfully train the base-classifiers with. We balance these two goals with the following heuristic:

1. Let n_k equal the average number of records in each cluster created from k-Means clustering. For a number of clusters k, $n_k = n/k$, where n is the total number of records in the dataset.
2. We limit the maximum size of k such that $n_k \geq k^2$. That is, each cluster has an average number of records equal to at least the square of the number of clusters.
3. Thus we have $n/k \geq k^2$. Re-arranging this formula gives us: $n \geq k^3$.
4. The maximum number of clusters K is therefore:

$$K = \left\lfloor \sqrt[3]{n} \right\rfloor .$$

5. We define the minimum k value for k-Means clustering at $k = 1$.
6. Our proposed ensemble classifier therefore executes k-Means clustering K times, for $k = 1, \ldots, K$.

This balances the number of clusters with the size of each cluster. It is based on a similar concept used to bound k-Means clustering when using it on its own [12,23].

3.4 Pruning Repeated Clusters

In Sects. 3.2 and 3.3, we described how the proposed algorithm creates an increasing number of clusters, from 1 to K, using k-Means clustering. This results in a total of $\sum_{i=1}^{K} i$ clusters, or in other words, $K(K+1)/2$ clusters. This creates a

Table 1. Average difference in classification error compared to when $\theta = 0.9$, across the eight datasets presented in Table 2.

θ	0.5	0.6	0.7	0.8	0.9	1.0	No pruning
Error difference compared to 0.9	+0.056	+0.032	+0.023	+0.016	0.000	+0.007	+0.011

large set of clusters from which to then train base-classifiers from, which will in turn be used to vote on predicted class labels.

However there is a risk in using *all* of these clusters to train classifiers. If two clusters, made during different clusterings (i.e., when $k = i$, and then when $k = j$ such that $i \neq j$), contain all the same records, then the classifiers built from those two clusters will be very similar, maybe even identical (since there is zero data diversity). Not only does this waste computation time, but it also means that when voting on the final predicted label, the votes from these classifiers are doubling up (i.e. getting two votes), biasing the ensemble towards their output.

We remove this bias using the following process: as we grow k towards K, each cluster we create is compared to all previous clusters to check if it is sufficiently diverse. For each new cluster c created in clustering k, we compare the records in c (X_c) to the records of each cluster u in the set of accepted clusters U using the Jaccard Index [11]:

$$J(c, u) = \frac{X_c \cap X_u}{X_c \cup X_u}; \forall u \in U.$$

Computationally, we can calculate $J(c, u)$ using the indexes of the records in X, rather than comparing the contents of each $x \in X$. If there are no records in common, $J(c, u) = 0$; and $J(c, u) = 1$ if both clusters contain precisely the same set of records. If, for some u, $J(c, u) > \theta$, c is not added to U. Here, θ represents a threshold of similarity, which we empirically demonstrate is ideally placed at $\theta = 0.9$ in Table 1. Table 1 presents the average difference in classification error, across the eight datasets presented in Table 2, when $\theta = 0.5, 0.6, 0.7, 0.8, 1.0$ (and when there is no pruning) compared to when $\theta = 0.9$.

Table 2. Details of the eight datasets we use in our experiments, taken from the UCI Machine Learning Repository [15].

Dataset	Records	Features	Labels
Sonar	208	60	2
Heart	270	13	2
Bupa	345	6	2
Ionosphere	351	34	2
WBC	683	9	2
PimaDiabetes	768	8	2
Vehicle	846	18	4
Segmentation	2310	19	7

3.5 Achieving Classifier Diversity

We then build a collection of base-classifiers from the data in each non-pruned cluster. For our experiments in this paper, we use the following six classifiers: a decision tree, a support vector machine, a naive Bayes model, a discriminant analysis model, a k-nearest neighbors model and a randomly under-sampled boosted model. This collection of classifiers is independent of the proposed ensemble framework, and future work will involve investigating the optimal amount and types of classifiers to use.

This diverse collection of classifiers enables the ensemble to discover correlations and patterns in the data that would not be discovered if we limited ourselves to a single classifier, such as what Random Forest does [3]. By discovering different patterns with different classifiers, we diversify the errors made by each base-classifier, which in turn reduces the final classification error (as discussed in Sects. 1 and 2). We can see in Table 5 (presented later in Sect. 4) that we are able to outperform Random Forest on almost all datasets.

3.6 Classifying New Records

Once the ensemble has been built and trained (Steps 1–3 in Sect. 3.1), our model is ready to predict the label of unseen records. When inputting an unseen record z into our ensemble classifier, we propose finding the K clusters in U whose centroids are closest to z, and using the base-classifiers made from those K clusters to predict the label of z.

This approach means that clusters made from different clusterings (when k had different values) are not treated differently from clusters built from the same clustering; if the centroids of two clusters from the same clustering are closer to z than the centroids of two clusters from different clusterings, the closer clusters are used. We also do not want to use the base-classifiers made from every cluster to classify z. Many of the base-classifiers were trained on data that had very different distributions to the distributions that z follows, and did not contain any records that resemble z. To use those classifiers to predict z therefore makes little sense.

4 Experiments and Results

Here we present experiments that cover both individual components of the proposed algorithm, and the overall performance of the algorithm compared to the current state-of-the-art. All experiments are performed using stratified five-fold cross-validation, repeated ten times and aggregated. We perform our experiments on eight datasets from the UCI Machine Learning Repository [15]. The details of the datasets are presented in Table 2. We use Matlab's implementation of k-Means and the base-classifiers for our experiments [17]. We use the default settings in all cases, except for the following:

– The maximum number of iterations for k-Means is increased from 100 to 500, to ensure that centroid convergence occurs.

– Regularization is turned off for discriminant analysis models, to prevent the software throwing an error if a feature with zero variance is inputted.
– Kernel smoothing density estimation is used when building naive Bayes models, instead of using Gaussian distributions, to avoid the software throwing an error if a feature with zero variance is inputted.

4.1 Assessing Cluster Size and Pruning

The first step in our proposed algorithm is to define K. We argue that $\sqrt[3]{n}$ is an appropriate value of K, and this is supported empirically by the results seen in Table 3. In Table 3, we compare $K = \sqrt[3]{n}$ to one smaller value ($\sqrt[4]{n}$) and one larger value ($\sqrt{n}$) of K. Classification error is lowest when $K = \sqrt[3]{n}$ for six of the eight datasets, and very close to lowest for the remaining two (within one standard deviation).

As part of Step 2, we propose removing repeated clusters using the Jaccard Index. Based on the empirical results of Table 1, we recommend setting the similarity threshold to $\theta = 0.9$. This removal of repeated clusters represents a large saving in computation time, preventing v (in our experiments, $v = 6$)

Table 3. The classification error for three different values of K.

Dataset	$K = \sqrt[4]{n}$	$K = \sqrt[3]{n}$	$K = \sqrt{n}$
Sonar	0.1712	**0.1295**	0.1481
Heart	**0.1793**	**0.1778**	0.2252
Bupa	0.2817	**0.2609**	0.2916
Ionosphere	0.0814	**0.0797**	0.0866
WBC	0.0562	**0.0322**	**0.0301**
PimaDiabetes	0.2393	**0.2306**	0.2521
Vehicle	**0.2373**	**0.2352**	0.2532
Segmentation	0.0515	0.0325	**0.0275**

Table 4. The change in classification error with and without cluster pruning.

Dataset	With pruning	Without pruning
Sonar	**0.1295**	0.1501
Heart	**0.1778**	0.1881
Bupa	**0.2609**	0.2817
Ionosphere	**0.0797**	0.0832
WBC	**0.0322**	**0.0340**
PimaDiabetes	**0.2306**	0.2432
Vehicle	**0.2352**	0.2454
Segmentation	0.0325	**0.0269**

redundant classifiers from being trained per removed cluster. It also represents the removal of a high number of repeated votes. The reduction in classification error because of this removal of biased votes can be seen in Table 4. For seven of the eight datasets, the classification error after pruning repeated clusters is lower than or equal to the error without this pruning. On average across all datasets, the average reduction in error is 1.1 percentage points.

4.2 Comparison with Other Ensemble Algorithms

Table 5 presents the classification error our proposed algorithm achieves on eight datasets, compared to the classification error of three other algorithms. One of these algorithms, Random Forest, is included as a benchmark ensemble algorithm due to its reputation as a consistently high-performing algorithm [7]. The other two algorithms represent the current state-of-the-art in ensemble classification, with the results presented being the results the authors reported in their respective papers: Kuncheva and Rodriguez [13]; and Zhang and Suganthan [33]. In both cases, we present the results for the highest performing version of their proposed algorithms; the naive Bayes (NB) version of Kuncheva and Rodriguez's [13], and the version of Zhang and Suganthan's that uses oblique rotation forests with axis-parallel splits (MPRRoF-P) [33].

Out of the eight datasets, the approach proposed in this paper has the lowest classification error in five cases. It has the second-lowest in two cases, and the third-lowest for one dataset (Segmentation). Interestingly, as we saw in Table 4, the Segmentation dataset is also the only dataset for which our proposed cluster pruning does not perform well. This explains the sub-par performance compared to the state-of-the-art for this dataset.

Table 5. The classification error results for four ensemble algorithms, including our proposed approach.

Dataset	Proposed approach	Random forest	Kuncheva 2014 (NB)	Zhang 2015 (MPRRoF-P)
Sonar	**0.1295**	0.1460	0.238	0.1923
Heart	**0.1778**	0.1810	0.195	**0.1763**
Bupa	**0.2609**	0.2727	0.328	N/A
Ionosphere	0.0797	0.0703	0.083	**0.0530**
WBC	**0.0322**	0.0390	0.040	**0.0333**
PimaDiabetes	**0.2306**	0.2396	0.245	0.2474
Vehicle	0.2352	0.2435	0.275	**0.2219**
Segmentation	0.0325	**0.0200**	0.036	**0.0196**

5 Conclusion

Diversity is crucial for building a high-performing ensemble classifier. By performing K clusterings of different sizes, and using these clusters as training data for a diverse set of base-classifiers, the error of the ensemble classifier is reduced. Not only that, but by first pruning repeated clusters, biased votes can be removed from the majority voting process, further reducing classification error. On average, pruning repeated clusters reduces classification error by 1.1%. Looking forward, we plan to investigate what factors affect classifier diversity, and how classifier diversity impacts the performance of ensemble classification.

Acknowledgments. This research was supported by the Australian Research Council's Discovery Project funding scheme (Project Number DP160102639).

References

1. Asafuddoula, M., Verma, B., Zhang, M.: An incremental ensemble classifier leaning by means of a rule-based accuracy and diversity comparison. In: International Joint Conference on Neural Networks, p. 8. IEEE, Anchorage (2017)
2. Breiman, L.: Bagging predictors. Mach. Learn. **24**(2), 123–140 (1996)
3. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
4. Britto, A.S., Sabourin, R., Oliveira, L.E.S.: Dynamic selection of classifiers - a comprehensive review. Pattern Recogn. **47**(11), 3665–3680 (2014)
5. Chang, K.H., Parker, D.S.: Complementary prioritized ensemble selection. In: International Joint Conference on Neural Networks, pp. 863–872 (2016)
6. Didaci, L., Fumera, G., Roli, F.: Diversity in classifier ensembles: fertile concept or dead end? In: Zhou, Z.-H., Roli, F., Kittler, J. (eds.) MCS 2013. LNCS, vol. 7872, pp. 37–48. Springer, Heidelberg (2013). doi:10.1007/978-3-642-38067-9_4
7. Fernández-Delgado, M., Cernadas, E., Barro, S., Amorim, D., Amorim Fernández-Delgado, D.: Do we need hundreds of classifiers to solve real world classification problems? J. Mach. Learn. Res. **15**(1), 3133–3181 (2014)
8. Gopika, D., Azhagusundari, B.: An analysis on ensemble methods in classification tasks. Int. J. Adv. Res. Comput. Commun. Eng. **3**(7), 7423–7427 (2014)
9. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, San Diego (2006)
10. Ho, T.K.: The random subspace method for constructing decision forests. IEEE Trans. Pattern Anal. Mach. Intell. **20**(8), 832–844 (1998)
11. Jaccard, P.: Etude comparative de la distribution florale dans une portion des Alpes et du Jura. Bull. Soc. Vaudoise Sci. Nat. **37**, 547–579 (1901)
12. Jain, A.K.: Data clustering: 50 years beyond K-means. Pattern Recogn. Lett. **31**(8), 651–666 (2010)
13. Kuncheva, L.I., Rodríguez, J.J.: A weighted voting framework for classifiers ensembles. Knowl. Inf. Syst. **38**(2), 259–275 (2014)
14. Kuncheva, L.I., Whitaker, C.J.: Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. Mach. Learn. **51**(2), 181–207 (2003)
15. Lichman, M.: UCI Machine Learning Repository (2013). http://archive.ics.uci.edu/ml/
16. Mao, S., Jiao, L., Xiong, L., Gou, S., Chen, B., Yeung, S.K.: Weighted classifier ensemble based on quadratic form. Pattern Recogn. **48**(5), 1688–1706 (2015)

17. MathWorks: MATLAB and Statistics and Machine Learning Toolbox
18. Mendes-Moreira, J., Soares, C., Jorge, A.M., Sousa, J.F.D.: Ensemble approaches for regression. ACM Comput. Surv. **45**(1), 1–40 (2012)
19. Mika, S., Ratsch, G., Weston, J., Schölkopf, B., Muller, K.R.: Fisher discriminant analysis with kernels. In: IEEE Signal Processing Society Workshop, pp. 41–48. IEEE (1999)
20. Ng, A.Y., Jordan., M.I.: On discriminative vs. generative classifiers: a comparison of logistic regression and naive Bayes. In: Advances in Neural Information Processing Systems, pp. 841–848. NIPS (2002)
21. Quinlan, J.R.: C4.5: Programs for Machine Learning, 1st edn. Morgan Kaufmann, Burlington (1993)
22. Rahman, A., Verma, B.: A novel layered clustering based approach for generating ensemble of classifiers. IEEE Trans. Neural Netw. **22**(5), 781–792 (2011)
23. Rahman, M.A., Islam, M.Z.: A hybrid clustering technique combining a novel genetic algorithm with K-Means. Knowl.-Based Syst. **71**(1), 345–365 (2014)
24. Ren, Y., Zhang, L., Suganthan, P.N.: Ensemble classification and regression - recent developments, applications and future directions. IEEE Comput. Intell. Mag. **11**(1), 41–53 (2016)
25. Rokach, L.: Ensemble-based classifiers. Artif. Intell. Rev. **33**(1), 1–39 (2010)
26. Santucci, E., Didaci, L., Fumera, G., Roli, F.: A parameter randomization approach for constructing classifier ensembles. Pattern Recogn. **69**(1), 1–13 (2017)
27. Seiffert, C., Khoshgoftaar, T.M., Hulse, J.V., Napolitano, A.: RUSBoost: a hybrid approach to alleviating class imbalance. IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. **40**(1), 185–197 (2010)
28. Suykens, J.A.K., Vandewalle, J.: Least squares support vector machine classifiers. Neural Process. Lett. **9**(3), 293–300 (1999)
29. Tan, C., Li, M., Qin, X.: Random subspace regression ensemble for near-infrared spectroscopic calibration of tobacco samples. Anal. Sci. **24**(5), 647–653 (2008)
30. Verma, B., Rahman, A.: Cluster oriented ensemble classifier: impact of multi-cluster characterisation on ensemble classifier learning. IEEE Trans. Knowl. Data Eng. **24**(4), 605–618 (2012)
31. Weinberger, K., Blitzer, J., Saul, L.: Distance metric learning for large margin nearest neighbor classification. In: Advances in Neural Information Processing Systems, pp. 1473–1480 (2006)
32. Yang, Y., Jiang, J.: Hybrid sampling-based clustering ensemble with global and local constitutions. IEEE Trans. Neural Netw. Learn. Syst. **27**(5), 952–965 (2016)
33. Zhang, L., Suganthan, P.N.: Oblique decision tree ensemble via multisurface proximal support vector machine. IEEE Trans. Cybern. **45**(10), 2165–2176 (2015)