A Novel Multiple Classifier Generation and Combination Framework Based on Fuzzy Clustering and Individualized Ensemble Construction

Given Name Surname
dept. name of organization (of Aff.), organization, City, Country
abc@xyz.org

Abstract

Multiple classifier system (MCS) has become a successful method for improving classification performance. We believe that the two crucial steps of MCS - base classifier generation and multiple classifier combination, need to be designed coordinately to produce robust results. In this work, we show that for different testing instances, better classifiers may be trained from different subdomains of training instances including, for example, neighboring instances of the testing instance, or even instances far away from the testing instance. To utilize this intuition, we propose Individualized Classifier Ensemble (ICE). ICE groups training data into overlapping clusters, builds a classifier for each cluster, and then associates each training instance to the top-performing models while taking into account model types and frequency. In testing, ICE finds the k most similar training instances for a testing instance, then predicts class label of the testing instance by averaging the prediction from models associated with these training instances. Evaluation results on 49 benchmarks show that ICE has a stable improvement on a significant proportion of datasets over existing MCS methods. ICE provides a novel choice of utilizing internal patterns among instances to improve classification, and can be easily combined with various classification models.

1 Introduction

Multiple classifier system (MCS), including ensemble classifiers and mixture of experts, has established itself as an effective and practical solution to address challenges in supervised learning, such as functional complexity, insufficient training data, high dimensionality of feature space, and noise in training data, among others. Many excellent comprehensive reviews on MCS algorithms are available [Rokach, 2010; Kuncheva, 2004; Oza and Tumer, 2008].

Learning a MCS usually includes two critical steps: base classifier generation, and multiple classifier combination, although sometimes the two steps are intrinsically integrated. Different MCS methods can be distinguished by how these two steps are performed. According to model generation strategies, existing MCS methods usually fall into one of the following two categories: random methods and deliberate methods. The former generates models by injecting random perturbations into the training data or training process [Breiman, 1996; Breiman, 2001], In contrast, the latter attempts to generate multiple classifiers in a more systematic, principled way, e.g., by iteratively re-weighting the training instances with emphasis on previously misclassified instances, a technique known as boosting [Freund and Schapire, 1995], or by first clustering the training instances and then learning submodels from each cluster [Vilalta *et al.*, 2003; Kuncheva, 2000]. According to model combination strategies, MCS methods can also be grouped into two categories: voting-based and learning-based. Most popular ensemble methods (e.g., bagging and boosting) take a (weighted) voting from all models in the pool. Other methods attempt to learn a high-level model in order to determine which model(s) should be selected for the prediction task, or to learn a more complex function to combine the outputs of all models in the pool. Learning-based model combination algorithms include stacking, dynamic model selection, among many others [Wolpert, 1992; Giacinto and Roli, 2001].

Overall, ensemble approaches combining randomized model generation and voting (e.g. bagging and random forest) have been more successful / popular, probably due to their simplicity and less over-fitting. On the other hand, it has been shown that careful integration of deliberate models and learning-based model combination can be very effective on specific problem domains [Gashler *et al.*, 2008]. In particular, empirical studies suggest that many classification problems consist of subdomains, which can potentially benefit from constructing and selecting submodels [Jahid *et al.*, 2014; Vilalta *et al.*, 2003; Kuncheva, 2000]. The challenge, however, lies in whether these subdomains can be corrected identified at training, and whether the submodels can be corrected selected for individual cases at prediction time.

Here, we design a general MCS framework, Individualized Classifier Ensemble (ICE), with two key ideas. First, it constructs a large pool of submodels that have low bias when applied to appropriate instances. This is achieved by applying a strong learner (in contrast to the high-bias, low-variance models commonly used in a few ensemble methods) to individual overlapping clusters of instances that represent possible subproblems. Second, a simple yet effective, learning-free

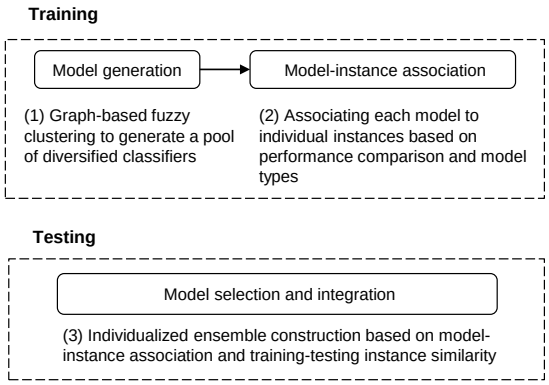

Figure 1: **Overview of the ICE framework.**

method is used to obtain different combinations of submodels for different testing instances. The learning-free nature of the method reduces the chance of selecting wrong models, therefore ensures that the combination of the selected submodels is better than, or at least no worse than, an average of all submodels.

Experimental results on 49 datasets from different domains show that ICE consistently outperforms the competing methods. Furthermore, detailed component analysis shows that both steps of our algorithm have positive contributions as expected. In addition, analysis of the submodels can shed light on the internal structure of the problem, which can potentially be used to further increase prediction performance, or to improve mechanistic understanding of the problem. The framework can be easily combined with existing classifiers and applied to many domains.

2 Methods

Fig. 1 shows a brief overview of ICE, which starts with generating a pool of diverse yet subdomain representative candidate classifiers from subsets of training instances, obtained by a graph-based clustering method that can detect overlapping clusters (Algorithm 1, 4). Then, these classifiers are associated to individual training instances based on their relative prediction performance on the instance, taking into account model types and frequency (Algorithm 2). Finally, in testing/prediction stage, the nearest neighbors of a test instance are identified from the training dataset and the classifiers associated with these neighboring instances are selected to form an ensemble for prediction (Algorithm 3). While the general ICE framework is flexible and the individual components can be re-designed with domain-specific information, several design principles are crucial and are discussed below.

2.1 Training

Basic denotations

We denote a dataset as $\boldsymbol{A} = \{(x_i, y_i)\}_{i=1}^{Q}$, which consists of Q training instances, where $x_i \in \boldsymbol{X}$ is an R dimensional feature vector and $y_i \in \boldsymbol{Y}$ is the binary label of instance i. The clustering result on $\boldsymbol{X}$ is denoted as $\boldsymbol{C} = \{c_i\}_{i=1}^{L}$; c_i is the ith cluster; L is the total number of clusters. Here we designate the last cluster c_L of $\boldsymbol{C}$ to be the whole set of instances. Without loss of generality, we assume the class labels are binary.

Algorithm 1 Training

procedure TRAIN ($\boldsymbol{X}$, $\boldsymbol{Y}$, L, w, s)
  $\boldsymbol{C} \leftarrow$ CLUSTERING ($\boldsymbol{X}, L$)
  $\boldsymbol{O} = \{o_i\}_{i=1}^{L} \leftarrow \emptyset$
  for c_j in $\boldsymbol{C}$ **do**
    $o_j \in \boldsymbol{O} \leftarrow$ Train model on c_j using base-classifier
  end for
  $\boldsymbol{D} \leftarrow$ ASSOCIATE ($\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{C}, \boldsymbol{O}, w, s$)
  return $\boldsymbol{D}, \boldsymbol{O}$
end procedure

Associating models to instances

Incorrect model selection can significantly degrade the performance of the algorithm compared to simply averaging all submodels. When the number of training instances is relatively small, supervised learning based model selection tends to overfit. Therefore, we propose a robust learning-free method, which performs model-instance association at training time and k-NN based model selection at prediction time. Importantly, the model-instance association step takes a Bayesian approach by using different cutoffs for different types of submodels, which reflects their frequency in the pool and the probability for them to outperform other types of submodels.

Formally, given the clustering result on instances, $\boldsymbol{C} = \{c_i\}_{i=1}^{L}$, where c_L is the whole set of instances, the corresponding set of models built on the clusters by a base learner (e.g., Random Forest) is denoted as $\boldsymbol{O} = \{o_i\}_{i=1}^{L}$. Here we call a model $o_i, i \in [1, L-1]$ as a '$partial$' model, since each model is built on a subset of the training instances, and, we call model o_L as the '$whole$' model, which is built on the whole set of instances. In the rare case that all instances in a cluster belong to the same label, the corresponding '$partial$' model will simply return the class label. (The average within-cluster class ratio across all datasets used in this study was found to be 73:27.) The class probabilities predicted by all models are stored in $\boldsymbol{P} = \langle p_{ij} \rangle_{Q \times L}$; p_{ij} is the predicted class probability for instance i by model j; p_{iL} is the prediction probability for instance i by model built on the whole set of training instances. Note that if instance i is NOT a member of cluster c_j (in which case, we call model o_j to be a '$remote$' model of instance i), the model is directly used to predict p_{ij} for instance i; on the other hand, if instance i is a member of cluster c_j (in which case we call model o_j a '$local$' model of instance i), the value p_{ij} is obtained by 10-fold cross-validation using instances in this cluster. This process ensures that the performance evaluation used for model-instance association is not inflated, as an instance is never evaluated by a model that used the instance in training. Importantly, by not having any designated validation dataset, we are able to keep as many instances as possible for training, an important feature for small training data.

The prediction error table, $\boldsymbol{E} = \langle e_{ij} \rangle_{Q \times L}$ is derived from $\boldsymbol{P}$; $e_{ij} = |p_{ij} - y_i|$ is the prediction error for instance i by model o_j. Each row of $\boldsymbol{E}$, $e_{i\bullet}$, represents the prediction error of different models on instance i. Given the empirical results that $local$ models usually work slightly better than $whole$ model and $remote$ models, as well as the fact that there are

Algorithm 2 Associate instances with models by calculating the decision table

procedure ASSOCIATE ($\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{C}, \boldsymbol{O}, w, s$)
  $\boldsymbol{P} = \langle p_{ij} \rangle_{Q \times L} \leftarrow \langle 0 \rangle_{Q \times L}$
  $\boldsymbol{E} = \langle e_{ij} \rangle_{Q \times L} \leftarrow \langle 0 \rangle_{Q \times L}$
  for c_j in $\boldsymbol{C}$ **do**
    $\langle p_{\bullet j} \rangle_{Q \times 1} \leftarrow$ Pred. $\boldsymbol{Y}$ with Cross Val. $(o_j, c_j, \boldsymbol{X})$
    $\langle e_{\bullet j} \rangle_{Q \times 1} \leftarrow |\langle p_{\bullet j} \rangle_{Q \times 1} - \boldsymbol{Y}|$
  end for
  for $\langle e_{i\bullet} \rangle_{1 \times L}$ in $\boldsymbol{E}$ **do**
    $e_{iL} \leftarrow (e_{iL} - w)$
    $e_{ij} \leftarrow (e_{ij} - s)$ if $x_i \in c_j$
  end for
  $\boldsymbol{D} = \langle d_{ij} \rangle_{Q \times L} \leftarrow \langle 0 \rangle_{Q \times L}$
  for d_{ij} in $\boldsymbol{D}$ **do**
    $d_{ij} \leftarrow 1$ if $e_{ij} \leq e_{iL}$, 0 otherwise
  end for
  return $\boldsymbol{D}$
end procedure

many more $remote$ models than $local$ models in the pool, we introduce two parameters to easily balance the proportion of $local$, $whole$ and $remote$ models in the ensemble: w as the advantage score of the $whole$ model, and s the advantage score of each $local$ model. Usually $s > w > 0$ to promote the inclusion of $local$ models and demote $remote$ models, unless the remote model is significantly better than the $whole$ model. Each row of $\boldsymbol{E}$ is adjusted such that $e_{iL} \leftarrow (e_{iL} - w)$, and, $e_{ij} \leftarrow (e_{ij} - s)$ if $x_i \in c_j$. Then, the decision table, $\boldsymbol{D} = \langle d_{ij} \rangle_{Q \times L}$, $d_{ij} \in \{1, 0\}$, where $d_{ij} = 1$ indicates association between model o_j and instance i, is derived from the error table $\boldsymbol{E}$, by

$$d_{ij} = \begin{cases} 1, & \text{if } e_{ij} \leq e_{iL} \\ 0, & \text{otherwise} \end{cases}$$

2.2 Testing / prediction

Algorithm 3 Testing

procedure PREDICT (x_t, $\boldsymbol{X}$, $\boldsymbol{D}$, $\boldsymbol{O}$, N, α, β)
  $K^{nb} = \langle k_i \rangle_{N \times 1} \leftarrow$ select N nearest Nbr. of x_t
  $\boldsymbol{O}^{nb} \leftarrow \emptyset$
  for k_i in K^{nb} **do**
    $J \leftarrow \langle d_{k_i \bullet} \rangle_{1 \times L} = 1$
    $\boldsymbol{O}^{nb} \leftarrow \boldsymbol{O}^{nb} \cup \boldsymbol{O}_J$
  end for
  $p^{whole} \leftarrow$ Predict by base-classifier (o_L, x_t)
  $M \leftarrow length(\boldsymbol{O}^{nb})$
  $\boldsymbol{P}^{partial} = \left\langle p_i^{partial} \right\rangle_{M \times 1} \leftarrow \emptyset$
  for o_i in $\boldsymbol{O}^{nb}$ **do**
    $p_i^{partial} \leftarrow$ Predict by base-classifier (o_i, x_t)
  end for
  $p^t \leftarrow$ Equation 1$(\boldsymbol{P}^{partial}, p^{whole}, M, N, \alpha, \beta)$
  return p^t
end procedure

For a test instance x_t, ICE first finds its N nearest neighbors from the training dataset, then predicts its class label y_t by averaging the class probabilities predicted by the models associated with the neighbor training instances (Algorithm 3). Formally, the *PREDICT*() algorithm first selects N nearest neighbors of x_t from $\boldsymbol{X}$, and stores the indices of the neighbor instances in $K^{nb} = \langle k_i \rangle_{N \times 1}$. Then, for each neighbor instance k_i, the algorithm looks up in the corresponding decision table $d_{k_i \bullet}$ to find the models associated with the neighbor instance, and stores the associated '$partial$' models of x_t in $\boldsymbol{O}^{nb}$. The number of '$partial$' models in $\boldsymbol{O}^{nb}$ is denoted as M. Note that although $o_i^{nb} \in \boldsymbol{O}$, $\boldsymbol{O}^{nb}$ is not a subset of $\boldsymbol{O}$, since $\boldsymbol{O}^{nb}$ may contain duplicated models. Then we denote $\boldsymbol{P}^{partial} = \left\langle p_i^{partial} \right\rangle_{M \times 1}$ as the '$partial$' model predictions, and each $p_i^{partial}$ is predicted by o_i on x_t. The predicted class probability by the $whole$ model is denoted as p^{whole}. Then the predicted class probability of x_t is calculated by:

$$p^t = \frac{\sum_{i=1}^{M} p_i^{partial} + (\alpha M + \beta N) \cdot p^{whole}}{(\alpha + 1)M + \beta N}, \qquad (1)$$

where α is the parameter to balance the weight of '$partial$' models and the '$whole$' model; β is the parameter to adjust the weight of '$whole$' models based on the number of top neighbors to ensure at least one p^{whole} will be used in case there is no '$partial$' model.

2.3 Graph-based Fuzzy Clustering

As clustering can be subjective and unstable, we recommend generating a large number of relatively independent but overlapping clusters. In addition, the clusters need to cover both local and global information. In our design, we use a graph-based clustering algorithm that chooses a set of furthest points to initiate a random walk process and use probability cutoffs to control cluster size, maintaining both microscopic and large-scale information.

The algorithm works as follows. We first calculate an instance-instance distance matrix on $\boldsymbol{X}$ by Euclidean distance and store it in $\boldsymbol{S}$. Then, we construct a k-NN graph $\boldsymbol{G}$ by keeping the top $\lceil log_{10} Q \rceil$ neighbors for each node in $\boldsymbol{S}$. Afterwards, a random walk with a restart probability of 0.3 is performed on the k-NN graph $\boldsymbol{G}$ to obtain an affinity matrix, $\boldsymbol{W}$ [Tong *et al.*, 2006]. Next, a set of points, $\boldsymbol{T} = \langle t_j \rangle_{\lceil 10 log_{10} Q \rceil \times 1}$, is identified as cluster centers: from $\boldsymbol{W}$, the node with the largest total incoming probability, t_1, is chosen as the center point of the first cluster; cluster centers for the other clusters are selected by finding the furthest node from the current center points.

Finally, a set of probability cutoffs are applied on $\boldsymbol{W}$ to identify direct neighbors of each cluster center as members of the cluster, such that the average cluster size is $z \in \{2^i \mid 4 \leqslant i \leqslant log_2 \lceil \frac{3}{4} Q \rceil\}$; therefore, each center point has a few clusters associating with it, maintaining both local and global information. We designate the last cluster c_L of $\boldsymbol{C}$ to be the whole set of instances. Here the number of clusters L is in log scale of Q. A classifier is then built using instances from each cluster.

2.4 Relationship with Existing MCS Methods

ICE differs from most existing ensemble methods significantly in both model generation and model combination. Popular ensemble methods such as Bagging and Random Forest (RF) generate submodels using random subsets of data, and combine them using voting. In order for these methods to work effectively, a large number of submodels is needed to reduce overall bias. In contrast, ICE generates submodels to deliberately increase model diversity by clustering training instances. A carefully designed model-instance association algorithm helps identify the best ensemble for individual instances at prediction time. In comparison, boosting generates submodels that focus on different groups of training instances. However, grouping of instances is done implicitly by iterative re-weighting and therefore lack a global view of instance space; in addition, since there is no model selection at prediction time, boosting tend to overfitting in the presence of noisy training instances.

Mixtures of experts is a class of neural network models attempting to simultaneously learn multiple submodels as well as a gating function that assigns each instance to one or more submodels [Jacobs *et al.*, 1991; Jordan and Jacobs, 1994] . With similar idea, several methods use clustering as a preprocessing step for classification [Vilalta *et al.*, 2003; Kuncheva, 2000]. These algorithms force each instance to be in a disjoint cluster, which reduces the number of instances at training time. In addition, prediction is done only by cluster-specific models so the cost of incorrect model selection is high. Empirical results presented in the original papers show mixed performance when compared to other MCS algorithms [Jacobs *et al.*, 1991; Vilalta *et al.*, 2003; Kuncheva, 2000].

Finally, a series of methods have been developed recently under the common name 'dynamic model selection' [Cruz *et al.*, 2015; Ko *et al.*, 2008; Ko *et al.*, 2008; Kurzynski *et al.*, 2016; Woloszynski *et al.*, 2012; Woods *et al.*, 1997; Giacinto and Roli, 2001; Giacinto and Roli, 1999]. These approaches take an ensemble of base classifiers (e.g, from bagging), then attempt to learn a high-level classification model using, for example, instance-instance similarities and model-model correlations, as input features. While conceptually appealing, these methods tend to overfit and have poor performance when training data is limited. In our opinion, the marriage between random model generation and learning-based model combination is a poor choice, since the relatively small number of random models (compared to the possible number of instance combinations) does not guarantee that there is necessarily any *predictably* better submodel than a simple average of all submodels.

3 Results

3.1 Data and Experimental Setup

Characteristics of the 49 benchmark datasets are shown in Figure 2. The datasets are collected from UCI machine learning repository and Kaggle Dataset for binary classification, with number of instance between 100 and 3,000, number of features between 3 and 1500, and percentage of majority class ranging from 50% to 77%. A total of 42 datasets

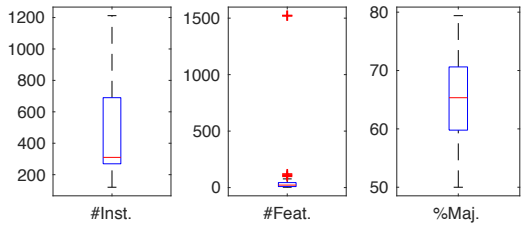

Figure 2: **Characteristics of datasets used in evaluation.** The three charts show the distribution of number of instances, number of features and percentage of majority class, respectively.

from UCI and 23 datasets from Kaggle meet the criteria (18 of which appeared in both repositories). In addition, we add two cancer-related datasets - breast-cancer-nki [Van De Vijver *et al.*, 2002] and breast-cancer-wang [Wang *et al.*, 2005]. The data preprocessing mainly follows [Fernández-Delgado *et al.*, 2014], which includes a Z-Score transformation based normalization. For nine datasets with nominal features we use two different methods to handle nominal features: (i) removing nominal features (denoted with suffix '-1' in Figure 2), (ii) using One-Hot encoding (denoted with suffix '-2' in Figure 2). Since all features in dataset 'tic-tac-toe' are nominal, this dataset only has the One-Hot encoding version.

Performance of each classification method is evaluated by 10-fold cross validation and measured by AUC. The base model of ICE in the evaluation is Random Forest with 100 trees (denoted as RF-100), while the main comparison model is Random Forest with 10,000 trees (denoted as RF-10k). It is worth noting that the parameters of ICE are chosen intuitively without extensive tuning. Parameter analysis results show that the performance of ICE is insensitive to changes within a wide range of parameter values.

3.2 Empirical Evidence Supporting Cluster-Based Ensemble Classification

To verify our assumption that, for each testing instance, some subset of training instances may provide a better classification model than the whole set of training instances, we perform a simple experiment as follows: first, each dataset is clustered into three disjoint clusters using k-means. We denote the clusters as cluster-a, b and c respectively, with their cluster size decreasing. Then using instances in each cluster for cross-testing: we compared the prediction AUC for each cluster using instances from cluster a, b, c or the whole dataset, respectively, as training data. We adopted notation a-b to denote the situation where we use the cluster a trained model to make predictions on cluster b instances.

To have a fair evaluation, when using a larger cluster to predict a smaller cluster, we randomly select the same number of instances from the larger cluster as the size of the smaller cluster to be the training data; when use a smaller cluster to predict a larger cluster, we use all instances in the smaller cluster and randomly select some instances from the larger cluster (making sure that they are not in the fold of testing) to be the training instances, such that the total number of training instances is the same as the number of instances in the

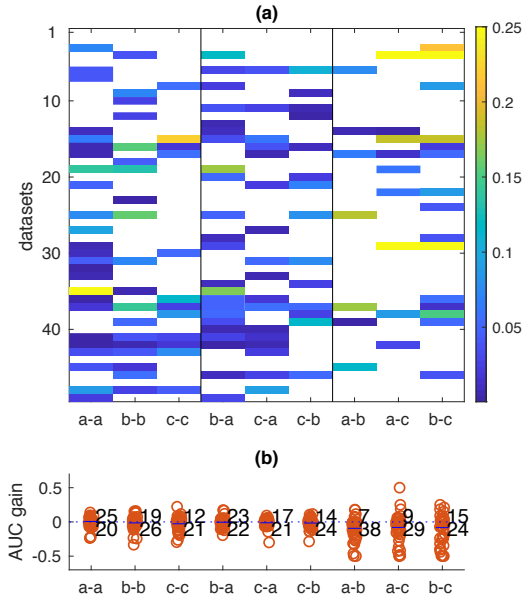

Figure 3: **AUC gain of** *partial* **models over** *whole* **model.** (a) Color scale represents AUC gain; AUC gain ≤ 0 is removed to emphasize on potential benefits of using subsets of training instances. (b) each node represents the AUC gain on a dataset using a cluster of training data to predict a testing cluster. The two numbers on the right of each column are the number of datasets with AUC gain > 0 and AUC gain < 0.

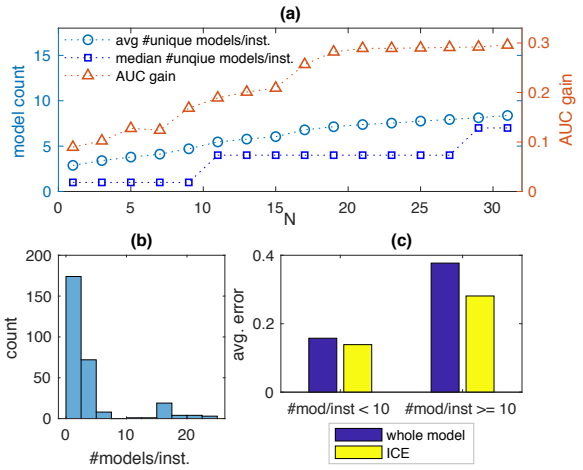

Figure 4: **Analysis of ICE submodels on a breast cancer dataset. (a)** AUC gain of ICE varies as a function of model counts per instance. **(b)** distribution of number of unique '*partial*' models per instance. **(c)** Predictions by ICE have a lower error rate than Random Forest on instances with $\geqslant$10 models.

larger cluster.

From Figure 3, with only three disjoint clusters, in more than 88% of the datasets, at least one of the *partial* models can outperform the *whole* model (Figure 3a and b. Interestingly, while in general the *remote* models do not perform well, some of them have the largest performance gain compared to the *whole* model (column *a-b*, *a-c* and *b-c*). Collectively, this experiment shows the potential benefit of using a cluster of instances to improve prediction accuracy. On the other hand, the results also signifies the importance to predict, for each test instance, whether *partial* models (and which) should be used.

3.3 ICE Outperforms Existing MCS Competitors

Table 1 shows AUC of ICE (RF-100 as the base model) on 49 benchmark datasets compared to multiple MCS methods, including RF-10k, and seven dynamic model selection approaches. META-DES [Cruz *et al.*, 2015] has two versions in this evaluation, using Perceptron (the base classifier choice of the original META-DES paper) and RF-100 (comparable with RF-10k and ICE) respectively. The base classifier is RF-100 for the other six dynamic model selection methods - KNORA-U [Ko *et al.*, 2008], KNORA-E [Ko *et al.*, 2008], DES-PRC [Kurzynski *et al.*, 2016; Woloszynski *et al.*, 2012], OLA [Woods *et al.*, 1997], MCB [Giacinto and Roli, 2001] and A Priori [Giacinto and Roli, 1999], which is the suggested setting plus RF-100 to make comparable with other methods. We use the suggested parameters for dynamic model selection approaches [Cruz *et al.*, 2018].

As shown, ICE has a much better performance than the other methods. Comparing ICE with Random Forest (RF-10k), we can see that ICE outperforms RF-10k on most datasets (ICE wins over RF on 28 out of 49 datasets - %57.1 winning ratio) and suffers from only minor performance loss on 13 datasets. The t-test p-value of gain = 0.018, which is significant, and, there are 7 datasets with AUC gain over 5%, while no dataset with AUC loss over 3%. Notably, ICE uses less than 100 unique models - on average 47 models per prediction. ICE still has room for improvement by parameter tuning and improved clustering methods.

The performance gain of ICE over RF can be attributed to the use of specifically generated models for subproblems and individualized model association and selection step. In addition, ICE outperforms the seven dynamic selection methods. Each of the dynamic selection methods has unique contributions on model selection or integration. However, none of them focuses on deliberately generating models for specific subproblems as the fuzzy clustering that ICE uses. In addition, the unique instance-model association of ICE can utilize all training instances, comparing to dynamic selection methods such as META-DES, which separates training data into META learning and dynamic selection datasets, therefore lead to more data loss and weaker base classifiers.

3.4 Classification Improved by Accurately Predicting 'Hard; Instances with '*partial*' Models

ICE has a stable AUC gain on most of datasets over a large range of parameter variation, and the dataset with one of the most dramatic improvement using ICE is the 15-breast-cancer-1 dataset. In Figure 4a, firstly, as the parameter N increases, the average number of models per instance also increases and the performance of ICE continue to increase until $N = 20$ and reaches a plateau (where AUC is near 1). In

Table 1: **AUC of ICE and competing methods on 49 datasets**

		1	2	3	4	5	6	7	8	9	10
		META-DES Perceptron	META-DES RF	KNORA-U	KNORA-E	DES-PRC	OLA	MCB	A Priori	Random Forest	ICE
1	acute-nephritis	1.000	0.960	0.960	0.960	0.960	0.960	0.931	0.953	1.000	1.000
2	acute-inflammation	1.000	1.000	1.000	1.000	1.000	1.000	0.983	0.967	1.000	1.000
3	echocardiogram	0.756	0.780	0.769	0.711	0.780	0.746	0.671	0.642	**0.837**	***0.854***
4	hepatitis	0.570	0.666	0.627	0.689	0.643	0.665	0.648	0.641	**0.856**	***0.872***
5	wine	0.930	0.941	0.960	0.960	0.960	0.897	0.890	0.892	***0.999***	***0.999***
6	planning	0.477	0.435	**0.489**	0.450	**0.489**	0.431	0.429	0.464	0.420	***0.557***
7	parkinsons	0.591	0.769	0.678	0.776	0.692	0.706	0.675	0.731	**0.956**	***0.970***
8	breast-cancer-wisc-prog	0.543	0.583	0.569	0.583	0.569	0.600	0.526	0.571	***0.677***	***0.677***
9	conn-bench-sonar-mines-rocks	0.675	0.694	0.694	0.693	0.698	0.639	0.622	0.652	**0.934**	***0.944***
10	glass-id	0.763	0.839	0.842	0.852	0.835	0.832	0.807	0.869	***0.983***	**0.979**
11	spect	0.574	0.708	0.691	0.686	0.700	0.634	0.670	0.614	**0.736**	***0.759***
12	spectf	0.515	0.789	0.789	0.720	0.796	0.706	0.714	0.697	***0.869***	**0.860**
13	statlog-heart-1	0.688	0.698	0.733	0.705	0.724	0.688	0.632	0.668	**0.816**	***0.817***
14	statlog-heart-2	0.812	0.814	0.838	0.788	0.824	0.755	0.773	0.761	**0.902**	***0.904***
15	breast-cancer-1	0.517	**0.625**	0.557	0.612	0.546	0.554	0.546	0.515	0.608	***0.978***
16	breast-cancer-2	0.487	0.589	0.568	0.581	0.578	0.583	0.607	0.561	**0.668**	***0.925***
17	breast-cancer-wang	0.528	0.594	0.585	0.548	0.585	0.507	0.568	0.543	***0.697***	**0.671**
18	colposcopies	0.545	0.663	0.660	0.600	0.660	0.642	0.598	0.551	***0.806***	**0.778**
19	heart-hungarian-1	0.739	0.704	0.724	0.640	0.724	0.619	0.651	0.626	**0.807**	***0.894***
20	heart-hungarian-2	0.758	0.762	0.787	0.778	0.779	0.752	0.760	0.794	**0.890**	***0.987***
21	breast-cancer-nki	0.512	0.555	0.541	0.565	0.548	0.562	0.536	0.519	***0.763***	**0.756**
22	horse-colic-1	0.604	0.771	0.754	0.731	0.763	0.701	0.707	0.716	**0.830**	***0.836***
23	horse-colic-2	0.744	0.834	0.845	0.852	0.840	0.781	0.772	0.764	***0.918***	***0.918***
24	haberman-survival	0.514	0.515	0.535	0.524	0.549	0.482	0.498	0.519	**0.642**	***0.650***
25	vertebral-column-2clases	0.779	0.796	0.758	0.759	0.763	0.753	0.742	0.747	**0.916**	***0.997***
26	mesothelioma	0.990	***1.000***	***1.000***	***1.000***	***1.000***	0.917	0.899	0.947	***1.000***	***1.000***
27	ionosphere	0.817	0.921	0.929	0.927	0.933	0.876	0.824	0.858	***0.982***	**0.980**
28	student-performance	0.510	0.606	0.611	0.621	0.619	0.600	0.537	0.602	**0.718**	***0.734***
29	congressional-voting	0.519	0.567	0.522	0.559	0.583	0.585	0.581	0.517	**0.624**	***0.633***
30	las-vegas-trip	0.496	0.489	0.482	0.452	0.465	0.448	0.473	0.455	**0.665**	***0.676***
31	cylinder-bands-1	0.547	0.585	0.587	0.582	0.589	0.556	0.581	0.601	**0.850**	***0.854***
32	cylinder-bands-2	0.555	0.611	0.615	0.595	0.612	0.552	0.550	0.572	***0.900***	**0.885**
33	breast-cancer-wisc-diag	0.956	0.953	0.953	0.943	0.950	0.893	0.919	0.920	***0.991***	**0.990**
34	ilpd-indian-liver	0.536	0.563	0.564	0.573	0.555	0.549	0.569	0.559	***0.740***	**0.738**
35	climate-model-simulation-crashes	0.926	0.994	0.988	0.993	0.991	0.955	0.946	0.890	***1.000***	***1.000***
36	credit-approval-1	0.835	0.811	0.834	0.829	0.836	0.808	0.791	0.792	**0.929**	***0.938***
37	credit-approval-2	0.821	0.846	0.847	0.826	0.850	0.779	0.767	0.799	**0.930**	***0.939***
38	statlog-australian-credit-1	0.504	0.561	0.535	0.509	0.545	0.504	0.530	0.511	**0.610**	***0.626***
39	statlog-australian-credit-2	0.515	0.545	0.546	0.538	0.541	0.522	0.513	0.508	***0.610***	**0.606**
40	breast-cancer-wisc	0.958	0.965	0.962	0.954	0.958	0.937	0.942	0.947	***0.994***	**0.993**
41	cervical-cancer	0.529	0.877	0.767	0.829	0.769	0.740	0.776	0.664	***0.970***	**0.961**
42	pima	0.658	0.727	0.722	0.661	0.722	0.638	0.683	0.649	***0.831***	**0.826**
43	oocytes-trisopterus-nucleus-2f	0.690	0.668	0.674	0.668	0.675	0.642	0.656	0.626	**0.899**	***0.900***
44	tic-tac-toe	0.974	0.954	0.925	0.927	0.920	0.828	0.823	0.824	***1.000***	***1.000***
45	mammographic	0.773	0.756	0.784	0.756	0.783	0.777	0.775	0.743	**0.833**	***0.842***
46	statlog-german-credit-1	0.508	0.575	0.594	0.558	0.591	0.549	0.564	0.563	**0.684**	***0.690***
47	statlog-german-credit-2	0.538	0.655	0.654	0.654	0.659	0.641	0.611	0.628	**0.795**	***0.799***
48	oocytes-merluccius-nucleus-4d	0.709	0.753	0.749	0.717	0.748	0.661	0.660	0.648	**0.862**	***0.864***
49	hill-valley	---	0.524	**0.546**	0.516	0.532	0.529	0.531	0.542	0.539	***0.673***
	Avg. AUC	0.677	0.726	0.721	0.713	0.723	0.687	0.683	0.680	**0.826**	***0.852***
	Avg. rank	7.673	4.786	4.878	5.714	4.786	7.724	8.010	7.959	**2.010**	***1.459***

1st rank	**2nd rank**	3rd rank

addition, analysis of the models used by each test instance of ICE shows an interesting bimodal distribution: most of the test instances (254 out of 286 cases) use less than 10 models (mostly $local$ models); in contrast, a few instances (32 cases) use more than 10 unique models (including both $local$ and $remote$ models) (Figure 4b), which are presumably the more difficult instances that are hard to be clustered and/or classified.

Comparing the performance difference on instances in the two peaks in Figure 4b, we can see that ICE has a much lower prediction error when compared to the'$whole$' model on instances with $>$10 models by ICE (Figure 4c). The t-test p-value of the error differences between the'$whole$' model prediction and ICE prediction on instances with $>$10 models (32 cases) is very significant (3.34×10^{-11}). Taking together, different instances should be treated differently on this dataset, and the ICE algorithm shows a potential way of separating and treating these different instances.

4 Conclusion

Based on the intuition that classifiers generated from different subdomains of training instances are needed in classification task, we proposed ICE, a novel multiple classifier generation and combination framework, which generally increases the diversity among submodels, and successfully associates the submodels to subdomains of instances. Evaluation results on 49 benchmarks show that our model has a stable improvement on a significant proportion of datasets over multiple existing MCS methods. We believe that ICE can provide a novel choice of utilizing subdomain models to improve classification.

References

[Breiman, 1996] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[Breiman, 2001] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[Cruz *et al.*, 2015] Rafael MO Cruz, Robert Sabourin, George DC Cavalcanti, and Tsang Ing Ren. Meta-des: A dynamic ensemble selection framework using meta-learning. *Pattern recognition*, 48(5):1925–1935, 2015.

[Cruz *et al.*, 2018] Rafael MO Cruz, Robert Sabourin, and George DC Cavalcanti. Dynamic classifier selection: Recent advances and perspectives. *Information Fusion*, 41:195–216, 2018.

[Fernández-Delgado *et al.*, 2014] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems. *J. Mach. Learn. Res*, 15(1):3133–3181, 2014.

[Freund and Schapire, 1995] Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer, 1995.

[Gashler *et al.*, 2008] M. Gashler, C. Giraud-Carrier, and T. Martinez. Decision tree ensemble: Small heterogeneous is better than large homogeneous. In *2008 Seventh International Conference on Machine Learning and Applications*, pages 900–905, Dec 2008.

[Giacinto and Roli, 1999] Giorgio Giacinto and Fabio Roli. Methods for dynamic classifier selection. In *Image Analysis and Processing, 1999. Proceedings. International Conference on*, pages 659–664. IEEE, 1999.

[Giacinto and Roli, 2001] Giorgio Giacinto and Fabio Roli. Dynamic classifier selection based on multiple classifier behaviour. *Pattern Recognition*, 34(9):1879–1881, 2001.

[Jacobs *et al.*, 1991] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[Jahid *et al.*, 2014] Md Jamiul Jahid, Tim H Huang, and Jianhua Ruan. A personalized committee classification approach to improving prediction of breast cancer metastasis. *Bioinformatics*, 30(13):1858–1866, 2014.

[Jordan and Jacobs, 1994] Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

[Ko *et al.*, 2008] Albert HR Ko, Robert Sabourin, and Alceu Souza Britto Jr. From dynamic classifier selection to dynamic ensemble selection. *Pattern Recognition*, 41(5):1718–1731, 2008.

[Kuncheva, 2000] Ludmila I Kuncheva. Clustering-and-selection model for classifier combination. In *Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 2000. Proceedings. Fourth International Conference on*, volume 1, pages 185–188. IEEE, 2000.

[Kuncheva, 2004] Ludmila I Kuncheva. *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2004.

[Kurzynski *et al.*, 2016] Marek Kurzynski, Maciej Krysmann, Pawel Trajdos, and Andrzej Wolczowski. Multiclassifier system with hybrid learning applied to the control of bioprosthetic hand. *Computers in biology and medicine*, 69:286–297, 2016.

[Oza and Tumer, 2008] Nikunj C Oza and Kagan Tumer. Classifier ensembles: Select real-world applications. *Information Fusion*, 9(1):4–20, 2008.

[Rokach, 2010] Lior Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2):1–39, 2010.

[Tong *et al.*, 2006] H. Tong, C. Faloutsos, and J. y. Pan. Fast random walk with restart and its applications. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 613–622, Dec 2006.

[Van De Vijver *et al.*, 2002] Marc J Van De Vijver, Yudong D He, Laura J Van't Veer, Hongyue Dai, Augustinus AM Hart, Dorien W Voskuil, George J Schreiber, Johannes L Peterse, Chris Roberts, Matthew J Marton, et al. A gene-expression signature as a predictor of survival in breast cancer. *New England Journal of Medicine*, 347(25):1999–2009, 2002.

[Vilalta *et al.*, 2003] Ricardo Vilalta, M-K Achari, and Christoph F Eick. Class decomposition via clustering: a new framework for low-variance classifiers. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 673–676. IEEE, 2003.

[Wang *et al.*, 2005] Yixin Wang, Jan GM Klijn, Yi Zhang, Anieta M Sieuwerts, Maxime P Look, Fei Yang, Dmitri Talantov, Mieke Timmermans, Marion E Meijer-van Gelder, Jack Yu, et al. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *The Lancet*, 365(9460):671–679, 2005.

[Woloszynski *et al.*, 2012] Tomasz Woloszynski, Marek Kurzynski, Pawel Podsiadlo, and Gwidon W Stachowiak. A measure of competence based on random classification for dynamic ensemble selection. *Information Fusion*, 13(3):207–213, 2012.

[Wolpert, 1992] David H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.

[Woods *et al.*, 1997] Kevin Woods, W. Philip Kegelmeyer, and Kevin Bowyer. Combination of multiple classifiers using local accuracy estimates. *IEEE transactions on pattern analysis and machine intelligence*, 19(4):405–410, 1997.