Dynamic Ensemble Associative Learning

Md Rayhan Kabir
Alberta Machine Intelligence Institute
University of Alberta
Edmonton, Canada
rayhan.kabir@ualberta.ca

Osmar R Zaiane
Alberta Machine Intelligence Institute
University of Alberta
Edmonton, Canada
zaiane@ualberta.ca

***Abstract*—Associative classifiers have shown competitive performance with state-of-the-art methods for predicting class labels. In addition to accuracy performance, associative classifiers produce human readable rules for classification which provides an easier way to understand the decision process of the model. Early models of associative classifiers suffered from the limitation of selecting proper threshold values which are dataset specific. Recent work on associative classifiers eliminates that restriction by searching for statistically significant rules. However, a high dimensional feature vector in the training data impacts the performance of the model. Ensemble models like Random Forest are also very powerful tools for classification but the decision process of Random Forest is not easily understandable like the associative classifiers. In this study we propose Dynamic Ensemble Associative Learning (DEAL) where we use associative classifiers as base learners on feature subspaces. In our approach we select a subset of the feature vector to train each of the base learners. Instead of a random selection, we propose a dynamic feature sampling procedure which automatically defines the number of base learners and ensures diversity and completeness among the subset of feature vectors. We use 10 datasets from the UCI repository and evaluate the performance of the model in terms of accuracy and memory requirement. Our ensemble approach using the proposed sampling method largely decreases the memory requirement in the case of datasets having a large number of features and this without jeopardising accuracy. In fact, accuracy is also improved in most cases. Moreover, the decision process of our DEAL approach remains human interpretable by collecting and ranking the rules generated by the base learners predicting the final class label.**

***Index Terms*—Associative classifier, Ensemble Learning, Explainable Ensemble**

I. Introduction

With the vast increase of the application of machine learning techniques in real life scenarios, the importance of classification is steadily and assuredly increasing. Classification is a method of analyzing the data and learning to label them into different classes. Currently a huge number of classification algorithms are available. Many researchers performed in depth studies to compare and analyze the performance of classifiers. Among them the study of Fernández-Delgado, Manuel, et al. [1] is mention-worthy where they analyzed the performance of 179 classifiers using 121 different datasets. Associative classifiers, however, remain little known and unexplored. Moreover, most of the classifiers work in a black box fashion, that is, after training the model, the model remains opaque without revealing what has actually been learned. For instance, even though amazingly accurate, neural networks, after convergence of the edge weights and node biases, are not interpretable. In addition, during inference, they do not provide necessary explanation for the prediction, which becomes troublesome in some application fields like medical diagnosis, financial decision making, etc. Since Machine learning is becoming popular and ubiquitous, explanation is now even legislated in many jurisdictions. This has lead to many effort attempting to explain salient features in deep learning, but also a return to rule-based classifiers. Associative classifiers are among them. Associative classifiers use an association rule mining approach [3] to discover frequent patterns, and lately statistically significant patterns, from which to derive classification rules. The test data is then classified using the rules that are generated during the training phase. A rule is in the form of $X \rightarrow Y$ where X, the antecedent, is a conjunction of co-occurring features, and Y, the consequent, is a class label. Several associative classifiers have been proposed namely CBA [2], ARC [4] , CMAR [5] , CPAR [6]. Rules generated by the associative classifiers are simple and human readable thus their predictions can be interpreted. Though the associative classifiers are competitive with the state of the art classifiers and have a huge advantage of built-in explainability, they suffer from the limitation imposed by the required threshold tuning, namely support and confidence, which are dataset specific.

To solve this issue, Li and Zaiane [8] proposed SigDirect where they used the Kingfisher algorithm [9] to find statistically significant classification rules. Sood and Zaiane [10] proposed SigD2, a classifier based on SigDirect with a two stage rule pruning strategy which removes the noisy rules. The improvement is promising as it reduces the number of rules without compromising the accuracy. In fact it was shown that SigD2 outperforms other rule-based classifiers as well as classical classifiers such as SVM, Bayesian, C4.5 and even simple neural networks [10]. However, the performance of both SigDirect and SigD2 is limited in terms of required memory and run-time if the feature vector of the feature space is large. An Ensemble of classifiers, each trained on a subset of the feature space, could address this issue. However, one has to fixed the size of the ensemble. In this paper we propose Dynamic Ensemble Associative Learning (DEAL) to improve the efficiency of SigD2 in terms of memory requirement and runtime without hampering the accuracy. As a base learner, we considered SigD2 as Sood and Zaiane [10] showed SigD2

IEEE/ACM ASONAM 2022, November 10-13, 2022
http://dx.doi.org/10.1145/XXXXXXX.XXXXXXX
978-1-6654-5661-6/22/$31.00 © 2022 IEEE

outperforms other associative classifiers. In our model, we added a dynamic feature sampling technique inspired from the work of Cao et al. [11] which eliminates the necessity of defining the number of base learners for the ensemble model and ensures diversity and coverage of the feature space among the base learners. We first sampled the features randomly to form a subset of the feature vector. After sampling, we calculate the overlap of the new subset with the previously generated subsets. If the overlap of the new subset is high we discard the subset and start another sampling. We stop the sampling procedure either if after subsequent sampling, we do not find any more subset which has low overlap with the previous subsets or all the features are covered by the generated subsamples. We train one SigD2 model with each of the subsamples and perform max-vote among the class label predictions. The class label with the highest vote is assigned to the new test instance. The main contribution of our work are:

- Our proposed model DEAL significantly decreases memory requirement and the runtime in comparison with SigD2 without affecting the accuracy for dataset with large feature vector,
- We proposed a feature sampling technique which automatically determines the number of base learner in an associative classifier bagging approach ensuring diversity and feature space coverage.

The following sections are organized as follows: Section 2 provides the necessary background and related works, Section 3 describes the methodology that we followed in our experiment, Section 4 is the evaluation and analysis of the performance of DEAL, and finally Section 5 is the conclusion with some future research direction of the work.

II. Related Works

While still little known, Associative Classifiers have attracted the attention of many researchers for many years, and are unfortunately now overshadowed by the very successful deep learning approaches. Associative Classifiers hinge on one of the canonical tasks in data mining, association rule mining introduced by Agarwal and Srikant's algorithm Apriori [3]. Liu et al. [2] showed it is possible to use association rule mining to build a classifier. Their proposed Classification Based on Association (CBA) uses an Apriori based approach to generate class association rules. In their approach, all constrained association rules from the database are generated and ranked based on metrics. The highest matching rule is used for the classification task. Their subsequent work [12] attempts to decrease noisy rules. They improved the performance of the classifier by choosing the most accurate class association rules for the classification task. Being inspired from this work many researchers came forward with different approach to improve the performance of associative classifier, mainly differentiating themselves by proposing different strategies to select the applicable rules during inference. Yin and Han [6] proposed Classification based on Predictive Association Rules (CPAR). They generate association rules directly from the training data by a greedy approach. To evaluate each of the generated rules, they use an expected accuracy and finally select the best k rules for the classification task. Li et al. [5] proposed CMAR which is Classification based on Multiple Association Rules. They extended FP-growth [7] frequent pattern mining method to construct an FP-tree for class distribution association. In this approach the class association rules are stored in a special data structure. The generated rules are pruned based on the confidence, correlation and database coverage. Using multiple strong association rules with a Weighted χ^2 measure, a datapoint is labeled to the appropriate class. Antonie and Zaiane. [4] propose another Association rule based classifier that they apply for text categorization. They put forward two different approaches; ARC-AC, considering all generated rules from the whole training set, and ARC-BC, where data from each class is mined separately allowing the handling of unbalanced datasets. Indeed, being based on frequency, minority classes risk not being expressed in a predictive model when all dataset entries are mined together. Another approach, CCCS, proposed by Arunasalam and Chawla [14] introduces a measure named "Complement class support" (CCS). The authors claim CCS guarantees a positive correlation among the class label and the generated rules. Antonie et al. [15] also propose a two stage associative classifier where associative classification rules are discovered in a first stage and in a second stage, another algorithm learns how to use those rules for class prediction. Instead of basing the selection of rules to apply during inference on some heuristics, they use a neural network to learn how to predict the best rules to apply. This approach showed improved efficiency in terms of accuracy.

One limitation of associative classifiers is the generation and the need to evaluate a huge number of rules, among them many are noisy rules. To overcome this problem Zaiane and Antonie [17] performed an extensive study with the focus of reducing the number of rules without decreasing the accuracy of the classifier. The authors propose a new pruning strategy to reduce the number of class association rules. They also propose different heuristics for selecting rules which can obtain high accuracy for a given instance. However, it remains that the proper support and confidence values have to be selected and tuned. This is one major drawback of associative classifiers inherited from association rule mining. The performance of the model largely depends on these values. It is a tedious task to find the proper confidence and support values for each of the dataset, hampering the adoption of associative classifiers. To solve this issue, Li and Zaiane [8] propose SigDirect where they improvised the kingfisher algorithm [9] to find the statistically significant rules for classification, instead of frequent ones. Their proposed method increases the accuracy of the classifier. The main contribution of their work is eliminating the necessity of the annoying support and confidence thresholds. However, SigDirect also has some limitation. Sood and Zaiane [10] show that noisy rules can be further eliminated proposing in SigD2 a two stage pruning technique which can reduce the number of rules without jeopardising the accuracy of the classifier and therefore making a learned model even more practically interpretable.

Another noteworthy rule-based classifier is RIPPER proposed by WW Cohen [13]. RIPPER forms classification rules in the process of rule growing and pruning. During the growing phase, rules are generated in a very restrictive manner and less restriction is put on the rules during pruning to avoid overfitting. This model, while not based on association rules, also shows very competitive result.

As we stated earlier, SigDirect and SigD2, suffer from the limitation of large memory requirement and run time if the feature vector of the training dataset is large. To solve this, we propose an ensemble method using SigD2 as base learner since ensemble based classifiers can boost the accuracy of classification models by combining the results of the base learners. Here base learners can be applied on different subspaces for the input feature space. In an extensive study, Bauer and Kohavi [18], show empirically that the ensemble model can enhance the performance of a classifier for most of the classification tasks.

III. Methodology

In this section we describe our methodology for the bagging approach using SigD2 as a base learner. As mentioned above, SigD2 was shown to outperform other rule based classifiers and certainly all other associative classifiers in terms of accuracy and number of generated classification rules. In most cases of bagging, the dataset instances are randomly sampled. We instead sample the feature space and use all instances. In most known ensemble approaches, the number of classifiers in the ensemble is predetermined and fixed. Therefore, initially we start proceeding with the same and randomly sampling the feature space for each classifier in the ensemble. This random sampling does not guarantee completeness as some features may never be picked, and does not ensure diversity among the classifiers since different classifiers in the ensemble may end-up with exactly the same features selection. For that reason we later introduce our dynamic sampling technique that ensures completeness and diversity and automatically determines the necessary number of learners in the ensemble. The procedures are explained in detail in the next sub sections.

A. *Ensemble with random subsample*

At first, we use 100 base learners in the ensemble model. Each of the base learners is trained on a random subsample of size N of the original feature space. We show the process of creating random subsamples in Algorithm 1. Following this method, we generate 100 subsamples and train the base learners with each of the subsamples. During inference, the prediction of the class label is done with each of the base learners and a vote determines the final prediction.

Generating subsample in this way presents some limitation. A feature can be selected every time or some of the features might not be selected at all. Thus many important features which are strongly correlated with the class label can be excluded from the training process. Another limitation is the need to fix the number of base learners, in our case 100, based on common practice in the literature. If the number of features is not large enough, there is no need for 100 base learners. Indeed many base learners would be trained on the same subsample of the feature space. To solve these issues we propose a new feature sampling technique which addresses both of the limitations.

Algorithm 1 Subsample generation by randomly selecting features

Input: features: all features of the feature space; **N:** Number of the features in each subsample
Output: 100 subsample of the feature space

```
1: all_subsample ← []
2: for i in range(100) do:
3:     new_subsample ← []
4:     n ← 0
5:     reset features
6:     while n < N
7:         f ← randomly select without replacement a feature from features
8:         add f to new_subsample
9:         $n \leftarrow n + 1$
10:    end while
11:    add new_subsample to all_subsamples
12: end for
13: return all_subsamples
```

B. *Dynamic Ensemble Associative Learning*

Our goal is to make sure all input features are used by at least one base learner, and the set of base learners are diverse. Therefore, to subsample from the feature vector, we follow a sequential procedure until all features are selected (ensuring completeness). To ensure diversity, we make sure the feature spaces of any pair of base learners do not overlap more than some fraction threshold. Our method to generate diverse subsamples is provided in Algorithm 2 and Algorithm 3. In the input of the algorithm, N is the number of features to be selected in each of the subsamples. From previous literature we know that if the feature vector of the dataset is large, SigD2 requires high volume of memory and run time. Therefore we limited the size of the feature vector using N. In our experiments, we have tested across different values for N but in all cases we used small value of N. The next parameter of the algorithm is O_v. O_v is the maximum allowed overlap between any two subsamples. The value of O_v ranges from 0 to 1 where 0 indicates no overlap between any two subsamples and 1 indicates features of two subsamples can be identical. If we put a very small value close to 0 for O_v, all the subsamples would be almost unique and the number of generated subsamples small, which results in a small number of base learners, defeating the purpose of the ensemble. If we use a value close to 1, the subsamples would be almost similar to subsamples generated using a random subsampling. Thus we have to find an optimum value for O_v. The last parameter is T which indicates how many times we will try to get a satisfactory new subsample with our required overlap. This is to avoid blocking infinite loops. When generating a subsample by selecting features, if there is at least one previously generated subsample with which overlap of the selected features of new subsample is more than O_v, we discard the current subsample and generate a new one. T provides a boundary for how many times we try for a single subsample. If after trying T times we do not find any subsample with less than overlap O_v, we stop the sampling procedure. We also keep track of whether all the features are already covered by the subsamples or not. If all the features in feature space are covered by the generated subsamples, we stop our sampling procedure.

Algorithm 2 Algorithm **GenerateSubsample**

Input: Features: all features in the feature space; **N:** Number of the features in each subsample, O_v**:** Maximum overlap between any two subsamples, **T:** maximum number of tries to generate a subsample
Output: Subsamples of the feature space

```
1:  try=0;
2:  all_subsamples ← []
3:  while try < T do
4:     new_subsample ← subsampleGen(Features, N)
5:     if(Overlap(new_subsample)< $O_v$) then
6:        Add new_subsample to all_subsamples
7:        try ← 0
8:     else
9:        try ← try+1
10:    end if
11:    if (all_feature_covered(all_subsample) ==True) then
12:       break
13:    end if
14: end while
15: return all_subsamples
```

Algorithm 3 shows how a subsample is generated. Here we randomly select a feature from the feature space and add it to the new subsample. N indicates the number of features in each subsample.

Algorithm 3 Algorithm **SubsampleGen**

Input: Features: all features in the feature space; **N:** Number of the features in each subsample
Output: subsample of the feature space

```
1: subsample← []
2: n← 0
3: while n < $N$ do
4:    new_feature ← randomly select a feature
5:    Add new_feature to subsample
6:    n ← n+1
7: end while
8: return subsample
```

After generating a new subsample we evaluate whether the features of the newly generated subsample has more overlap than the O_v ratio with any of the previously generated subsamples. Between two subsample S_1 and S_2 where the features are F_{S_1} and F_{S_2} respectively and the number of features in each of the subsamples is N then the overlap between the subsamples is calculated using the following equation:

$$Current\_overlap = \frac{F_{S_1} \cap F_{S_2}}{N} \tag{1}$$

If there is at-least one previously generated subsample with which the feature overlap is greater than O_v, we discard the subsample. If there is no previous subsample with which the newly generated subsample has overlap greater than O_v, we consider the current subsample satisfies the required condition and accepts the sample. After generating the subsamples, we train one SigD2 model with each of the generated subsamples. The size of the output set of subsamples determines the size of the ensemble. We use the trained base learners to predict the class label of the test dataset. The class label with the highest vote of the base learners is determined as the final predicted class label. We evaluate the model using the predicted value in terms of accuracy, memory requirement and runtime.

IV. Result Analysis

We use 10 different UCI datasets [19] to test our model. Before using a dataset we discretize the numerical values as stated in [20]. We convert the features to a binary feature vector. We used the same vector form of discretized values for all our experiments will all contenders. Thus the results of mentioned algorithms can be slightly different from their original papers. For each of the datasets, we use 80% of the data as training data and the rest 20% as test data. We compare the result of our model DEAL with different values of N which is the number of features for each base learner, against the result of C4.5 [21], Random Forest [22], RIPPER, SigD2 and the ensemble using 100 random sampling of features, we denote as RS. We also experiment with an ensemble of C4.5 as base learner using our proposed feature sampling method DEAL.

The reported result is the average of 20 runs for each of the datasets. We compare the performance of our proposed model in terms of accuracy, memory requirement and runtime. We also provide an analysis on the performance of DEAL for different values of O_v. Finally, we discuss the interpretability of the prediction by DEAL.

A. Classification Accuracy

We compare the performance of our proposed method with SigD2 since it was demonstrated to have better performance than other associative classifiers, rule-based classifiers [8], [10]. We include RIPPER, however, since it is a rule induction algorithm different from association rules. We also compared the performance of DEAL with C4.5 and an ensemble of C4.5. As a representative of the traditional ensemble approaches, we use Random Forest [22] to compare our result. In the case of random sampling of the features (RS), we experiment with different values of N (i.e. number of features in each subsample) and different values of O_v (i.e. maximum overlap among the selected features of base learners). From the experiment, we find the optimum result when the value of N is between 25 and 30.

The comparison in accuracy of C4.5, ensemble of C4.5 using proposed sampling method, Random Forest (RF), RIPPER, SigD2, Random feature Sampling method (RS) with DEAL using different values of N are provided in Table I. From the table we can see, among the datasets, in comparison with Random Forest and C4.5, Random Forest has better performance than DEAL in 4 of the datasets while DEAL is a close runner-up, and in 4 others DEAL has better performance, with one ex aequo with the zoo dataset. In one dataset RIPPER performed better than DEAL. On average DEAL is better. In comparison with the original SigD2, we can see that apart from the anneal dataset, DEAL presents an improvement for the accuracy. The probable cause might be that some of the feature vectors in the anneal dataset being highly correlated which plays an important role in the class prediction. DEAL selects a subset of the feature vectors thus the feature vectors that are needed to be together get separated. Further investigation is required for this dataset to find how the important features can be kept together. That might improve the performance of DEAL in those datasets where some feature vectors together play an important role in class prediction. The random sampling of the features with 100 base learners could not perform at the same level as DEAL. In comparison with SigD2 sometimes it improves the accuracy and in some cases the accuracy is decreased. As the random sampling is completely random, many important features might not be selected for any of the base learners at all. This is the reason of the poor performance of RS. On the average, DEAL with a feature vector size $N = 30$ and $O_v = 0.6$ has the best performance. We also compare the performance of DEAL with other models in terms of F1 score. Figure 1 shows the F1 score of the models for each dataset. From the experiments, We find DEAL, either with $N = 25$ or $N = 30$, has very competitive F1 score compared to other models.

We can get another interesting observation from Table I. With the increase of N, the number of features for each of the base learners, the average accuracy increases. For better understanding of this, we experimented with different values of N. The average accuracy for different values of N is plotted in Figure 2. We can see a sharp increase when the value of N is increased from 15 to 25. After 25, the average accuracy continues to increase but not with the same steep rate.

There is a compromise to make with the required memory and runtime, particularly due to the aforementioned limitations of the used base learner SigD2. Therefore, we chose not to increase the number of features per base learner more than 30. We are also interested to know the number of base learners for each of the dataset as we mentioned the number of base learners is determined automatically by DEAL. Table II provides the number of base learners for each of the dataset. These are significantly less than the typical 100 or even 50 used in the literature for classical ensembles.

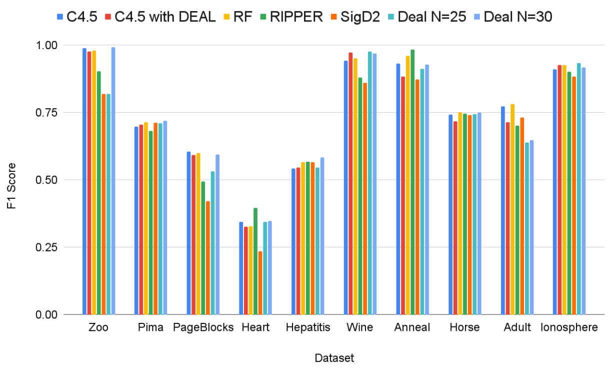

Fig. 1: F1 score of C4.5, C4.5 using sampling of Deal, Random Forest (RF), RIPPER SigD2, and DEAL.

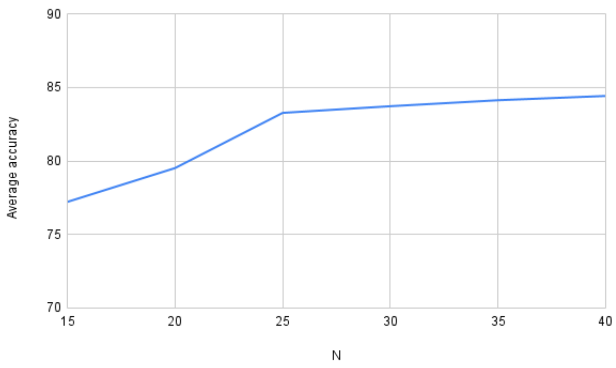

Fig. 2: Average accuracy of DEAL for different values of N(Number of selected feature for each base learner)

To understand the effect of different values of O_v we also experiment with values starting from 0.1 to 0.9 increasing 0.1 in each step. The result is provided in Figure 3. From the figure we can see, the performance of the model is best when the value of O_v is in the middle. Because a small value results in almost unique base learners and a very small number of base learners. In the case of a large value of O_v, the results would allow repetitions of base learners like with simple random sampling procedure. Thus in both cases the performance is affected. We can see good performance when the value of O_v is between 0.4 to 0.6 for N=30 and 0.5 to 0.7 when N=25 for the dataset we used in our experiments. Thus overlap between 0.4 to 0.7 provides optimum result for most cases.

To understand whether the improvement in accuracy by DEAL is significant or not, we performed a statistical significance test. We carried-out a student paired t-test with the null hypothesis, that the improvement in the accuracy of DEAL is not significant. We run the whole experiment 20 times and recorded the average accuracy each time for each model. For DEAL we used $N = 30$, $O_v = 0.6$. The difference in the accuracy in each pair is calculated by Equation 2.

Dataset	#cls	#rec	Feature vector Size	C4.5	C4.5 with DEAL	RF	RIPPER	SigD2	RS N=25	RS N=30	DEAL N=20 O$_v$ = .6	DEAL N=25 O$_v$ = .6	DEAL N=30 O$_v$ = .6
Zoo	7	101	35	95.24	95.24	**97.62**	85.48	94.55	85.53	85.53	91.66	92.85	**97.62**
Pima	2	768	36	76.62	73.38	67.41	76.08	76.44	77.21	79.81	75.26	**79.82**	77.61
PageBlocks	5	5473	41	93.15	92.23	**93.51**	80.37	91.67	90.04	91.23	84.16	91.84	88.53
Heart	5	303	47	50.82	52.45	52.45	**64.9**	45.9	46.86	48.51	47.54	49.18	49.18
Hepatitis	2	155	54	70.97	74.19	78.38	78.72	81.93	80.65	81.29	78.06	80.64	**82.58**
Wine	3	178	65	91.67	91.67	77.78	93.33	92.7	91.57	93.26	86.51	**94.38**	92.13
Anneal	6	898	67	97.22	86.67	**98.89**	97.5	92.22	84.41	88.86	80.51	83.85	92.2
Horse	2	368	83	78.38	75.67	75.67	80.56	78.8	70.38	81.25	80.43	**82.33**	81.79
Adult	2	48842	95	84.06	82.31	**84.97**	83.91	84.59	79.67	81.81	80.52	84.92	81.51
Ionosphere	2	336	155	88.73	**95.78**	**95.78**	92.81	90.18	88.1	90.77	90.47	92.85	94.04
Average				82.686	81.96	82.25	82.77	82.89	79.44	82.23	79.512	83.27	**83.72**

TABLE I: Accuracy of C4.5, Ensemble of C4.5 using feature sampling of DEAL, Random Forest (RF), RIPPER, bagging using random sampling (RS), SigD2, and DEAL.

Dataset	#Base Learner	Dataset	#Base Learner
Zoo	4	Wine	13
Pima	4	Anneal	12
PageBlocks	8	Horse	19
Heart	7	Adult	21
Hepatitis	10	Ionosphere	45

TABLE II: Number of base learners using $N = 30$ and $O_v = 0.6$

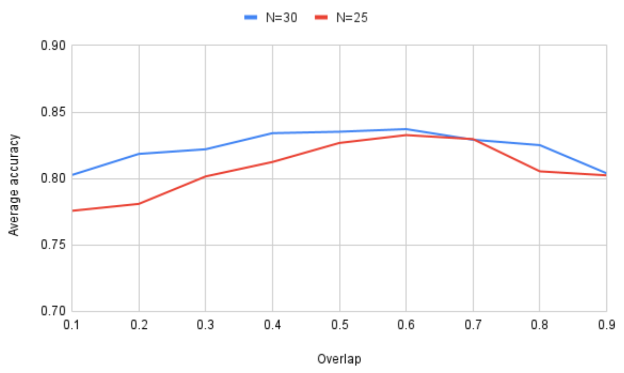

Fig. 3: Average accuracy across different values of Overlap (O_v)

$$\delta = \mathcal{M}_{DEAL}(test\_data) - \mathcal{M}_i(test\_data) \qquad (2)$$

Here, $\mathcal{M}_{DEAL}$ is our proposed model and the subscript i stands for the other models. Running the models on test data provides the accuracy of the models and δ is the difference in the accuracy.

We performed the paired t-test followed by the work of Henry et al. [23] for pairwise comparison among the models and calculated the p-value. If the calculated p-value is less than the threshold alpha(0.05) then we can say the improvement by the proposed model is significant. The calculated p-values are provided in Table III.

From Table III we can see the p-value for each pair is less than the threshold alpha(0.05) except for DEAL vs RIPPER. Thus, except RIPPER, for the other models we can say, the improvement in accuracy of our proposed over other models is significant.

Algorithm	p-value
DEAL vs C4.5	0.0091
DEAL vs C4.5 ensemble	4.4e-05
DEAL vs Random Forest	0.0077
DEAL vs SigD2	0.0001
DEAL vs RIPPER	0.087

TABLE III: Statistical result

B. *Memory Requirements*

One of the main goals of DEAL is to reduce the memory requirements for datasets having large feature vector size. In previous sections we showed that DEAL does not reduce the accuracy in comparison to the original SigD2 rather in almost all datasets, it increases the accuracy. Using DEAL we can reduce the memory requirements for datasets having a large number of features. The memory requirements of C4.5, ensemble of C4.5 using proposed feature sampling method, Random Forest, SigD2 and DEAL using N as 25 and O_v as 0.6 are shown in Table IV. Here we did not provide memory requirement using the random sampling procedure as for any given number of feature vectors in base learners, the memory requirement of SigD2 is the same. In the memory requirements, the reported result for DEAL is the highest memory used by a base learner in the process as we run the base learners sequentially. In the case of Random Forest, the trees in the forest are executed in parallel. Thus the memory requirement reported here is the memory required for running 100 decision tree in parallel.

Running base learners in parallel has a huge advantage for runtime, which we discuss in the next subsection. Further studies are required to understand whether this trade-off between memory requirement and runtime is feasible by executing the base learners of DEAL in parallel. From Table IV we can see, in 8 datasets out of 10 DEAL has the least memory requirement among the contenders. In adult dataset RIPPER requires the least memory. In Heart dataset, DEAL and RIPPER has the same performance which is the least

Dataset	C4.5	C4.5 with Deal	Random Forest	RIPPER	SigD2	DEAL
Zoo	106	108	127	107	121	**105**
Pima	103	109	128	102	102	**101**
PageBlocks	111	117	134	111	112	**110**
Heart	105	112	129	**104**	107	**104**
Hepatitis	103	111	127	102	113	**100**
Wine	104	112	128	105	165	**100**
Anneal	107	113	129	107	150	**106**
Horse	109	112	128	109	235	**106**
Adult	195	211	252	**124**	257	252
Ionosphere	124	117	128	112	2519	**108**

TABLE IV: Memory requirement(in Megabyte) by contenders and DEAL($N = 30$ and $O_v = 0.6$)

memory requirement in comparison to other algorithms. The memory requirement in SigD2 increases with the increase in features. One main reason behind this is when working with a large size feature vector, SigD2 has to go through a huge number of class association rules (CAR). CAR increases with the increase of the feature vector size. But in our approach, with a dataset with large feature vector size, the number or rules for each of the base learners is reduced dramatically. In case of a dataset having very large size of feature vector, our proposed sampling method decreases the memory requirement to a significant extent.

C. Runtime

We also measure the runtime of DEAL for the same datasets in comparison to Random Forest and SigD2. The runtime of the methods are provided in Table V. In this table we do not include random sampling (RS) as in random sampling we have to train 100 base learners in a sequential manner which would always require a longer runtime than DEAL. From Table V, we can see Random Forest is always the fastest except for Zoo dataset where C4.5 is faster. Despite the fact that DEAL is an ensemble and we are not running the base learners in parallel, DEAL is still faster than SigD2 in many cases. SigD2 beats DEAL when the input feature space is small, because for datasets with a small feature space, DEAL still has to go through the sampling and for not having enough feature, DEAL has to sample many subsample and compare with the previously generated subsamples which increases the runtime. However, with larger feature vectors, DEAL outperforms SigD2.

D. Interpretability of the model

One of the major advantages of our ensemble model is that the prediction of the model is interpretable. After creating subsamples of the feature space and training the base learners with each of the subsamples, base learners uncover classification rules expressed in their respective feature subspace. When predicting the class label of an instance, base learners use their own rules. There are applicable rules, applicable to the instance case, and there are applied rules, effectively used for the decision by each base learner. The applied rules that agree with the majority vote are kept. Using DEAL, we can easily gather all the applied rules selected by base learners. Those would be part of the decision explanation. In addition, the set of applied rules and the applicable rules can be used to rank features by importance vis-à-vis the final prediction. Rules gathered from the base learners can be grouped according to the class label, and ranked based on their frequency among the base learners. Features are ranked by importance for a particular class label prediction. The more a feature appears in an applied rule, the more important it is. Features appearing in applicable rules get an additional boost of importance. Table VI gives a glimpse of the individual interpretable models learned by the different base learners. In the case of the Zoo dataset, there are 4 base learners. The learned models are sets of human readable classification rules with their strength measures, typically sorted by these measures). In Table VI we provide a subset of these rules and the feature subspace for each base learner. At inference time, each base learner finds the applicable rules for that instance and selects the rules to apply for the decision. Table VII shows the applicable rules per base learner for a given test instance as well as the decision for each learner before the vote. We can see that the consensus is for class 5 in this case and that there is agreement for the importance of features 16 and 28. What constitutes the decision explanation is the list of important features but also the list of applied rules as well as applicable rules with the final class label decision as their consequent.

Dataset	C4.5	C4.5 with Deal	Random Forest	RIPPER	SigD2	DEAL
Zoo	**0.06**	0.58	0.12	0.08	3.64	6.65
Pima	1.76	3.84	**0.15**	0.41	0.47	0.89
PageBlocks	21.04	35.83	**0.24**	0.46	10.23	17.86
Heart	0.69	3.04	**0.13**	0.40	8.55	3.11
Hepatitis	0.23	2.62	**0.11**	0.26	10.41	1.25
Wine	0.17	1.27	**0.11**	0.33	0.46	1.05
Anneal	2.1	9.46	**0.14**	0.56	7.99	5.38
Horse	0.92	6.24	**0.13**	0.53	32.49	2.97
Adult	13.2	19.8	**3.19**	171.56	263.12	167.16
Ionosphere	1.86	16.13	**0.12**	0.32	1905.44	6.22

TABLE V: Runtime (in Seconds) by contenders and DEAL ($N = 30, O_v = 0.6$)

V. Conclusion and future works

We propose Dynamic Ensemble Associative Learning (DEAL) where we use SigD2 as a base learner and present a feature sampling method which eliminates the need to fix the size of ensemble while ensuring diversity and feature space coverage. Evaluating the model over different datasets, reveals an increase of the accuracy. DEAL solves the limitation of SigD2 requiring huge memory and runtime if the feature vector space is large. Memory requirement and runtime of DEAL does not increase with the increase in feature vector like SigD2. Along with decreasing the runtime and memory requirement, we designed DEAL in such a way that the decision process of DEAL is human readable and explainable.

Base learner 1		**Base learner 2**	
Selected features	[0, 2, 4, 6, 9, 10, 11, 12, 14, 14, 15, 15, 16, 19, 19, 19, 19, 21, 21, 23, 24, 25, 26, 29, 30, 31, 33, 33, 34, 34]	**Selected Features**	[0, 1, 1, 3, 3, 3, 4, 4, 6, 6, 15, 15, 15, 18, 20, 20, 21, 23, 25, 25, 27, 27, 28, 29, 29, 31, 32, 32, 33, 34]
Rules	6, 23 -> 3;(0.1375,1.000,-29.980) 0, 25 -> 1;(0.1625,1.000,-33.384) .... 2, 34, 19 -> 0;(0.3500,0.966,-28.991) 31, 2, 33, 14 -> 6;(0.1125,0.750,-20.776)	**Rules**	27 -> 6;(0.0250,1.000,-4.475)) 18, 29 -> 6;(0.0750,1.000,-15.090) .... 33, 15, 21 -> 2;(0.0250,0.667,-5.185) 0, 29 -> 6;(0.1000,0.615,-14.797)
Base learner 3		**Base learner 4**	
Selected features	[1, 3, 6, 7, 7, 8, 10, 13, 14, 15, 16, 17, 19, 20, 21, 22, 24, 24, 24, 25, 27, 28, 29, 30, 30, 30, 31, 31, 32, 33]	**Selected features**	[0, 0, 1, 1, 1, 2, 3, 4, 5, 7, 8, 9, 9, 10, 11, 12, 12, 14, 16, 17, 17, 18, 20, 21, 22, 23, 28, 28, 30, 34]
Rules	27 -> 6;(0.0250,1.000,-4.475)) 3 -> 1;(0.1625,1.000,-33.384) .... 15, 6, 19 -> 4;(0.0375,0.600,-9.014) 10, 6, 17, 8 -> 2;(0.0375,0.600,-7.647)	**Rules**	7 -> 0;(0.4500,1.000,-52.636) 3 -> 1;(0.1625,1.000,-33.384) .... 0, 14, 17 -> 1;(0.1625,0.929,-30.745) 0, 16 -> 6;(0.1125,0.750,-20.776)

TABLE VI: Rules from the learned model in the form of "Antecedent -> Class label;(support, confidence, -ln(p-value))" where Antecedent is a conjunction of tokenised features. For interpretability, tokens are mapped back to features (attribute-value pairs)

Test instance: [1, 2, 5, 6, 9, 10, 12, 14, 16, 19, 20, 22, 28, 29, 31, 33]	
Vectorized test instance:[0,1,1,0,0,1,1,0,0,1,1,0,1,0,1,0,1,0,0,1,1,0,1,0,0,0, 0,0,1,1,0,1,0,1,0]	
Applicable rules by BL 1	**Applicable rules by BL 2**
9, 29 -> 5;(0.0250,1.000,-6.267) **16**, 12 -> 5;(0.0500,0.667,-11.566) 31, **16** -> 6;(0.1125,0.750,-20.776)	**28** -> 5;(0.0500,0.667,-11.566) 1, 20 -> 0;(0.4250,1.000,-45.694)
Predicted class label: 6	**Predicted class label: 5**
Applicable rules by BL 3	**Applicable rules by BL 4**
19, **28** -> 5;(0.0500,1.000,-14.274)	10, **28** -> 5;(0.0500,1.000,-14.274)
Predicted class label: 5	**Predicted class label: 5**

TABLE VII: Applicable rules from the generated rules by the base learners(**BL**) and decision process of DEAL

The sampling method of DEAL provides promising results when using SigD2 as a base learner since it ensures diversity and feature space coverage. Random Forest also uses a subset of the features and the subsets are chosen randomly. It would be interesting to use our sampling method in the case of Random Forest. This would eliminate the need to predetermine the number of estimators in Random Forest. As we have shown DEAL provides better result in the case of tabular data. In the future we would like to deploy DEAL on text and image data. DEAL is a framework where the base learner SigD2 could be replaced. A better rule based learner that is not as constrained by the dimensionality of the feature space could allow experimenting with a larger sampled feature subspace. Finally, we are considering investigating counter factual explanations given the availability of other applicable rules that lead to other class label then the one voted for by the ensemble.

References

[1] Manuel Fernandez-Delgado and Eva Cernadas and Senen Barro. "Do we need hundreds of classifiers to solve real world classification problems?." The Journal of Machine Learning Research 15.1 (2014): 3133-3181.

[2] Bing Liu and Wynne Hsu and Yiming Ma. "Integrating classification and association rule mining." Intl. Conf. on Knowledge Discovery and Data Mining. 1998.

[3] Rakesh Agrawal and Ramakrishnan Srikant. "Fast algorithms for mining association rules." Intl. Conf. very large data bases, VLDB. 1994.

[4] ML Antonie and Osmar R. Zaiane. "Text document categorization by term association." IEEE Intl. Conf. on Data Mining (ICDM). pp. 19–26 (2002)

[5] Wenmin Li and Jiawei Han and Jian Pei. "CMAR: Accurate and efficient classification based on multiple class-association rules." IEEE Intl. Conf. on data mining. 2001.

[6] Xiaoxin Yin and Jiawei Han. "CPAR: Classification based on predictive association rules." SIAM Intl. Conf. on data mining. 2003.

[7] Jiawei Han and Jian Pei and and Yiwen Yin. "Mining frequent patterns without candidate generation." ACM SIGMOD Record 29.2 (2000): 1-12.

[8] Jundong Li and Osmar R. Zaiane. "Exploiting statistically significant dependent rules for associative classification." Intelligent data analysis 21.5 (2017): 1155-1172.

[9] Wilhelmiina Wilhelmiina and Matti Nykänen. "Efficient discovery of statistically significant association rules." IEEE Intl. Conf. on data mining. 2008.

[10] Nitakshi Sood and Osmar R. Zaiane. "Building a competitive associative classifier." Intl. Conf. on Big Data Analytics and Knowledge Discovery. 2020.

[11] Peng Cao and Dazhe Zhao and Osmar R. Zaiane. "Cost sensitive adaptive random subspace ensemble for computer-aided nodule detection." IEEE Intl. Symp. on Computer-Based Medical Systems. 2013.

[12] Bing Liu and Yiming Ma and Ching Kian Wong. "Improving an association rule based classifier." European Conference on Principles of Data Mining and Knowledge Discovery. Springer, Berlin, Heidelberg, 2000.

[13] William W. Cohen "Fast effective rule induction." Machine learning proceedings 1995. Morgan Kaufmann, 1995. 115-123.

[14] Bavani Arunasalam and Sanjay Chawla. "CCCS: a top-down associative classifier for imbalanced class distribution." ACM SIGKDD Intl. Conf. on Knowledge discovery and data mining. 2006.

[15] Maria-Luiza Antonie and Osmar R. Zaiane and Robert C. Holte. "Learning to use a learned model: A two-stage approach to classification." IEEE Intl. Conf. on Data Mining, 2006.

[16] Jundong Li and Osmar R. Zaiane. "Associative classification with statistically significant positive and negative rules." ACM Intl. Conf. on information and knowledge management. 2015.

[17] Osmar R. Zaiane and Maria-Luiza Antonie. "On pruning and tuning rules for associative classifiers." Intl. Conf. on Knowledge-Based and Intelligent Information and Engineering Systems. 2005.

[18] Eric Bauer and Ron Kohavi. "An empirical comparison of voting classification algorithms: Bagging, boosting, and variants." Machine learning 36.1 (1999): 105-139.

[19] Dheeru Dua and Casey Graff. "UCI machine learning repository." (2017).

[20] Frans Coenen, The lucs-kdd software library, 2004. [Online]. Available: https://cgi.csc.liv.ac.uk/f̃rans/KDD/Software/LUCS-KDD-DN/lucs-kdd_DN.html

[21] J. Ross Quinlan. C4.5: programs for machine learning. Elsevier, 2014.

[22] Leo Breiman. 2001. “Random Forests.” Machine Learning 45: 5–32.

[23] Henry Hsu, and Peter A. Lachenbruch. "Paired t test." Wiley StatsRef: statistics reference online (2014).