Cascade Generalization: One versus Many

Nahla Barakat

Faculty of Informatics and Computer Science, the British University in Egypt (BUE), Cairo, Egypt.

* Corresponding author. Email: nahla.barakat@uqconnect.edu.au

Manuscript submitted November 30, 2015; March 22, 2016.

doi: 10.17706/jcp.12.3.238-249

Abstract: The choice of the best classification algorithm for a specific problem domain has been extensively researched. This issue was also the main motivations behind the ever increasing interest in ensemble methods as well as the choice of ensemble base and meta classifiers. In this paper, we extend and further evaluate a hybrid method for classifiers fusion. The method utilizes two learning algorithms only, in particular; a Support Vector Machine (SVM) as the base-level classifier and a different classification algorithm at the meta-level. This is then followed by a final voting stage. Results on nine benchmark data sets confirm that the proposed algorithm, though simple, is a promising ensemble classifier that compares favourably to other well established techniques.

Key words: Cascade generalization, classification, ensemble methods, SVM.

1. Introduction

Several machine learning and data mining techniques have been successful in discovering interesting patterns in large data sets. However, there is no single “best” algorithm which generalizes well in all problem domains [1]. Each algorithm has its own inductive bias which leads it to learn different hypotheses from the data, hence, showing different generalization performance [2]. Therefore, the problem of choosing the best technique for a specific problem domain has been extensively researched [3]. This issue is also one of the main motivations behind the ever increasing interest in ensemble methods since 1992 [4]. It has been shown that designing an ensemble composed of classifiers with different inductive biases leads to better performance, compared to those which utilize similar families of algorithms in both base and meta levels [5].

In general, designing a classifier ensemble has two phases: the first is the choice of base-level (level 0) individual classifiers. In the second phase, the predictions of base-level classifiers are (somehow) combined at the meta-level (level 1) [6], [7]. There are several methods to build base-level classifiers. One approach is to apply different machine learning algorithms to a single dataset [3]. Another approach is to apply a single algorithm to different versions of a given dataset [6], e.g., Bagging [8] and Boosting [9]. Similarly, different approaches have been proposed to combine the outputs of the base-level classifiers [10]. Voting and weighted voting are among the most commonly used methods, which have been used in both Bagging and Boosting [5]. Other effective methods for combining base-level classifiers include Stacking [4] and Cascade Generalization [5] (please refer to Section 3 for more details on these methods). In this paper, we extend and further evaluate a new method for classifiers fusion [11], which integrates cascade generalization and majority voting techniques. New sets of experiments have been added and the results have been further discussed and justified.

The method suggests using two learning algorithms at the base-level; the output of *only* one of these

algorithms is utilized to extend the input space, hence getting the meta-level data (meta-data), while the output of the other algorithm is utilized for the voting purpose only.

In particular, the main aim of this work is to further investigate the efficacy of applying a learning algorithm known to have good generalization performance (SVM) at the base-level; whose predictions are then appended to the original training data to create extended (meta-level) data set. That dataset is then used to train a meta-level classifier chosen from a different family of algorithms, e.g. decision trees or inductive rule learners.

The hypothesis is that the good generalization performance of an SVM *alone* as the base-level classifier can improve the prediction accuracy of a meta-level classifier, compared to its performance at the base level (where it is trained with the original training data). This idea was motivated by some of the results obtained in [12]-[15], where the generalization performance of the rules extracted from SVMs showed superior performance compared to the rules learned directly from the original data. In addition, our work has also been motivated by the success of methods based on cascade generalization in achieving good performance in a large number of problem domains [5]. As the results section shows, the proposed method has achieved good classification accuracy on most of the benchmark datasets. These results are comparable to the state-of-the-art ensemble methods which utilize variety, and a larger number of base-level classifiers.

The paper is organized as follows: Section 2 provides a brief background of SVMs and other base and meta-level algorithms utilized in this study, and Section 3 briefly discusses related work. Section 4 provides a description of the proposed method, while the experimental methodology is discussed in Section 5. Results followed by discussion and conclusions are presented in sections 6, 7 and 8 respectively.

2. Background

In this section we briefly review the learning algorithms utilized in this paper.

2.1. Support Vector Machines (SVMs)

Support Vector Machines are based on the principle of structural risk minimization, which aims to minimize the true error rate. SVMs operate by finding a linear hyper-plane that separates the positive and negative examples with a maximum interclass distance or *margin*. In that sense, SVMs can be considered as stable classifiers as they are robust to noise, and have good generalization performance.

In the case of non-separable data, a soft margin hyper-plane is defined to allow errors ξ_i (slack variable) in classification. Hence, the optimization problem is formulated as follows:

$$\begin{aligned} & \text{minimize } \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i \\ & \text{subject to } y_i (w \cdot x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \end{aligned}$$

where C is a regularization parameter which defines the trade-off between the training error and the margin. In the case of non-linearly separable data, SVMs map input data to be linearly separable in the feature space using kernel functions. Including kernel functions, and Lagrange multiplier α_i , the dual optimization problem is modified as follows [16]:

$$\text{maximize } w(\alpha) = \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i=1, j=1}^l \alpha_i y_i \alpha_j y_j K(x_i, x_j) \quad C \geq \alpha_i \geq 0 \quad \forall_i, \quad \sum_{i=1}^l \alpha_i y_i = 0$$

Solving for α , the training examples with non zero α 's are called the support vectors (SVs). The separating hyper-plane is completely defined by the SVs and they are the only samples which contribute to the classification decision [16].

The Weka [17] implementation of SVMs, trained using the Sequential Minimal Optimization (SMO) algorithm [18], was utilized in all of our experiments.

2.2. Meta-Level algorithms

To evaluate the effectiveness of the proposed approach, several algorithms are utilized at the meta-level, one at a time. Specifically, in addition to the SVM, we utilize two decision tree methods, a direct rule learner, and an instance-based learner. Rule learners were selected because in applications such as medical diagnosis and credit card applications, it is preferable to have both good performance (achieved by the SVMs) and transparent classification decisions (facilitated by rule learners).

The following paragraphs summarize their main features:

C5 [19] is a uni-variate decision tree learner. For growing the tree, and at each internal node, the test for best split chooses the input feature which maximizes the information gain on that node [17].

CART (Classification and Regression Trees) [20] is a multi-variate tree learner that uses a linear combination of several features at each single node, which maximize the worth of split.

JRIP is an implementation of the RIPPER algorithm [21], which learns rules directly from data by employing a sequential covering algorithm. RIPPER learns rules to cover the minority class, while the majority class is considered as the default. The FOIL [22] information gain measure is used to select the best attribute to be added to rule antecedent.

K^* is an instance-based learning algorithm with an entropy based distance measure [23]. Unlike other classifiers, which build a global model that represents the input space; instance-based learning classifiers (also known as lazy classifiers) do not build a model, but depend on local information only to classify test samples.

3. Related Work

In this section we summarize some of the most successful ensemble methods to date, namely Stacked Generalization, Cascade Generalization, Boosting, Bagging and Voting.

3.1. Stacked Generalization

Stacked Generalization, also referred to as Stacking [4], is an effective method for classifier fusion. In this method, a meta-level classifier predicts the final ensemble output by learning a model directly from the predictions of base-level classifiers (the meta-data)[4].

At the base-level, Stacked Generalization proceeds in a way similar to an n -fold cross validation on the training data. This process is repeated for each base-level classifier $k \in K$ [4].

Let $D = \{(X_m, Y_m), m = 1, \dots, M\}$, $X_i = (x_1, x_2, \dots, x_G)$, $y_m \in \{1, 2, 3, \dots, R\}$ be the training set for base-level classifiers, where y_m is the output (true) class, R is the number of classes, M is the total number of examples and G is the total number of features.

The training set is then randomly split into k disjoint datasets $(D_1, \dots, D_k)$ of equal size. For each iteration k , each of the base-level classifiers (k) is trained with the data partition D excluding the partition D_k . The resulting classifier is then applied to the validation set D_k . The predicted output class (c_k) of each classifier k_i , $i = 1, \dots, K$, for each sample m in D_k along with the target (true) class of that example constitutes the expanded meta-level training set, that is, the meta-data. At the end of this process, each example in the meta-dataset D_{L1} will be of the form:

$D_{L1} = \{(c_{1m}, \dots, c_{Km}, y_m), m = 1, \dots, M, k = 1, \dots, K\}$, where y_m is the target (true) class for the example m in the training set, c_{km} is the output class predicted by a base-level classifier k , for the example m . This meta-data (D_{L1}) is then used to train a meta-level classifier (level 1 classifier), which learns how to predict the final class labels of new sample based on the base-level predictions only, i.e., the original input features are not

utilized at the meta-level.

To classify a new (unseen) example, the output class of that example is first predicted by each of the base-level classifiers to obtain the meta-data. The final class label is then obtained by the meta-level classifier, given the example's meta-data [4].

3.2. Cascade Generalization

Similar to Stacking, Cascade Generalization [5] also operates on base and meta-levels. However, unlike Stacked Generalization, which only uses the predictions of the base-level classifiers, cascade generalization uses the predictions of the base-level classifiers to extend the dimensionality of the input space. This is done by appending the output of each of base-level classifiers as a new feature to each training example. Therefore, both base and meta-level classifiers utilize the original input features, while the meta-level classifiers also have access to additional features (the base-level classifier predictions) [5].

Let $D = \{(X_m, Y_m), m = 1, \dots, M\}$, $x_i = (x_1, x_2, \dots, x_G)$, $y_m \in \{1, 2, 3, \dots, R\}$ be the training set for base-level classifiers, where y_m is the target class, R is the number of classes, M is the total number of examples, and G is the total number of features.

An example in the meta-dataset will be in the form: $D_{L1} = \{(X_m, c_{1m}, \dots, c_{Km}, y_m) \mid k=1, \dots, K\}$, where K is the total number of base-classifiers. For base-level classifiers which outputs conditional probability distributions, the class c_{im} is replaced by the class probability.

The meta-classifier takes as its input, the outputs of base-classifiers and treats them in the same way as the original training data set. Therefore, and as stated in [5] "*Cascade generalization generates a unified theory from the base theories generated earlier*".

3.3. Voting

Majority and weighted voting are two of the most common methods for combining the outputs of base-level classifiers. In majority voting, the output class that gains the most votes becomes the ensemble output. In the case of classifiers which produce rankings or probability distributions, the highest median ranking, or the largest joint probability of each class, respectively, becomes the ensemble output [6]. In the case of weighted voting, the output of each of the base-level classifiers is given a weight usually related to its classification accuracy or posterior probability [9].

3.4. Bagging

Bagging [8], (which is also known as bootstrap aggregation) works by repeatedly creating bootstrap training sets, to train the base-level classifiers. The samples in each of the bootstrap sets are drawn randomly (with replacement) from the original dataset. A number of classifiers are trained with those bootstraps, i.e., the same algorithm is used, but with different bootstrapped data. Due to sampling by replacement, some samples in the original training set will appear several times, while others may be omitted from bootstrap sets.

The main idea of bagging is to reduce the variance of base-level classifiers [24], hence improving the final generalization performance. To decide the class of unseen test example, each base-level classifier (k_i) is used to predict the class of that example and then simple majority voting is used to decide the final class label [8].

3.5. Boosting

Similar to Bagging, Boosting [9] adaptively manipulates the distribution of the training data. The idea is to combine weak consecutive classifiers [7], where the following classifier in the sequence focuses on misclassified samples. To achieve this, boosting assigns different weights to training examples, where the

weights are adjusted after each iteration to allow the next classifier to select the previously misclassified examples. However, for this technique to improve performance each base-level classifier must perform better than random guessing.

The meta-classifier is then constructed using weighted voting on the base-level classifier predictions. Weights are given to each of these classifiers based on their individual accuracies [9]. The class that receives the largest weighted vote is then assigned to that example.

4. The Proposed Method

We propose a hybrid cascade generalization model, which extends cascade generalization by utilizing only an SVM as the base-level classifier, and adding a simple majority voting stage at the end, to decide the class of a test example. The voting stage considers three inputs: the output of the meta-classifier at the meta-level, the output of the same classifier type at the base-level (trained with the original training data) and the output of the SVM at the base-level (also trained on the original training set). The idea is to try to make use of the bias (predictions) of a specific classifier, before influencing it (relaxing it) at the meta-level by the SVM bias, represented by the extended (meta) data.

It should be noted here that only the predictions of the SVM are used to extend the input space and generate the meta-data. The prediction of the other classifier is only used at the voting stage. A description of the proposed method is provided in the following subsections:

4.1. Terminology

The following terminologies are used to describe the proposed algorithm:

D_0 : the original training set used at the base-level;

$d_0 \in D_0$: a training example;

T_0 : the original test set used for testing base-level classifiers;

$t_0 \in T_0$: a test example;

SVM_0 : SVM learned at the base level;

C_0 : classifier C learned at the base level;

D_1 : the extended training set used at the meta level (the meta-data), where each example $d_0 \in D_0$ is concatenated with the predicted class from SVM_0 ;

C_1 : classifier C (of same type as C_0) learned at the meta level;

T_1 : extended test set used to test the meta level classifier (each sample $t_0 \in T_0$ is concatenated with SVM_0 the base level predicted class for that sample);

$C_0(t_0)$: are the predicted classes for a test example at the base level by the SVM_0 and C_0 respectively;

$C_1(t_1)$: the predicted class of a test example at the meta level;

It should be noted that *base-level* and *level 0* are used interchangeably throughout the paper as is *meta-level* and *level 1*.

4.2. The Algorithm: CGen-SVM

As indicated in Section 1, the proposed algorithm, which we term CGen-SVM [11], belongs to the classifier ensemble family. The algorithm generally proceeds in the same way as traditional cascade generalization [5]. However, our approach extends the technique in the following respects:

- It integrates cascade generalization and voting for classifiers fusion, using simple majority voting of both base and meta-level classifiers, to decide the final class of a test example;
- The base-level has two classifiers, C_0 and SVM_0 , where only the output of the SVM_0 is used to create the meta-data (to train the meta-level classifier C_1), while the output of C_0 is only considered at the voting stage. To obtain good generalization performance, the classifier C is chosen from a different family of

algorithms than the SVM [5];

To clarify this point, assume that a C5 decision tree learner is used as the second classifier C , in addition to the SVM. To decide the final class of an unseen example, the predictions of the following three classifiers are considered for voting: the SVM trained with original training data (SVM_0), C5 trained with original training data (C_0), and C5 trained at the meta-level, with the extended meta-data (D_1), as per Figure 2.

The following are the steps in the algorithm:

Training Phase:

- 1) Train an SVM with D_0 , to obtain SVM_0 ;
- 2) Use SVM_0 to classify D_0 , therefore $SVM_0(d_0)$ is obtained;
- 3) Concatenate each feature vector (training sample d_0) with $SVM_0(d_0)$ for that example (please refer to Section 3), therefore, the meta-dataset D_1 is obtained;
- 4) Train the classifier C with D_1 to obtain the meta-level classifier C_1 ;
- 5) Train the same classifier C with D_0 , so a classifier C_0 is obtained.

Testing Phase:

- 6) To classify a test sample t_0 :
 - use SVM_0 to predict the class $SVM_0(t_0)$ of t_0 ;
 - concatenate t_0 with $SVM_0(t_0)$ as per step 3, so a test example t_1 for level 1 is obtained;
 - use C_1 to classify t_1 to get $C_1(t_1)$;
- 1) For the Voting step, use C_0 to classify t_0 , therefore $C_0(t_0)$ is obtained;
- 2) Use majority voting technique with $SVM_0(t_0)$, $C_0(t_0)$, $C_1(t_1)$, and the class which gets more votes is the final class of the test example t_0 .

Figs. 1 and 2 illustrate the proposed algorithm:

Fig. 1. CGen-SVM training phase.

Fig. 2. CGen-SVM testing phase.

5. Experimental Methodology

Details of the datasets used and the CGen-SVM evaluation experiments are described in the following subsections.

5.1. Datasets

The experiments were performed using nine benchmark datasets from the UCI machine learning repository [25] as shown in Table 1, details as follows:

- **Pima Indians diabetes:** A sample of 438 samples were used from the original dataset, after removing all samples with a zero value for the features 2-hour OGTT plasma glucose, diastolic blood pressure and

triceps skin fold thickness which are clinically insignificant;

- **Heart diseases:** The reduced Cleveland heart diseases dataset was used. All samples with missing values were discarded;
- **Breast cancer:** The Wisconsin breast cancer dataset was used. All repeated samples were discarded to avoid the bias resulting from the effect of those samples;

Hypothyroid: the experiments were executed as *binary classification* task: normal against all other class labels. All samples with missing values were discarded;

Australian Credit Approval: This dataset represents credit card applications, with a good mixture of attribute types. All samples with missing values were discarded;

German: This dataset represents German Credit data. It has 7 numerical, 13 categorical features;

Wine: represents wine classification data. It has 13 continuously valued features. The experiments were conducted as *binary classification* task (first class/other wine);

Ionosphere: This dataset represents radar data. All of the features are continuous in value;

Glass: This dataset is used for classification of types of glass. All features are continuous in value and the experiments were conducted as *binary classification* task (window/non-window glass);

Table 1. Datasets

Dataset	Features
Pima Indians	8
Breast cancer	9
Heart Disease	13
Thyroid	21
Australian	14
German	20
Wine	13
Ionosphere	35
Glass	9

5.2. Experiments

A 10-fold cross validation was used to select the SVM training parameters (kernel type and the regularization parameter C). The parameters that minimized the error rate over the training set were selected.

To test if the difference in accuracies were statistically significant, a large sample z test was used at a significance level of 5%.

Two sets of experiments were carried out to evaluate the proposed algorithm:

In the first set, 10-fold cross validation has been used to train each base and meta-classifier. In the second set, we compare the performance of our method (CGen-SVM) with other meta-classifiers, again utilizing 10-fold cross validation.

To train the meta-classifiers, a number of different parameter settings were tested. The parameters that gave the best performance over 10-fold CV on the training set were utilized.

The Weka [17] implementations for CART, RIPPER, and K^* were used in our experiments, while SPSS Clementine implementation of C5 decision tree has been used.

6. Results

6.1. Results of Individual Classifiers

Initially, the original training data set D_0 has been used to train different types of learning algorithms, namely; SVMs, Jrip, K^* , CART and C5, to select the best candidate for the meta level classifier.

Table 2 shows the results of different classifiers, trained using 10-fold cross validation on the original training set. From this table, it can be seen that SVM_0 achieved better accuracies than those of the other classifiers on 7 out of 9 datasets.

It was also noted that the SVM_0 trained with RBF Kernel, C 100, G 0.01, L 0.002, achieved significantly better results than other algorithms over the German data set. The best average performance over all datasets was also achieved by SVM_0 , followed by CART.

Table 2. 10-Fold CV Accuracies of Individual Classifiers on d_0

Dataset	10-fold CV %accuracies $\pm$ Std				
	SVM_0	JRIP	CART	K^*	C5
Pima Indians	85.40	78.94	80.59	76.11	83.40
Breast cancer	96.60	95.19	95.67	97.11	93.26
Heart Diseases	82.10	78.02	78.92	73.99	72.64
Hypothyroid	97.00	99.5	99.56	95.7	99.5
Australian	86.35	85.24	86.18	77.5	82.2
German	78.30	71.70	74.10	71.24	70.50
Wine	98.20	94.64	93.75	97.32	95.53
Ionosphere	91.54	85.07	87.56	79.10	84.07
Glass	99.53	96.26	98.13	89.71	96.72
Average	90.56	87.17	88.27	84.20	86.43

6.2. Results of Meta-Classifier

Several experiments were conducted with SVM_0 as base classifier, each of the other algorithms as meta classifier, one at a time, to see if the choice was a right choice.

Table 3 shows 10-fold cross validation accuracies achieved by the CGen-SVM, utilizing different types of meta-classifiers.

Comparing results in Table 3 with those of Table 2, it can be seen that meta classifiers obtained improved accuracies over the corresponding individual, base classifiers on almost all datasets. However, only the differences in the results shown in bold are statistically significant ($p < 0.05$). It has also been noted that C5 as the meta classifier achieved better performance on five out of the nine data sets. The best average performance was again obtained by C5 decision tree.

Table 3. 10-Fold CV Accuracies of CGen-SVM Utilizing Different Meta-Classifiers

Dataset	Meta-level 10-fold CV %Accuracies $\pm$ Std			
	JRIP	CART	K^*	C5
Pima Indians	86.23	85.47	79.75	84.01
Breast cancer	96.63	97.59	97.07	97.61
Heart Diseases	81.50	83.70	77.13	82.60
Hypothyroid	99.51	99.15	97.17	99.54
Australian	86.18	86.18	77.01	86.37
German	77.53	79.30	72.28	80.11
Wine	98.24	96.00	97.32	98.86
Ionosphere	90.05	91.78	80.60	91.30
Glass	98.59	98.33	90.52	98.53
Average	90.50	90.17	85.43	90.99

6.3. Results of CGen-SVM Compared to Other Meta-Classifiers

To further evaluate the performance of CGen-SVM, its performance was benchmarked against other well established ensemble methods like Stacking, Bagging, Boosting and Voting.

A 10-fold cross validation has been utilized, using the same classification algorithms utilized in building CGen-SVM meta-level classifiers. However, we decided to exclude K^* from this set of experiments due to its

low performance on the majority of datasets in the previous sections.

The training parameters that gave the best performance on the inner 10-fold CV were again utilized. The following paragraphs briefly describe these experiments:

Bagging

For this technique, several experiments have been conducted utilizing C5, CART and JRIP. However, the best accuracies have been obtained using C5.

Boosting

Similar to bagging, C5 again obtained best performance results compared to CART and JRIP.

Voting

For this set of experiments, four base-classifiers, namely; SVM, CART, JRIP and C5 were used as the base-classifiers.

Stacking

For all datasets, different meta classifiers were tested to choose the best combination of base and meta classifiers. It was found that JRIP as a meta classifier, with SVM, CART and C5 as *base-classifiers* obtained the best performance on the majority of datasets.

The best results achieved on these sets of experiments compared to CGen-SVM are shown in Table 4. As can be seen, CGen-SVM obtained the best results on six out of the nine datasets, followed by bagging with C5 and Stacking with Jrip. The best average result on all datasets was again obtained by CGen-SVM.

Table 4. Accuracies of CGen-SVM as Compared to Those of Other Ensembles, Using Exactly the Same Datasets

Dataset	CGen-SVM With C5 (10fold CV)	Stacking With JRIP as meta classifier	Bagging C5	Boosting C5	Voting
Pima Indians	84.01	83.18	82.59	80.5	83.40
Breast cancer	97.61	96.15	95.68	97.64	95.67
Heart Diseases	82.06	79.37	79.37	77.57	77.23
Hypothyroid	99.54	99.51	99.58	99.58	97.0
Australian	86.37	84.55	86.75	83.67	85.72
German	80.11	74.99	71.00	67.92	68.00
Wine	98.86	97.32	97.32	97.32	97.32
Ionosphere	91.30	87.17	88.55	88.0	86.57
Glass	98.53	95.33	96.73	96.73	97.67
Average	90.99	88.62	88.62	87.66	87.62

6.4. Comparison with of Previously Published Methods

Finally, results of CGen-SVM were benchmarked against other methods on common data sets.

Table 5 shows a comparison between CGen-SVM performance, against other published methods like bagging, boosting, meta-decision trees, stacked generalization and cascade generalization .

From this table, it can be seen that CGen-SVM compare favourably to these methods on most of the datasets, and obtained better results on two out of six data sets. However, methods like boosting, bagging and stacking use larger number of base-classifiers. In addition, we have used the output class to create the meta-data rather than class probabilities, where the latter gives better performance, as shown in [5]. However, Meta Decision Trees (MDT) obtained the best average results, followed by CGen-SVM.

From the results shown in this section, it can be seen that CGen-SVM achieved high accuracy on almost all datasets, which compares favourably to the best obtained results achieved by utilizing a larger number of base-classifiers.

Results also show that over the majority of datasets, the voting stage did not result in significant improvements in accuracies. This again reflects the effect of the SVM's strong bias on the performance of the meta-classifier. It can be argued that the difference between a specific classifier's performance at the

base-level (C_0) and the same classifier type's performance at the meta-level (C_1) can be used as an indicative measure for the effect of the SVM's inductive bias on the input space and therefore, the predictions of the meta-classifiers, where the best results were obtained by decision trees as meta-classifiers.

Table 5. Accuracies of CGen-SVM as Compared to Other Published Methods

Dataset	CGen-SVM With C5 (10fold CV)	Cascade General. [5]	Stacked general. [5]	Bagging [6]	Boosting [6]	MDT [6]
Pima Indians	84.01	73.94	-	76.59	74.41	96.68
Breast cancer	97.61	97.39	97.34	95.4	96.78	97.37
Heart Diseases	82.06	81.69	83.50	81.00	79.89	83.83
Australian	86.37	84.59	86.00	86.75	86.77	85.67
German	80.11	73.80	75.00	75.54	74.50	75.06
Ionosphere	91.30	88.45	90.37	93.00	94.00	90.68
Average	86.91	83.31	86.44	84.71	84.39	88.22

7. Discussion

In this paper, a hybrid ensemble method with one base classifier, and an additional voting stage was extensively evaluated.

Like the other cascade generalization ensembles, our algorithm combines the output of more than one classifiers, in order to obtain better performance as compared to the performance of individual classifiers. However, in the case of CGen-SVM, the predictions of the SVM were utilized, as the *only base classifier* to extend the feature space.

It has been shown that utilizing an SVM as *the only base-level classifier*, with a decision tree learner at the meta-level, leads to better performance on almost all datasets to varying degrees. The reason is that, "*In the Cascade framework lower level learners delay the decisions to the high level learners*" [5].

Support Vector Machines belong to the class of *wide margin* classifiers, and are considered "stable" classifiers, due to their excellent generalization performance. Therefore, it can be argued that the good performance of SVMs has positively influenced the performance of CGen-SVM. It has also been noted by Ali and Pazzani [26] and others, that decision trees are considered "unstable" classifiers, as small variations in the training set often result in significant changes in their rules and performance. This explains why the best performance of CGen-SVM was obtained using the C5 tree learner.

Our results also agree with those in [5], where it has been shown that the most promising ensemble classifiers use a decision tree as high-level classifier, as they have low bias and their performance is sensitive to small changes in datasets, and a classifier with strong bias at the lower level [5].

The obtained results are also consistent with the findings of earlier studies [26], where it has been shown that "*the combined error rate depends on the error rate of individual classifiers and the correlation between them*" [26], and the higher the diversity between classifiers, the better the ensemble's performance[10].

Other studies also argue that if the performance is somehow equal, an ensemble classifier with fewer components will be preferred [5]. This gives an advantage to our proposed algorithm, where we only use SVMs as the base-level classifiers and at the same time, make use of three different predictions for the ensemble at the voting stage.

8. Conclusions

In this paper a simple hybrid ensemble classifier composed of two types of classifiers only, which we have termed CGen-SVM was presented and extensively evaluated. CGen-SVM extends Cascade Generalization, where an SVM has been utilized as the base-level classifier with an additional, different,

classifier used at the meta-level. As an additional fusion step, majority voting on the predictions of both base and meta-level classifiers is also utilized.

It has been shown that CGen-SVM is a promising technique which compares well to some of other well established ensemble methods. In addition, it also shows that the SVM's inductive bias, which aims at reducing the true error rate, is beneficial to the ensemble. Our results also confirm the findings of previous studies which have shown that the performance of an ensemble classifier is correlated to the performance of the individual classifiers, and the diversity between them. Therefore, an improved performance can be obtained using different families of classifiers at the base and meta-levels.

As future research, posterior class probabilities instead of class predictions at the voting stage may be utilized, which may lead to better results after the majority voting stage.

References

- [1] Schaffer, C. (1994). *A Conservation Law for Generalization Performance*.
- [2] Mitchell, T. (1997). *Machine Learning*, McGraw Hill.
- [3] Merz, C. J. (1999). Using correspondence analysis to combine classifiers. *Machine Learning*, 36, 33-58.
- [4] Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5, 241-259.
- [5] Gama, J., & Brazdil, P. (2000). Cascade generalization. *Machine Learning*, 41, 315-343.
- [6] Todorovski, L., & Dzeroski, S. (2003). Combining classifiers with meta decision trees. *Machine Learning*, 50, 223-249.
- [7] Galar, M., *et al.* (2012). A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on System, Man and Cybernetics Part C*, 42, 463-484.
- [8] Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24, 123-140.
- [9] Freund, Y., & Schapire, R. E. (1996). *Experiments with a New Boosting Algorithm*, Morgan Kaufmann.
- [10] Kuncheva, L. I., & Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationships with the ensemble accuracy. *Machine Learning*, 51, 181-207.
- [11] Barakat, N. (2010). *Cascade Generalization: Is svms' Inductive Bias Useful?* Istanbul, Turkey: IEEE.
- [12] Barakat, N., & Bradley, A. (2007). Rule extraction from support vector machines: A sequential covering approach. *IEEE Transactions on Knowledge and Data Engineering*, 19, 729-741.
- [13] Fu, X., *et al.* (2004). Extracting the knowledge embedded in support vector machines.
- [14] Barakat, N., & Bradley, A. (2006). Rule extraction from support vector machines: Measuring the explanation capability using the area under the roc curve. IEEE Press.
- [15] He, J., *et al.* (2006). Rule generation for protein secondary structure prediction with support vector machines and decision tree. *IEEE Transactions on Nanobioscience*, 5, 46-53.
- [16] Cristianini, N., & Taylor, J. S. (2000). *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge: Cambridge University Press.
- [17] Witten, I., & Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques* (2nd ed.). San Francisco: Morgan Kaufmann.
- [18] Platt, J. (1998). Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. Burges, & A. Smola (Eds.), *Advances in Kernel Methods - Support Vector Learning*, MIT Press.
- [19] Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*, SanMateo, CA: Morgan Kaufmann.
- [20] Breiman, L., *et al.* (1984). *Classification and Regression Trees*, Monterrey, Ca: Wadsworth and Brooks.
- [21] Cohen, W. W. (1995). *Fast Effective Rule Induction*.
- [22] Quinlan, J. R. (1990). Learning logical definitions from relations. *Machine Learning*, 5, 239-266.
- [23] Cleary, J. G., & Trigg, L. E. (1995). *K*: An Instance-Based Learner Using an Entropic Distance Measure*,

Morgan Kaufmann.

- [24] kuncheva, L. I., *et al.* (2002). An experimental study on diversity for bagging and boosting with linear classifiers. *Information Fusion*, 3, 245-258.
- [25] Merz, C., & Murphy, P. (1998). Uci machine learning repository.
- [26] Ali, K. M., & Pazzani, M. J. (1996). Error reduction through learning multiple descriptions. *Machine Learning*, 24, 173-202.

Nahla H. Barakat received the Ph.D. degree in computer science from the University of Queensland, Brisbane, Australia. She is currently an associate professor in computer science with The British University in Egypt (BUE) in Cairo, Egypt. She has more than 15 years' university teaching experience, this in addition to more than ten year industry experience in the area of IT in a multinational environment. Her current research interests include machine learning and medical data mining.