Selective Fusion of Heterogeneous Classifiers

Grigorios Tsoumakas, Lefteris Angelis and Ioannis Vlahavas
Dept. of Informatics, Aristotle University of Thessaloniki
54124 Thesaloniki, Greece
{greg,lef,vlahavas}@csd.auth.gr

Abstract

There are two main paradigms in combining different classification algorithms: Classifier Selection and Classifier Fusion. The first one selects a single model for classifying a new instance, while the latter combines the decisions of all models. The work presented in this paper stands in between these two paradigms aiming to tackle the disadvantages and benefit from the advantages of both. In particular, this paper proposes the use of statistical procedures for the selection of the best subgroup among different classification algorithms and the subsequent fusion of the decision of the models in this subgroup with simple methods like Weighted Voting. Extensive experimental results show that the proposed approach, Selective Fusion, improves over simple selection and fusion methods, leading to performance comparable with the state-of-the-art heterogeneous classifier combination method of Stacking, without the additional computational cost and learning problems of meta-training.

1 Introduction

A very active research area during the recent years involves methodologies and systems for the combination of multiple predictive models. It has attracted scientists from several fields including Statistics, Machine Learning, Pattern Recognition and Knowledge Discovery. Within the Machine Learning community this area is commonly referred to as Ensemble Methods [8].

The main motivation for combining multiple predictive models is the improvement over the accuracy of a single classification or regression model. Another reason is the scaling of inductive algorithms to very large databases. Most inductive algorithms are too computationally complex and suffer from memory problems when applied to very large databases. A solution to this problem is to horizontally partition the database into smaller parts, train a predictive model in each of the smaller manageable part and combine the predictive models. Finally, the problem of learning from multiple physically distributed data sets that can't be collected to a single site due to privacy or size reasons, can also be tackled with the combination of multiple predictive models, each trained on a different distributed data set.

Models that have been derived from different executions of the same learning algorithm are often called *Homogeneous*. Such models can be induced by injecting randomness into the learning algorithm or through the manipulation of the training instances, the input attributes and the model outputs [9]. On the other hand, models that have been derived from running different learning algorithms on the same data set are often called *Heterogeneous*.

This paper deals with the combination of heterogeneous classification models for accuracy improvement. There are two main paradigms in combining different classification algorithms: *classifier selection* and *classifier fusion*. The first one selects a *single* algorithm for classifying a new instance, while the latter combines the decisions of *all* algorithms.

A very simple but effective method of the first paradigm is Evaluation and Selection. This method evaluates each of the models (typically using

10-fold cross-validation) on the training set and selects the best one for application to the test set. Dzeroski and Zenko [10] have argued that this method should be considered as a base line for comparison with other methods, because it is simple and shows good performance. However, it does not always manage to select the most accurate model [27].

A very simple and widely-used method of the second paradigm is Voting. According to this procedure, each model outputs a class value (or ranking, or probability distribution) and the class with the most votes (or the highest average ranking, or average probability) is the one proposed by the ensemble. This method is expected to have better accuracy than that of the best individual model, due to the correction of uncorrelated errors through voting. However, the fact that all models (even the less accurate ones) participate with equal vote, has eventually an overall negative effect on the performance of the method. In Weighted Voting, the classification models are not treated equally. Each model is associated with a coefficient (weight), usually proportional to its classification accuracy. This partly amends the problem of inferior models, but it does not eliminate it completely as it still allows them to affect the final decision.

The main idea of this work is to completely exclude learning algorithms with low performance which may produce misleading results. It seems more reasonable to select a subset of algorithms that perform significantly better than the rest, than to use the whole initial set of algorithms. In particular, this paper proposes: a) the use of statistical techniques for the *selection* of the best subgroup among different classification algorithms and b) the subsequent *fusion* of the decision of the models in this subgroup with a simple method like Weighted Voting. For this reason the approach is dubbed *Selective Fusion.*

For the first part of the approach, we adapt three *multiple comparisons procedures* (Tukey's test, Hsu's test, Scott & Knott's procedure) to the problem of finding the best subgroup of algorithms and apply them to estimates of the algorithm's errors obtained by k-fold cross-validation. For the second part we experiment with Voting and Weighted Voting for the fusion of the subgroup's decisions.

Extensive experimental results with 10 different classification algorithms and 40 data sets show that the proposed approach improves significantly simple combination methods, leading to predictive performance comparable with the state-of-the-art method of Stacking with Multi-Response Model Trees [10] without the additional computational cost of meta-training.

In addition, we varied the number of algorithms from 3 to 10 and noticed that the accuracy of the proposed approach is not affected negatively by the number of learning algorithms, in contrast to other effective approaches. This is an important advantage of Selective Fusion that allows for the increase of its predictive performance with the addition of more, accurate and diverse learning algorithms in the framework.

The rest of this paper is organized as follows. Section 2 presents related work on combining multiple classification algorithms and Section 3 describes the details of the proposed approach. Section 4 presents the experimental setup for comparing Selective Fusion with other established algorithm combination methods and Section 5 discusses the results. Section 6 studies the computational complexity and predictive performance of Selective Fusion and other methods with respect to the number of algorithms. Finally, Section 7 summarizes and concludes this work.

2 Combining Multiple Classification Algorithms

The combination of multiple classification algorithms is based on the idea that no single algorithm can be universally optimal for all learning problems [24]. As already mentioned, there are two main paradigms for handling an ensemble of different classification algorithms: *Classifier Selection* and *Classifier Fusion*. This section reviews several methods from both categories.

2.1 Classifier Selection

A very simple method of this category is found in the literature as Evaluation and Selection or SelectBest. This method evaluates each of the classification algorithms (typically using 10-fold cross-validation) on the training set and selects the best one for application on the test set. Although this method is simple, it has been found to be highly effective and comparable to other more complex state-of-the-art methods [10].

Another line of research proposes the selection of a learning algorithm based on its performance on similar learning domains. Several approaches have been proposed for the characterization of learning domain, including general, statistical and information-theoretic measures [5], landmarking [21], histograms [16] and model-based data characterizations [2]. Apart from the characterization of each domain, the performance of each learning algorithm on that domain is recorded. When a new domain arrives, the performance of the algorithms in the k-nearest neighbors of that domain are retrieved and the algorithms are ranked according to their average performance. In [5], algorithms are ranked based on a measure called Adjusted Ratio of Ratios (ARR), that combines accuracy and learning time of algorithm, while in [17], algorithms are ranked based on Data Envelopment Analysis, a multicriteria evaluation technique that can combine various performance metrics, like accuracy, storage space, and learning time.

In [11, 31], the accuracy of the algorithms is estimated locally on a number of examples that surround each test example. Such approaches belong to the family of Dynamic Classifier Selection [12] and use a different algorithm in different parts of the instance space.

Two similar, but more complicated approaches that were developed by Merz [20] are Dynamic Selection and Dynamic Weighting. The selection of algorithms is based on their local performance, but not around the test instance itself, rather around the meta-instance comprising the predictions of the classification models on the test instance. Training meta-instances are produced by recording the predictions of each algorithm, using the full training data both for training and for testing. Performance data are produced by running m k-fold cross-validations, and averaging for each training instance the m evaluations.

2.2 Classifier Fusion

Unweighted and Weighted Voting are two of the simplest methods for combining not only Heterogeneous but also Homogeneous models. In Voting, each model outputs a class value (or ranking, or probability distribution) and the class with the most votes (or the highest average ranking, or average probability) is the one proposed by the ensemble. Note that this type of Voting is in fact called Plurality Voting, in contrast to the frequently used term Majority Voting, as the latter implies that at least 50% (the majority) of the votes should belong to the winning class. In Weighted Voting, the classification models are not treated equally. Each model is associated with a coefficient (weight), usually proportional to its classification accuracy.

Stacked Generalization [30], also known as Stacking, is a method that combines multiple classifiers by learning a meta-level (or level-1) model that predicts the correct class based on the decisions of the base-level (or level-0) classifiers. This model is induced on a set of meta-level training data that are typically produced by applying a procedure similar to k-fold cross-validation on the training data:

Let D be the level-0 training data set. D is randomly split into k disjoint parts $D_1 \dots D_k$ of equal size. For each fold i=1..k of the process, the base-level classifiers are trained on the set $D \setminus D_i$ and then applied to the test set D_i. The output of the classifiers for a test instance along with the true class of that instance form a meta-instance.

A meta-classifier is then trained on the meta-instances and the base-level classifiers are trained on all training data D. When a new instance appears for classification, the output of all base-level classifiers is first calculated and then propagated to the meta-level classifier, which outputs the final result.

Ting and Witten [26] have shown that Stacking works good when meta-instances are formed by probability distributions for each class instead of just a class label. A recent study [10] has shown that Stacking with Multi-Response Model Trees as the meta-level learning algorithm and probability distributions, is the most accurate heterogeneous classifier combination method of the Stacking family.

3 Selective Fusion

Selective Fusion stands in between the paradigms of Selection and Fusion aiming to deal with the disadvantages and benefit from the advantages of both. Instead of selecting a single algorithm or fusing all algorithms, Selective Fusion initially selects a subset of algorithms with significantly better performance than the rest and then combines their decision with a simple combination method.

The critical part of the approach is the first one and is essentially a solution to the problem of finding the most accurate algorithms for a particular data set. It is clear that such a problem cannot have a unique solution as the notion of *best* is quite subjective and depends on the criterion used. In fact, one can always order the algorithms according to their accuracy, but then the critical question is: how many of them are truly the best in the sense that they differ significantly from all the others? So, the problem of finding the best classifiers can be rephrased as the problem of finding a subset satisfying certain criteria ensuring that: a) all the classifiers in the subset have similar good performance and b) the performance of at least one classifier in the subset differs significantly from those not in the subset.

In order to deal with this problem, we employ statistical analysis techniques that can produce groups of classifiers, homogeneous with respect to their performance. The techniques used for this task belong to the broader class of statistical procedures under the generic name *multiple comparisons procedures*. There exists a vast statistical literature on the subject and a large amount of different methods [13]. Here we adapt three of the most representative ones (Tukey's test, Hsu's test, Scott & Knott's procedure) to our problem of finding just one group, the one with the lowest error rate means. These three procedures are described in the following subsections.

Statistical methods require data. In our case, we need to collect several independent error estimates for each of the classification algorithms in order to select the best subgroup. For this purpose we employ stratified k-fold cross-validation. Let L_i, i=1..n be an ensemble of different classification algorithms and D a data set containing training examples $d_i(x_i, y_i)$, where x_i is the feature vector and y_i the class of example i. D is broken into k

disjoint parts $D_1 \dots D_k$ of equal size and equal class distribution. Each part D_j is used in turn as a test set, while the rest of the parts are merged into a training set $D'_j = D \setminus D_j$. Each of the classification algorithms L_i is used to build a classifier C_{ij} based on each of the training set D'_j. Each classifier C_{ij} is evaluated on the corresponding test set D_j. This results in k error-rate estimates e_{ij}, i=1..n, j=1..k for each of the n classification algorithms L_i.

All three statistical procedures that follow initially calculate for each algorithm the mean error rate across all estimations:

$$\bar{e}_i = \frac{1}{k} \sum_{j=1}^{k} e_{ij} \tag{1}$$

3.1 Tukey's test

This is the most known and widely used test and can be considered as a representative of the whole class of statistical tests known as *multiple range tests*. The general purpose of these tests is to make all pairwise comparisons between the levels of a factor affecting a response variable in order to identify the most significant differences between them. This is achieved by computing confidence intervals for each difference of the true means and then characterizing the difference significant if the interval does not contain the zero value.

In our setup, the factor levels (or treatments) are the different classifiers while the response variable is the error rate. The multiple range tests put together in homogeneous groups all classifiers that do not have significant difference between them. It is clear that these groups are not disjoint, i.e. a classifier can belong to more than one groups. However, in our case this is not a problem since we are interested only in one of the groups, the one with the smallest mean error rates.

Tukey's honestly significant difference (HSD) or wholly significant difference (WSD) test is used to make all of the $m = \binom{n}{2}$ pairwise comparisons between the mean of the n algorithms. The significance of any pairwise difference between treatments (i_1, i_2) is tested using the 100(1-α)% confidence interval for the differences of the unknown mean errors:

$$\mu_{i_1} - \mu_{i_2} \in \left[\bar{e}_{i_1} - \bar{e}_{i_2} \pm Q_{n,\nu;\alpha} S/\sqrt{k}\right] \quad \text{for} \quad 1 \leq i_1 < i_2 \leq n \tag{2}$$

where

$$S = \sqrt{\frac{\sum_{i=1}^{n} \sum_{j=1}^{k} (e_{ij} - \bar{e}_i)^2}{n(k-1)}} \tag{3}$$

and $Q_{n,\nu;\alpha}$ is the upper α point of the Studentized range distribution with parameter n and $\nu = n(k-1)$ degrees of freedom. For more details and tables of critical values for that distribution we refer to Hochberg & Tamhane [13]. The difference is not significant if the above confidence interval contains the 0. Tukey's test is more powerful when testing a large number of pairs of means (as in our case).

Regarding the implementation of this method for our case, the procedure involves the following steps (note that for our case it is not necessary to conduct all pairs of comparisons since we want to derive only one group with the best classifiers):

1. Sort the mean errors in ascending order:

$$\bar{e}_1 \leq \bar{e}_2 \leq \ldots \leq \bar{e}_n \tag{4}$$

2. Set the group of classifiers to contain the algorithm with the lowest error mean: $G \leftarrow \{L_1\}$

3. For each algorithm L_i, i=2..n do the following:

   - Calculate the upper bound of the confidence interval:

$$u = \bar{e}_1 - \bar{e}_i + Q_{n,\nu;\alpha} S/\sqrt{k} \tag{5}$$

   - if $u > 0$ then no significant difference is detected and therefore we include L_i in the subgroup with the best algorithms $G = G \bigcup L_i$, otherwise goto step 4

4. Return subset of algorithms G

3.2 Hsu's test

The adaptation of Hsu's approach [14] to our problem is an attempt to find the true best algorithm. Although in a sample there is only one best algorithm, others which do not differ significantly should also be considered as candidates, since the true best algorithm is unknown. Hsu's approach considers the largest values as best and for this reason we formalize its description based on the notion of accuracy and the mean accuracy:

$$a_{ij} = 1 - e_{ij} \qquad \bar{a}_i = \frac{1}{k}\sum_{j=1}^{k} a_{ij} \tag{6}$$

For each algorithm i=1..n, Hsu's approach calculates the following 100(1-α)% simultaneous confidence interval:

$$\mu_i - \max_j \mu_j \in \left[(\bar{a}_i - \max_{j \neq i} \bar{a}_j - d)^-, (\bar{a}_i - \max_{j \neq i} \bar{a}_j + d)^+\right] \tag{7}$$

where

$$d = T_{n-1,\nu,\rho;\alpha} S \sqrt{\frac{2}{k}} \qquad S = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{k}(a_{ij} - \bar{a}_i)^2}{n(k-1)}} \tag{8}$$

$$x^- = \min(x, 0) \qquad \text{and} \qquad x^+ = \max(x, 0) \tag{9}$$

$T_{n-1,\nu;\alpha}$ is the critical value of the n-variate equicorrelated t-distribution with parameters n-1, $\nu = n(k-1)$ and significance level α. A table with critical values can be found in Hochberg & Tamhane [13].

If the upper bound of the above simultaneous confidence interval is 0 then we can reject algorithm i from the candidates for the most accurate classifier.

3.3 Scott and Knott's procedure

Scott and Knott [25] developed a procedure for obtaining homogeneous groups of n treatments (here n algorithms) based on the means of the response variable. We assume that each treatment has replications of the response variable (for each algorithm we have k=10 replications of the error rate). In our case we are interested only in the homogeneous group with the smallest mean values of the error rate. The procedure of Scott and Knott is given in the following steps:

1. Sort the mean errors in ascending order:

$$\bar{e}_1 \leq \bar{e}_2 \leq \ldots \leq \bar{e}_n \tag{10}$$

2. For each i=1..n-1 separate the group of ordered error means E into 2 subgroups $E_1 = \{\bar{e}_1..\bar{e}_i\}$ and $E_2 = \{\bar{e}_{i+1}..\bar{e}_n\}$ and compute the between-groups sum of squares:

$$B_i = k(|E_1|(\bar{e}_{E_1} - \bar{e}_E)^2 + |E_2|(\bar{e}_{E_2} - \bar{e}_E)^2) \tag{11}$$

where $|E_1|$, $|E_2|$ are the cardinalities of the two subgroups (i.e. $|E_1| = i$ and $|E_2| = n - i$), and $\bar{e}_E$, $\bar{e}_{E_1}$, $\bar{e}_{E_2}$ are the means of groups E, E_1 and E_2:

$$\bar{e}_E = \frac{1}{n}\sum_{i=1}^{n} \bar{e}_i \qquad \bar{e}_{E_1} = \frac{1}{|E_1|}\sum_{i \in E_1} \bar{e}_i \qquad \bar{e}_{E_2} = \frac{1}{|E_2|}\sum_{i \in E_2} \bar{e}_i \tag{12}$$

3. Find the partition that maximizes the value of the above sum of squares:

$$B_{i^*} = max\left\{k(|E_1|(\bar{e}_{E_1} - \bar{e}_E)^2 + |E_2|(\bar{e}_{E_2} - \bar{e}_E)^2)\right\} \tag{13}$$

4. Compute:

$$s^2 = \frac{\sum_{i=1}^{n}\sum_{j=1}^{k}(e_{ij} - \bar{e}_E)^2}{nk} \tag{14}$$

and the statistic λ:

$$\lambda = \frac{\pi}{2(\pi - 2)}\frac{B_{i^*}}{s^2} \tag{15}$$

which has approximately a χ^2_ν distribution where the degrees of freedom are given by $\nu = k/(\pi - 2)$ (rounded).

5. If $\lambda > \chi^2_{\nu;\alpha}$ (where α is a predefined level), then set $n = |E_1|$, $E = E_1$ and return to step 1 (i.e. repeat the procedure with the first group

with the smallest means). If $\lambda < \chi^2_{\nu;\alpha}$ then all the means fall in the same homogeneous group.

4 Experimental Setup

This section provides information on the datasets, combination methods and participating algorithms that were used for the experiments. The WEKA machine learning software [29] was used as the platform for all the experiments.

4.1 Datasets

The predictive performance of the combination methods was evaluated on 40 data sets from the UCI Machine Learning repository [4]. Table 1 presents the details of the datasets (Folder in UCI server, number of instances, classes, continuous and discrete attributes, percentage of missing values).

4.2 Combination Methods and Participating Algorithms

We compared the following classifier combination methods: Stacking with Multi-Response Model Trees (SMT), Voting (V), Weighted Voting (WV), Evaluation and Selection (ES) and the proposed combination method, Selective Fusion (SF), using the procedures of Tukey (Tuk), Hsu (Hsu) and Scott and Knott (S&K) and combining the selected models with Voting and Weighted Voting.

These methods are used in conjunction with the WEKA implementations of the following 10 base-level classification algorithms, which are run with default parameter values unless otherwise stated:

- DT: the decision table algorithm of Kohavi [18].

Table 1: Data sets used in the experiments: Id, folder in UCI server, number of instances, classes, continuous and discrete attributes, percentage of missing values

Id	UCI Folder	Inst	Cls	Cnt	Dsc	MV
01	annealing	898	6	6	32	64.98
02	audiology	226	24	0	69	2.03
03	autos	205	7	15	10	1.15
04	balance-scale	625	3	4	0	0.00
05	breast-cancer	286	2	0	9	0.35
06	breast-cancer-wisconsin	699	2	9	0	0.25
07	car	1728	4	0	6	0.00
08	chess (kr-vs-kp)	3196	2	0	36	0.00
09	cmc	1473	3	2	7	0.00
10	dermatology	366	6	1	33	0.01
11	ecoli	336	8	7	0	0.00
12	glass	214	7	9	0	0.00
13	heart-disease (cleveland)	303	5	6	7	0.18
14	heart-disease (hungary)	294	5	6	7	20.46
15	heart-disease (switzerland)	123	5	6	7	17.07
16	heart-disease (va)	200	5	6	7	26.85
17	hepatitis	155	2	6	13	5.67
18	horse-colic	368	2	7	15	23.80
19	image	2310	7	19	0	0.00
20	ionosphere	351	2	34	0	0.00
21	iris	150	3	4	0	0.00
22	labor	57	2	8	8	35.75
23	lymphography	148	4	3	15	0.00
24	pima-indians-diabetes	768	2	8	0	0.00
25	primary-tumor	339	22	0	17	3.90
26	soybean	683	19	0	35	9.78
27	statlog (australian)	690	2	6	9	0.65
28	statlog (german)	1000	2	7	13	0.00
29	statlog (heart)	270	2	13	0	0.00
30	statlog (satimage)	6435	6	36	0	0.00
31	statlog (segment)	2310	7	19	0	0.00
32	statlog (vehicle)	846	4	18	0	0.00
33	thyroid-disease	3772	4	7	22	5.54
34	tic-tac-toe	958	2	0	9	0.00
35	undocumented (sonar)	208	2	60	0	0.00
36	undocumented (vowel-context)	990	11	10	3	0.00
37	voting-records	435	2	0	16	5.63
38	waveform	5000	3	40	0	0.00
39	wine	178	3	13	0	0.00
40	zoo	101	7	1	16	0.00

- JRip: the RIPPER rule learning algorithm [7].
- PART: the PART rule learning algorithm [28].
- J48: the decision tree learning algorithm C4.5 [23], using Laplace smoothing for predicted probabilities.
- IBk: the *k* nearest neighbor algorithm [1].
- K*: an instance-based learning algorithm with entropic distance measure [6].
- NB: the Naive Bayes algorithm [15] using the kernel density estimator rather than assume normal distributions for numeric attributes.
- SMO: the sequential minimal optimization algorithm for training a support vector classifier using polynomial kernels [22].
- RBF: WEKA implementation of an algorithm for training a radial basis function network [3].
- MLP: WEKA implementation of an algorithm for training a multilayer perceptron [3].

The meta-level training data for Stacking are produced using 10-fold stratified cross-validation on the training set. Exactly the same procedure is used for estimating the accuracy of the above base-level algorithms. All combination methods operate on probability distributions of the base-level classifiers.

5 Results and Discussion

For the comparison of the different classifier combination methods we use two main approaches: a) we repeat 4 times a 10-fold stratified cross-validation experiment with different splits, and b) we make statistical significance tests for all pairs of combination methods based on each of the 10-fold experiments. The two above strategies are suggested in the literature as the most reliable ways for comparing learning schemes [28]. Some other methods for performance evaluation are hold-out, bootstrap, leave-one-out, [28] and leave-one-batch-out [19].

Table 2, shows the average error rate of the combining methods for each data set, averaged over the 10 folds of all four cross-validation experiments. The last line presents the geometric mean, which is a normalized version of the standard mean. The geometric mean is considered more robust for comparison over several data sets that give error values not normally distributed with many outliers.

As far as the standard methods are concerned we notice that Voting has the highest geometric mean of error, followed by Evaluation and Selection, Weighted Voting and Stacking with Multi-Response Model Trees. As expected, the complex state-of-the-art method of Stacking is the best among these four considering the geometric mean. Also, as expected, Weighted Voting was found to give better results than Voting.

Looking at the results of Selective Fusion, we notice that combining with (Weighted) Voting the subgroups produced by the 3 statistical procedures lead to better results than combining all algorithms. Comparing the performance of the 3 statistical procedures, we notice that Scott & Knott's gives the best results in both Voting and Weighted Voting, followed by Tukey's, while the worst results are obtained using Hsu's test. Tukey's procedure selects 6.99 algorithms on average over the 40 data sets and 4 experiments, while Hsu's selects 6.11 and Scott & Knott's 5.53.

The most important finding in the results is that Selective Fusion with

Table 2: Average error rate of combination methods on the 40 data sets

		Voting				Weighted Voting				
Id	ES	All	Tuk	Hsu	S&K	All	Tuk	Hsu	S&K	SMT
01	0.0145	0.0253	0.0109	0.0134	0.0134	0.0240	0.0106	0.0128	0.0128	0.0100
02	0.1776	0.1578	0.1577	0.1588	0.1632	0.1566	0.1577	0.1588	0.1643	0.2409
03	0.2097	0.1770	0.1683	0.1756	0.1817	0.1757	0.1658	0.1719	0.1792	0.2023
04	0.0897	0.1096	0.1020	0.1024	0.1009	0.1076	0.1016	0.1024	0.1013	0.0428
05	0.2909	0.2637	0.2681	0.2725	0.2619	0.2637	0.2672	0.2716	0.2619	0.2980
06	0.0247	0.0290	0.0293	0.0293	0.0300	0.0293	0.0297	0.0293	0.0300	0.0315
07	0.0058	0.0278	0.0058	0.0058	0.0058	0.0259	0.0058	0.0058	0.0058	0.0058
08	0.4681	0.4606	0.4515	0.4474	0.4501	0.4582	0.4521	0.4477	0.4499	0.4581
09	0.1495	0.1441	0.1489	0.1462	0.1475	0.1454	0.1482	0.1462	0.1468	0.1645
10	0.1460	0.1319	0.1370	0.1359	0.1435	0.1312	0.1370	0.1355	0.1435	0.1482
11	0.2520	0.2305	0.2373	0.2390	0.2450	0.2318	0.2378	0.2390	0.2453	0.2368
12	0.0279	0.0245	0.0287	0.0293	0.0280	0.0245	0.0293	0.0293	0.0280	0.0279
13	0.2334	0.2340	0.2366	0.2356	0.2369	0.2337	0.2363	0.2347	0.2363	0.2382
14	0.1324	0.1293	0.1338	0.1367	0.1375	0.1293	0.1330	0.1351	0.1375	0.1433
15	0.2547	0.2533	0.2497	0.2555	0.2590	0.2544	0.2485	0.2554	0.2602	0.2417
16	0.1717	0.1684	0.1708	0.1733	0.1625	0.1684	0.1717	0.1725	0.1633	0.1966
17	0.1504	0.1716	0.1733	0.1716	0.1724	0.1725	0.1742	0.1716	0.1716	0.1759
18	0.6527	0.6433	0.6537	0.6497	0.6579	0.6599	0.6599	0.6558	0.6641	0.6155
19	0.1630	0.1676	0.1657	0.1630	0.1602	0.1676	0.1648	0.1630	0.1593	0.1759
20	0.6863	0.6438	0.6438	0.6425	0.6513	0.6500	0.6500	0.6463	0.6575	0.7363
21	0.1544	0.1545	0.1545	0.1577	0.1545	0.1529	0.1529	0.1561	0.1529	0.1793
22	0.0050	0.0203	0.0039	0.0039	0.0039	0.0179	0.0039	0.0039	0.0039	0.0038
23	0.0312	0.0172	0.0166	0.0166	0.0167	0.0165	0.0165	0.0166	0.0167	0.0209
24	0.0919	0.0747	0.0719	0.0669	0.0683	0.0740	0.0712	0.0669	0.0683	0.0790
25	0.0433	0.0483	0.0483	0.0483	0.0483	0.0483	0.0483	0.0483	0.0483	0.0433
26	0.0070	0.0054	0.0052	0.0052	0.0050	0.0053	0.0052	0.0052	0.0050	0.0062
27	0.1058	0.0558	0.0517	0.0475	0.0475	0.0558	0.0558	0.0475	0.0475	0.0658
28	0.1742	0.1623	0.1638	0.1570	0.1555	0.1639	0.1621	0.1554	0.1538	0.1773
29	0.5119	0.5406	0.5434	0.5398	0.5258	0.5436	0.5419	0.5383	0.5229	0.5833
30	0.0943	0.0833	0.0848	0.0854	0.0875	0.0822	0.0848	0.0855	0.0875	0.0813
31	0.0300	0.0158	0.0161	0.0165	0.0173	0.0158	0.0160	0.0163	0.0171	0.0207
32	0.1593	0.1499	0.1439	0.1402	0.1474	0.1427	0.1439	0.1402	0.1402	0.1655
33	0.0699	0.0622	0.0615	0.0615	0.0629	0.0626	0.0615	0.0619	0.0633	0.0747
34	0.0144	0.0175	0.0175	0.0177	0.0177	0.0170	0.0175	0.0177	0.0177	0.0060
35	0.1781	0.2426	0.1781	0.1781	0.1781	0.2379	0.1752	0.1781	0.1781	0.1643
36	0.0431	0.0391	0.0391	0.0391	0.0385	0.0391	0.0391	0.0391	0.0385	0.0374
37	0.0098	0.0164	0.0096	0.0093	0.0091	0.0116	0.0096	0.0093	0.0091	0.0109
38	0.1351	0.1543	0.1554	0.1406	0.1352	0.1538	0.1554	0.1406	0.1352	0.1453
39	0.0196	0.0098	0.0098	0.0126	0.0098	0.0084	0.0098	0.0126	0.0098	0.0252
40	0.0516	0.0391	0.0391	0.0391	0.0368	0.0391	0.0391	0.0391	0.0368	0.0914
Avg.	0.0832	0.0847	0.0749	0.0754	0.0749	0.0828	0.0749	0.0752	0.0747	0.0796

(Weighted) Voting exhibit lower error rate than Stacking with Multi-Response Model Trees. The latter is considered the state-of-the-art in Stacking which is in turn considered the state-of-the-art in heterogeneous classifier combination. What is more interesting is the fact that the computational cost of Selective Fusion is much less than that of Stacking. A computational complexity analysis follows in Section 6.

In order to detect whether the above described performance differences are significant, we applied a paired t-test to the errors of the methods in the 10 folds of one of the 10 cross-validation experiments with a significance level of 0.05. From the outcome of this test on all data sets we report the statistically significant wins and losses for each pair of methods.

Table 3 shows the significant wins and loses for each pair of combination methods, averaged over the 4 experiments and rounded to the closest integer. The results show that Selective Fusion improves significantly over simple Voting, Weighted Voting and Evaluation and Selection, and slightly over the state-of-the-art method of Stacking with Multi-Response Model-Trees. In addition, as already mentioned, Selective Fusion is a much simpler and less computationally expensive method.

6 Varying the number of classifiers

In this section we will study the relationship of the number of classifiers with the computational complexity and the predictive performance of the methods that participated in the experiments.

6.1 Computational Complexity

Consider a set of Classification Algorithms C_i, i=1..n and a training set D, with m instances and f features. The computational cost of training

Table 3: Significant wins and losses (w:l) for each pair of methods

			Voting				Weighted Voting				
		ES	All	Tuk	Hsu	S&K	All	Tuk	Hsu	S&K	SMT
	ES		6:6	2:5	1:6	1:4	5:6	2:5	2:6	1:5	4:5
Vot.	All	6:6		1:4	0:5	1:6	0:2	1:4	0:5	1:6	4:7
”	Tuk	5:2	4:1		0:1	0:1	4:1	0:0	0:1	0:1	4:2
”	Hsu	6:1	5:0	1:0		0:1	4:0	1:0	0:0	1:1	4:2
”	S&K	4:1	6:1	1:0	1:0		5:1	1:0	0:0	0:0	5:2
W.Vot.	All	6:5	2:0	1:4	0:4	1:5		1:4	0:4	1:5	4:6
”	Tuk	5:2	4:1	0:0	0:1	0:1	4:1		0:1	0:1	3:2
”	Hsu	6:2	5:0	1:0	0:0	0:0	4:0	1:0		0:0	5:2
”	S&K	5:1	6:1	1:0	1:1	0:0	5:1	1:0	0:0		5:3
	SMT	5:4	7:4	2:4	2:4	4:5	5:3	2:4	2:5	3:5	

each algorithm C_i using D is $cost(C_i, D)$. The lower bound of any classifier combination method is:

$$\sum_{i=1}^{n} cost(C_i, m, f) \tag{16}$$

This is actually the cost of Voting. Evaluation and Selection, Weighted Voting and Selective Fusion require an estimation of accuracy, which is typically obtained by k-fold cross-validation. This demands another k training sessions for each algorithm on data sets of size (rows) equal to $\frac{k-1}{k}m$. If we make the assumption that the complexity of the inductive algorithms used for base-level learning is linear to the instances of the data set, then the total computational cost for these methods is:

$$\sum_{i=1}^{n} cost(C_i, m, f) + k\sum_{i=1}^{n} cost(C_i, \frac{k-1}{k}m, f) \approx k\sum_{i=1}^{n} cost(C_i, m, f) \tag{17}$$

Stacking requires a further step of meta-level training. The data for the next level of training are obtained with a process similar to k-fold cross-validation. The rows of the meta-level data are m. The number of meta-level features is n for Stacking with class labels and nc for Stacking with probability distributions. Therefore the cost of Stacking is:

$$k \sum_{i=1}^{n} cost(C_i, m, f) + cost(MC, m, nc) \tag{18}$$

where MC is the meta-classifier. This extra cost of learning might be small compared to the previous $k+1$ training processes, although it depends on the learning algorithm used for MC and the number of base-level algorithms and classes. Some algorithms have square complexity with respect to the number of features. Therefore if $nc >> f$, then the cost of meta-training could dominate the total cost of Stacking.

Another known problem of Stacking that shows up with this analysis is the high-dimensionality of the meta-level training data for problems with many classes and many base-level classifiers. Stacking will have a severe problem to generalize well from the meta-level training data for a large ensemble size.

6.2 Predictive Performance

Figure 1, shows the average geometric mean of the error of the participating methods (vertical axis) on all data sets and experiments, using different number of classifiers (horizontal axis). SF denotes Selective Fusion and is the average geometric mean for all six Selective Fusion approaches. We start with the 10 classifiers and in turn remove one algorithm, the one with the largest error rate on the test sets, until only 3 remain. By removing the algorithm with the largest error rate, we are actually being less fair to Selective Fusion and Evaluation and Selection that can discard algorithms of poor performance automatically.

Figure 1: Mean error of combining methods with a varying ensemble size

As far as the standard methods are concerned, we notice that Weighted Voting is consistently better than Voting and that Stacking with Multi-Response Model Trees is consistently better than both. An interesting result is the performance of Evaluation and Selection. For ensemble sizes of 3 to 5 classifiers, it is better than the rest of the standard methods and especially in the beginning it has the lowest error from all methods. For 6 classifiers and more it's performance starts to degrade and for 8 and 9 classifiers it becomes the method with the highest mean error. One should expect ideally that the performance of Evaluation and Selection will never decrease with ensemble size. However with more (and more noisy) classifiers in the ensemble, the probability of selecting the most suitable classifier decreases and that of accidentally selecting a much worse one increases. In addition, Evaluation and Selection fails to benefit from the diversity of the different algorithms by selecting only one of them.

The most interesting result is that Selective Fusion is consistently the best method for combining more than 3 classifiers. Although the performance of Stacking consistently increases with the addition of classification algorithms, it does not increase with the same rate as the performance of Selective Fusion. As mentioned in the previous subsection, the problem with Stacking is the high dimensionality of the meta-data for many classifiers.

Overall, the best results are obtained with large numbers of classifiers, with the best geometric mean obtained for 9 classifiers by Selective Fusion. This shows that in order to obtain the best possible performance from a heterogeneous ensemble, one should equip it with many diverse algorithms in order to ensure that some of them will be appropriate for this data set and at the same time exclude the ones not suitable using a process like the one proposed in this paper.

7 Conclusions and Future Work

This paper has presented Selective Fusion, an approach for combining different classification algorithms that uses statistical methods to select the most accurate subgroup among an ensemble of classifiers and then combines their decisions through weighted voting. Through a large and thorough experimental study we showed that the proposed method isn't worse in accuracy to recent state-of-the-art heterogeneous classifier combination methods, such as Stacking with Multi-Response Model Trees having at the same time reduced computational cost.

Selective Fusion is actually a generalization of selection and fusion as it contains the simple methods of Evaluation and Selection and Weighted Voting as special cases. Indeed, if for a data set a single algorithm is by far the best then Selective Fusion will use just this, as would Evaluation and Selection. In another data set where all algorithms exhibit the same performance, with no algorithm being significantly better than another, then Selective Fusion becomes the method of Weighted Voting.

The concept of using statistical tests for selecting the best subset of algorithms, could be considered as a pre-processing step for any fusion method. Such an approach could also be applied to homogeneous models, for filtering of potentially bad-performing models produced by procedures, such as bagging and randomization [9].

As a general conclusion, we believe that it is worth researching more into advancing simple heterogeneous ensemble methods such as evaluation and selection, voting and weighted voting instead of complex methods that require more input parameters and a lot more computational cost with questionable results.

For future work, we intend to research into alternative methods for classifier performance evaluation, in order to further improve the Selective Fusion framework. In addition we intend to investigate the applicability of the proposed ideas to more complex Classifier Evaluation methods such as those that are based on local accuracy estimates.

We would also like to explore the role that domain experts could play in the process of Selective Fusion. Classifier combination methods are generally complex and usually don't give explanations for their decisions. Therefore domain experts should participate in this process to validate these decisions. For example, in our case, domain experts could look into the group of algorithms that our method selected and decide on expanding or shrinking the group based on their expertise on the specific domain and algorithms.

Acknowledgements

The authors would like to thank the anonymous referees for their valuable comments and suggestions on this work.

References

[1] D. Aha, D.W. Kibler, and M.K. Albert. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[2] H. Bensusan, C. Giraud-Carrier, and C. Kennedy. A higher-order approach to meta-learning. In *ECML'2000 workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, 2000.

[3] C.M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[4] C.L. Blake and C.J. Merz. UCI repository of machine learning databases. http://www.ics.uci.edu/∼mlearn/MLRepository.html, 1998.

[5] P.B. Brazdil, C. Soares, and J.-P. Da Costa. Ranking Learning Algorithms: Using IBL and Meta-Learning on Accuracy and Time Results. *Machine Learning*, 50:251–277, 2003.

[6] J.G. Cleary and L.E. Trigg. K*: An instance-based learner using an entropic distance measure. In *Proceedings of the 12th International Conference on Machine Learning*, pages 108–114. Morgan Kaufmann, 1995.

[7] W.W. Cohen. Fast effective rule induction. In *Proceedings of the 12th International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann, 1995.

[8] T. G. Dietterich. Machine-learning research: Four current directions. *AI Magazine*, 18(4):97–136, 1997.

[9] T. G. Dietterich. Ensemble Methods in Machine Learning. In *Proceedings of the 1st International Workshop in Multiple Classifier Systems*, pages 1–15, 2000.

[10] S. Dzeroski and B. Zenko. Is Combining Classifiers with Stacking Better than Selecting the Best One? *Machine Learning*, 54:255–273, 2004.

[11] G. Giacinto and F. Roli. Adaptive selection of image classifiers. In *Proceedings of the 9th International Conference on Image Analysis and Processing*, pages 38–45, 1997.

[12] T.K. Ho, J.J. Hull, and S.N. Srihari. Decision combination in multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(1):66–75, 1994.

[13] Y. Hochberg and A. C. Tamhane. *Multiple comparison procedures*. Wiley, 1987.

[14] J. C. Hsu. Constrained simultaneous confidence intervals for multiple comparisons with the best. *Annals of Statistics*, 12:1136–1144, 1984.

[15] G.H. John and P. Langley. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Francisco, 1995. Morgan Kaufmann.

[16] A. Kalousis and T. Theoharis. Noemon: Design, implementation and performance results of an intelligent assistant for classifier selection. In *Intelligent Data Analysis*, 1999.

[17] J. Keller, I. Paterson, and H. Berrer. An integrated concept for multi-criteria ranking of data mining algorithms. In *ECML-00 Workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selction and Method Combination*, 2000.

[18] R. Kohavi. The power of decision tables. In *Proceedings of the 12th European Conference on Machine Learning*, pages 174–189, 1995.

[19] M. Kubat, R. Holte, and S. Matwin. Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, 30:195–215, 1998.

[20] C. J. Merz. Dynamical selection of learning algorithms. In D. Fisher and H. J. Lenz, editors, *Learning from Data: Artificial Intelligence and Statistics*. Springer-Verlag, 1995.

[21] B. Pfahringer, H. Bensusan, and C. Giraud-Carrier. Meta-learning by landmarking various learning algorithms. In *International Conference on Machine Learning 2000*, 2000.

[22] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1998.

[23] R.J. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufman, San Mateo, 1993.

[24] C. Schaffer. A conservation law for generalization performance. In *Proceedings of 11th International Conference on Machine Learning, ICML94*, pages 259–265, 1994.

[25] A. J. Scott and M. Knott. A cluster analysis method for grouping means in the analysis of variance. *Biometrics*, 30:507–512, 1974.

[26] K.M. Ting and I.H. Witten. Issues in stacked generalization. *Journal of Artificial Intelligence Research*, 10:271–289, 1999.

[27] G. Tsoumakas, I. Katakis, and I. Vlahavas. Effective Voting of Heterogeneous Classifiers. In *Proceedings of the 15th European Conference on Machine Learning, ECML2004*, pages 465–476, September 2004.

[28] I.H. Witten and E. Frank. Generating accurate rule sets without global optimization. In *Proceedings of the 15th International Conference on Machine Learning, ICML98*, pages 144–151, 1998.

[29] I.H. Witten and E. Frank. *Data Mining: Practical machine learning tools with Java implementations.* Morgan Kaufmann, 1999.

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

[31] K. Woods, W. P. Kegelmeyer, and K. Bowyer. Combination of multiple classifiers using local accuracy estimates. *IEEE Transanctions on Pattern Analysis and Machine Intelligence*, 19(4):405–410, 1997.