View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by University of Szeged

Acta Cybernetica **19** (2009) 41–60.

3-level Confidence Voting Strategy for Dynamic Fusion-Selection of Classifier Ensembles

Csaba Főző and Csaba Gáspár-Papanek[*]

Abstract

There are two different stages to consider when constructing *multiple classifier systems*: The *Meta-Classifier Stage* that is responsible for the combination logic and basically treats the ensemble members as black boxes, and the *Classifier Stage* where the functionality of members is in focus. Furthermore, on the upper stage - also called voting strategy stage - the method of combining members can be done by fusion and selection of classifiers. In this paper, we propose a novel procedure for building the meta-classifier stage of MCSs, using an *oracle* of three-level voting strategy. This is a dynamic, half fusion-half selection type method for ensemble member combination, which is midway between the extremes of fusion and selection. The MCS members are weighted and combined with the help of the oracle, which is founded on a voting strategy of three levels: (1) The Local Implicit Confidence (LIC), (2) The Global Explicit Confidence (GEC), and (3) The Local Explicit Confidence (LEC). The first confidence segment is dependent of classifier construction, via the implicit knowledge gathered simultaneously with training. Since this strongly depends on the internal operation of the classifier, it can not always be obtained, for example, when using some particularly complex classification methods. We used several, known classifier algorithms (Decision Trees, Neural Networks, Logistic Regression, SVM) where it is possible to extract this information. The second goodness index is calculated on the validation partition of the labeled train data. It is used to obtain the general accuracy of a single classifier using a data set independent of the training partition. And finally, the third part of the confidence triplet depends also on the unlabeled objects yet to be classified. Due to this, it can only be calculated in classification time.

Keywords: multiple classifier systems, supervised learning, classifier ensembles, voting strategies, confidence based voting, two-staged ensemble

[*]Budapest University of Technology and Economics, Department of Telecommunication and Media Informatics, Telephone: +36-1-463-2225, Fax: +36-1-463-3107, E-mail: `{csaba.fozo, gaspar}@tmit.bme.hu`

DOI: 10.14232/actacyb.19.1.2009.4

1 Introduction

One of the most promising ways of fighting the challenge of *supervised learning* in artificial intelligence, or classification as it is mostly known in the lingo of data mining, is using *classifier ensembles*. Classification is the most significant type of analysis in data mining [18], and thus an active area of research as well. More than 10 years ago a new performance increasing technique emerged by combining classifiers. Many studies have been published about the advantages of the paradigm over the individual classifier models [5, 14]. This seems to be a quite easy and widespread way of further increasing classification accuracy, although, beyond the basic idea there are also several new challenges of key importance, like preserving ensemble members' accuracy while increasing their diversity [11].

The aim of this paper is to present a new combination method for *Multiple Classifier Systems* (MCS) [16, 17, 13] to further increase accuracy of classifier ensembles. Thus, this novel method is applicable to any MCS, that uses a separated algorithm to handle the MCS members' individual results as inputs, and provides a single, final result as an output. Our method treats the underlying single classifiers as black boxes. The only requirement for the individual classifiers is to be able to provide an implicit confidence value measuring the sureness of their decision. By dividing ensembles to a classifier and a meta-classifier stage, it's possible to build a pure meta-classifier stage algorithm, such as the 3 level confidence voting method, that has the advantages of transportability.

Our method has three levels of confidence, GEC, LIC and LEC. In understanding the importance of each level, a parallel can be drawn between this voting technique and a group of specialists at a meeting. The process of classification would be paired with a discussion in the example. In case of a single classifier, the group would consist of only one specialist, who makes a decision. With more than one members, a chairman has the final word. The chairman would be the voting algorithm. The more information it has about the members, the easier to make the best final decision. In the three-level voting method, this information is materialized in the three levels. The global explicit level refers to a statistical information about the given specialist's general accuracy when making decisions, based on past situations observed by the chairman. The local implicit level is a self-confidence-like value, describing how sure he really is about his answer, which can have a significant influence on how persuasive he is at convincing the chairman. It's not enough to know the correct decision, but we have to know, how sure he is about it, e.g. when two, usually very accurate specialists decide differently, the chairman has to size up the situation taking into account which one of them is more confident. And finally the local explicit level weighs the specialists by separating the question space into fields and determine how accurate the specialist generally is on the field of the given matter statistically. This way it helps identifying the level of diversity between MCS members, and assigning greater weights to the ones that are having more expertise on the field.

The procedure is similar to [8], with the difference that we are interested in precise confidence values to select and weight members' decisions in the combination, instead of the best single classifier for an unlabeled data point. As we could see in the example of the previous paragraph, neither of the three confidence levels can be omitted. Even with perfectly specialized members and complementary expertise, we have to take into consideration that the field of the question, the given subject matter can not be identified perfectly. The classifiers wouldn't make a good decision, if the subject matter is, for example, near the boundary of two specialists expertise. In this case, using only the local confidence can give a quite inaccurate weighing. Solving the problem is even harder in real situations, when specialist's expertise is always overlapping, and nor the matters can be clearly assigned to a specialist, neither the real expertise on the field can be derived. The error of locating the given subject can distort the measured LEC value substantially. In conclusion, the imperfection of finding the question's local region results that, at these hard cases, a good concept to follow is to complete local confidence with a general decision accuracy, such as the global explicit confidence. In fact, if we only wanted to select the best specialist to make the decision, like Giacinto et al. [8], this wouldn't be such an important matter, however, our approach can provide significant improvement in classification accuracy.

The basic idea of combining these three levels of confidence is similar to that used at building classifier ensembles instead of individual classifiers. We combine several confidence values of each single classifier, to determine the best weighing amongst them, resulting the best combination. It is like using an other ensemble system to support the whole classifier ensemble. The three-level voting technique applied to an ensemble of classifiers can be described as an MCS with two ensembles, one traditional ensemble of the classifier stage, and a second one of the meta-classifier (voting) stage. The first one summarizes the members' decisions, and the second one summarizes the members' confidence indices to provide weights for the first one.

The following section offers a brief oversight of classifier ensembles and a possible categorization. Section 3 presents the three-level confidence voting technique. The dataset used for experimental testing and the results are reviewed in Section 4. And finally Section 5 concludes the paper by summarizing retrieved experiences, furthermore outlining possible directions of future work.

2 Classifier Ensembles

One of the main reason of building classifier ensembles was that many classifiers that were considered weak was found to keep a significant part of their capabilities hidden if applied individually. Even the most simple algorithms, like decision trees' performance can be boosted by using a properly collected ensemble [19, 13]. The fundamental thought is that the precision of an individual classifier can be superseded by the diversity of a classifier ensemble. The key idea is similar to the theory of division of labour. In some ways, diversity is identifiable with specialization [15, 11].

Definition 1 (Classifier Ensemble). *Let $\overline{x} = [x_1, ..., x_N]$ be a data point described by N features and let $\boldsymbol{X} = [\overline{x}_1, ..., \overline{x}_n]^T$ be the training dataset in a form of an $n \times N$ matrix. Let $\overline{y}_{\boldsymbol{X}} = [y_1, ..., y_n]^T$ be a vector of class labels for the training data, where y_i takes a value from $C = \{l_1, ..., l_c\}$ class label set. Let L be the number of classifiers in the ensemble denoted by $D_1, ..., D_L$. $D_i(\boldsymbol{X}, \overline{y}_{\boldsymbol{X}}, \overline{x}) = d_i$ is the decision of D_i classifier ensemble member, taking a value from C. Then E is a Classifier ensemble, if*

$$E = \{D_1, ..., D_L, V\}, \tag{1}$$

where

$$V = V(d_1, ..., d_L) \tag{2}$$

is the voting logic upon the D_i ensemble members.

2.1 Categorization

Many dimensions can be used to categorize classifier ensemble systems. Different approaches are known on how classifiers should be collected (1.), what are the construction terms (2.), in what manner should they work together (3.), and how the final output of the ensemble should be created based upon the members' decisions, i.e. the voting strategy (4.).

The member collection can be separated mainly into two large groups. In the first segment, the same kind of classifiers are used in the ensemble, and the diversity factor depends totally on the different methods of training the model. $E = \{D_1, ..., D_L, V\}$, where $D_i = D(X_i, Y_i, x)$. On the other hand, the dissimilarity of the ensemble can be largely enhanced by selecting members, based on different core algorithms. In certain terminologies, only the firstly mentioned collection is called a classifier ensemble, while the lingo of data mining is still unifying in this area, thus, from a more general view-point, any multiple classifier system (containing more than one classifier) can be called this way.

The terms of construction also consist of two possible directions. Considering dependencies between ensemble members, the single classifiers can be independent, hence the system can be built concurrently. While if there are dependencies, the construction has to be sequential.

Working manner is typically partitioned into two complementary approaches [10]: *Selection* and *fusion*. Selection follows the assumption that the members are having complementary competence on the feature space, thus in the divided space, they can be more specialized and accurate. While the underlying idea of classifier fusion is that the members are experts of the whole feature space. Due to diversity, the ensemble members may misclassify different input objects, thus a consensus type voting will filter errors. A mixture of the two methods is also possible.

Voting strategies can be various and can be heavily dependent from the above mentioned techniques also. Consensus type voting is generally used with fusion working manner. The simplest way is balanced majority voting. The weights used can be derived by model validation, when each member's accuracy is measured by detaching a validation segment from the training data and using it to score

Table 1: References of ensemble systems

	1		2		3		4			
	1.a	1.b	2.a	2.b	3.a	3.b	4.a	4.b	4.c	4.d
AdaBoost [7]	X			X	X			X		
MultiBoost [20]	X		X		X		X			
Bagging [4]	X		X		X		X			
RndForest [6]	X		X		X		X			
RndSubspace [9]	X		X		X		X			
RotForest [19, 13]	X		X		X		X			
RndLinOracle [12]		X	X		X	X				X
3-LevelVoting	-	-	-	-	X	X		X	X	X

aggregated performance of the particular classifier (e.g. Global explicit confidence). On the other hand, it is possible to adjust the before mentioned weights based upon the member itself, by attaching a confidence value to each of its decisions (e.g. Local implicit confidence). And finally, some systems use a higher authority called oracle to calculate members' expertise based upon the object to be classified, making uneven merging of member outputs, or even totally excluding some members (e.g. Local explicit confidence).

The uprising dimensions for characterizing ensemble systems are summarized in a list below and well-known algorithms with references are assigned to the classes in Table 1. Later, we will refer back to this enumeration to differentiate the analyzed methods to ease recognition of trends and possibly enticing fields of future work.

1. Member selection
   a) Same kind of classifiers (one learning method)
   b) Different classifiers (several learning methods)
2. Construction
   a) Concurrent
   b) Sequential
3. Working manner
   a) Selection based
   b) Fusion based
4. Voting strategy
   a) Balanced majority voting

b) Validation based average confidence

c) Self based confidence

d) Oracle type

In the following section, we apply the upper mentioned categorization on the *three-level confidence voting* as it's summarized in the last row of Table 1, and we give a detailed description of the algorithm and the three levels' logic.

3 The Three-Level Confidence Voting

As it was stated above, the three level voting technique is a meta-classifier stage algorithm. Thus, it is separable from the classifier stage, resulting categories 1 and 2 independent of our algorithm. On the 3. dimension of the system, our method falls within both 3.a and 3.b categories, since either of them can be applied. And finally, our voting strategy is a novel mixture of 4.b, 4.c, and 4.d types.

The general voting logic upon the D_i ensemble members in (2) is partly modified here, due to more than one confidence metric is used. Hence,

$$V = V_{3-level}(V_{LIC}(D_1, ...D_L), V_{GEC}(D_1, ...D_L), V_{LEC}(D_1, ...D_L)), \tag{3}$$

where *LIC* stands for Local Implicit Confidence, *GEC* for Global Explicit Confidence, and *LEC* for Local Explicit Confidence. First we demonstrate the details of the three levels, and then the $V_{3-level}$, that combines them.

3.1 Local Implicit Confidence (LIC)

The 'local' keyword means that this is a data point specific measure, that describes the data yet to be classified, instead of describing the classifier as a whole. Thus, it can be different for each test data point. Furthermore, it is measured implicitly, inside the classifier, bearing the information of both the model, and the test data. Most of the classifier algorithms are able to provide an index like this by the trained model and the given test data. With others that are not, some modifications have to be applied first. In this paper, we only consider classification algorithms that are able to produce this measure (Decision Trees, Neural Networks, Logistic Regression, SVM). While training the model, numerous attributes, weights are set by the classifier algorithm inside the model. These influence the model's accuracy as well, but their primary objective is to predict classification's expectable correctness, solely based on training experiences. Thus, when a classifier gives a decision, e.g. the label of data point x is $D_i(x) = l_j$, then it assigns a probability to this decision as well, e.g. $LIC_{D_i,x} = conf_{D_i(x)=l_j} = 70\%$ that discribes the classifier's certainty that it is the good decision.

3.2 Global Explicit Confidence (GEC)

The GEC is greatly different from the previous index. The global explicit level is a statistical information about the given, individual classifier's general accuracy.

Global refers to the generality, which means that it is a feature of the classifier, and it is identical for all incoming data to be classified. Explicit refers to the measuring method's independence from the classifier's inner processes. Practically, it's the probability of making a correct decision, based on the average accuracy reached on the validation segment of the labeled data. After training has finished, the trained model is tested on validation data to determine the model's accuracy on an input different from the training data. This way, the model's real accuracy can be estimated much more precisely than by performance measured on the same data that it was trained on. The specific algorithm for calculating the GEC always depends on the dataset and the task to resolve. Otherwise, we would not be able to compare the results of the MCS and the individual classifiers.

3.3 Local Explicit Confidence (LEC)

The local explicit confidence, considering it's features, is somewhere between the above presented two. Practically, it's the accuracy measure describing the classifier, and the test data points as well. The latter part is somewhat similar to LIC, although it's statistically calculated, independently from the classifiers inner processes, which means that it's similar to GEC. The basic idea is to locate the training data points that are similar to the data point to be classified, and calculate the classifiers' global explicit confidence upon this set of data points. Hence, it provides an accuracy that describes the local region of the test data point. So it depends on the test data point, because it's needed to gather the local region.

First, the algorithm locates the $\overline{x}_1, ..., \overline{x}_k$ nearest neighbors (k-NN), also called the local region of $\overline{z}$ unlabeled object in the feature space, with the help of an appropriate distance metric over data objects. Naturally, this metric is always strongly dataset dependent. In case of a normalized dataset, where data points' features are numeric values without further information, using an Euclidean distance metric proves to be a good choice. In other cases, the selection has to be considered more thoroughly.

In the next step, we count the $\overline{x}^{MCB} = [d_1(\overline{x}), ..., d_L(\overline{x})]$ Multiple Classifier Behaviour [21], where $d_j(\overline{x}) = D_j(\mathbf{X}, \overline{y}_{\mathbf{X}}, \overline{x})$, containing the ensemble members' decisions, for each $\overline{x}_i$ nearest neighbor, and for the $\overline{z}$ object to be classified. The local region is further narrowed by selecting the ones, having an MCB similar to the one $\overline{z}$ has. Afterwards, the remaining m objects' MCBs ($m \leq k$) are used to calculate the ratio of correct decisions for each single classifier, resulting the local explicit confidence.

In this case, similarity has a greater importance than in the feature space, because in the MCB space, the dimensions are far from being normalized. Even if the confidences (LICs) in these dimensions are normalized, it is hard to compare LIC values of different individual classifiers. This means that the same distance would mean a different numeric threshold in the dimensions that those individual classifiers are responsible for. For instance, in DT algorithms, the typical distance between LICs are 10 or 100 times greater, than in NN based classifiers. The solution is to normalize each dimension using the average distance of LICs. This average is calculated for each classifier D_i, and then for all x_j neighbors the $d_i(\overline{z}) - d_i(\overline{x_j})$ distance gets divided by it.

3.4 Combination of the Three Levels

The implicit level and the two explicit levels are representing classifier accuracy from different aspects, due to the dissimilarity of the classifier's inner process based (implicit) and the statistically measured (explicit) approach. Meanwhile, the two explicit levels are more similar. Beyond the usage of implicit measures, the application of two explicit indices is one of the most important difference between our 3-level technique and dynamic classifier selection of Giacinto et al. [8]. Our aim is to rank the classifier ensemble members as precisely as possible, weight them, and combine those that have the potential to improve the whole ensemble's decision. We do not presume that the best decision at classifying a data point can be made by only one classifier. In our approach, we assume that all members are responsible for the whole feature space, instead of just partitions of it. A well-chosen *selection threshold* is used to decide whether the given classifier should be involved in making the decision for a given data point or not. Thus, it is a fusion-selection type method, having the advantage of being more flexible.

Our task is to assign weights to all ensemble members, so that their weighted and summed decisions result a more accurate final decision. Any of the three confidences can be used as weights. Although, using all three confidence levels, and also in a selection-fusion manner is advisable. This enables the method to adapt at the same time to data points (1) that are very different from the training set, (2) that fall within more than one of the ensemble members' expertise, thus all these members can classify them with good accuracy, and (3) that don't clearly fall within any of the classifiers expertise, based on the locality computations.

For instance, a simple classifier selection chooses the best member of the ensemble to make the decision. Practically it is the member, that classified most data points correctly in the local region. The assumption of (1) results that the data points forming the local region will be highly different from the data point to be classified. Thus, the best member will not be able to classify the data point correctly. This means that the problem of encountering dissimilarities between the data point and its local region escalates also to the classifier selection. All in all, in this case it is highly probable, that a different ensemble member is selected, instead of the best one for the data point to be classified. With the help of our multi level technique, it's possible to detect these situations. If the local region becomes practically empty or highly unreliable because of these dissimilarities, our method uses the GEC instead of the LEC.

The second type of problematic data points are less critical, since in these cases even only one member is able to make the good decision with relatively high confidence. In (2), again the challenge is that the local region can not be determined perfectly. Hence, weights calculated for members of the ensemble, based on this neighborhood does not really guarantee a clear ranking among the several classifiers that are all have good expertise in that local region. Picking one that seems to be the best can be a defective choice. This is the typical case of classifier fusion overshadowing selection. Considering a subset of generally accurate individual classifiers with divers errors, our t_{sel} selection threshold enables each of these members, due to their high performance, and fusion will filter out errors.

And finally, solving the third problematic case is the hardest using only selection methods. When non of the individual members could make the right decision on their own. Choosing amongst almost equally bad classifiers clearly would not be a good idea. Instead, by identifying those that have a recognizable confidence and combining their weighted decisions, the result can be much more satisfiable. Moreover, the weighting can be further improved by combining the LEC and the GEC, assuming that some accuracy differences between members were masked by local region computation error.

LEC and GEC are utilizing the LIC values, as it was described in Sections 3.2 and 3.3. However, the LIC index can be used also to complete the above described weightings. Practically, we can regard it as the basic probability of the decision's correctness. It is a much more sophisticated measure to use, instead of weighting only the decisions.

Now we demonstrate the three-level voting method by and example. Assuming the ensemble has two members ($L = 2$), a binary classification task ($C = 2$), 10 training data points ($n = 5$), and 1 unlabelled test data point ($\overline{z}$). The members are trained (black-boxes), their LIC values and decisions are given in Table 2 by a combined LIC metric often used at binary classification tasks ($LIC^{comb} \in [0, 1]$). This index describes the decision and its confidence as well. The closer this value is to 0 or 1, the higher the confidence is. Thus a value of 0.5 would mean a random decision with zero confidence. From now on, we use LIC in the example as a shorter name that stands for LIC^{comb}. Then we calculate the GEC for both classifiers ($i = 1, 2$). Let the evaluation method be the number of correct decisions.

$$GEC_{D_i} = \frac{\sum_{j=1}^{n} \frac{sgn(0.5 - \left|LIC_{D_i, \overline{x}_j} - y_j\right|)}{2} + 0.5}{n},$$

assuming that $sgn(0) = 0$.

$$GEC_{D_1} = \frac{3}{5} = 0.6, \tag{4}$$

$$GEC_{D_2} = \frac{2}{5} = 0.4, \tag{5}$$

To calculate the LEC, that also uses LIC, first, we have to gather the neighborhood $X_{nb}(\overline{z})$ of the test point. First, k-NN is calculated from the N features of $\overline{x}_5$ and $\overline{z}$, where k is a parameter of the algorithm (now $k = 3$). The resulting training dataset is given in Table 3. The second step is the MCB filtering of $X_{nb}(\overline{z})$.

$$X_{nb}^{MCB}(\overline{z}) = \left\{ \overline{x}_i \in X_{nb}^{k-NN}(\overline{z}) | dist_{MCB}(\overline{x}_i) \leq t_{MCB} \right\},$$

where the t_{MCB} threshold is a parameter of the algorithm. The MCB distance is calculated with the above mentioned MCB normalization method, by the $dens_{D_i}$

average LIC density of classifiers (Section 3.3).

$$dist_{MCB}(\overline{x}_j) = \sqrt{\sum_{i=1}^{L}\left(\frac{LIC_{D_i,\overline{x}_j} - LIC_{D_i,\overline{z}}}{dens_{D_i}}\right)^2}$$

$$dens_{D_i} = \frac{\sum_{j=1}^{n-1}\left|LIC_{D_i,j}^{sorted} - LIC_{D_i,j+1}^{sorted}\right|}{\sum_{j=1}^{n-1} sgn\left|LIC_{D_i,j}^{sorted} - LIC_{D_i,j+1}^{sorted}\right|},$$

assuming that $sgn(0) = 0$.

$$dens_{D_1} = \frac{0.01 + 0.02 + 0.03}{3} = 0.02 \tag{6}$$

$$dens_{D_2} = \frac{0.1 + 0.1 + 0.1 + 0.3}{4} = 0.15 \tag{7}$$

$$dist_{MCB}(\overline{x}_2) = \sqrt{\left(\frac{0.07 - 0.01}{0.02}\right)^2 + \left(\frac{0.5 - 0.45}{0.15}\right)^2} = 3.02 \tag{8}$$

Results of MCB distance calculations for the whole k-NN local region (X_{nb}^{k-NN}) is shown in Table 3. In the example, we choose our second parameter $t_{MCB} = 2.5$, hence $X_{nb}^{MCB} = \{\overline{x}_4, \overline{x}_5\}$.

Table 2: Combined LIC values of the example

X	LIC_{D_1}	LIC_{D_2}	y
$\overline{x}_1$	0.01	0.2	0
$\overline{x}_2$	0.07	0.5	1
$\overline{x}_3$	0.01	0.3	0
$\overline{x}_4$	0.04	0.8	0
$\overline{x}_5$	0.02	0.4	1
$\overline{z}$	0.02	0.4	?

Table 3: k-NN filtered data point's combined LIC values of the example

$X_{nb}^{k-NN}(\overline{z})$	LIC_{D_1}	LIC_{D_2}	y	$dist_{MCB}$
$\overline{x}_2$	0.07	0.5	1	3.02
$\overline{x}_4$	0.04	0.8	0	2.32
$\overline{x}_5$	0.02	0.4	1	2.03
$\overline{z}$	0.01	0.45	?	0

The next step in deriving LEC is to calculate the average error rate of the ensemble members in this local region. The average distance of this error rate and the $LIC = 1$ value yields the confidence index we are looking for (higher confidence at lower error rate).

$$LEC_{D_i,\overline{z}} = \frac{\sum_{\forall j, \overline{x}_j \in X_{nb}^{MCB}} \left(1 - \left|LIC_{D_i,\overline{x}_j} - y_j\right|\right)}{\left|X_{nb}^{MCB}\right|},$$

if $\left|X_{nb}^{MCB}\right| \neq 0$. Generally, we produce a small neighborhood, and thus derive the LEC values quite strictly. Due to this, X_{nb}^{MCB} is a fairly small set, the numerosity is zero quite often. In this case, ensemble members' weight is determined by the GEC as we stated previously.

$$LEC_{D_i,\overline{z}} = GEC_{D_i},$$

if $\left|X_{nb}^{MCB}\right| = 0$. We can see in Equation (9) and (10), that neither of the individual classifier members of the ensemble has a convincing confidence based on the neighborhood.

$$LEC_{D_1,\overline{z}} = \frac{(1 - 0.04) + (1 - 0.98)}{2} = 0.49 \qquad (9)$$

$$LEC_{D_2,\overline{z}} = \frac{(1 - 0.8) + (1 - 0.6)}{2} = 0.3 \qquad (10)$$

The third parameter of our algorithm is the $t_{sel-fus}$ selection-fusion threshold. It is used to determine the minimum LEC value of D_i members that are selected and considered for fusing. We chosen $t_{sel-fus} = 0.25$ in our example. The resulting $TLV(\overline{z})$ probability that serves as the output of our method is calculated in Equation (11). Naturally, at least one member always has to be collected, otherwise no weight could be assigned. The TLV index of our Three-Level Voting strategy then derived using all three confidence measures (GEC is used implicitly in LEC).

$$TLV(\overline{z}) = \frac{\sum_{\forall i, D_i \in D_{sel}(\overline{z})} (LIC_{D_i,\overline{z}} \cdot LEC_{D_i,\overline{z}})}{\sum_{\forall i, D_i \in D_{sel}(\overline{z})} LEC_{D_i,\overline{z}}}$$

$$D_{sel}(\overline{z}) = \{D_i | LEC_{D_i,\overline{z}} \geq t_{sel-fus}\}$$

$$TLV(\overline{z}) = \frac{0.01 \cdot 0.49 + 0.45 \cdot 0.3}{0.49 + 0.3} = 0.18 \qquad (11)$$

Since D_1 is proved to be more accurate than D_2 on neighboring data points, we can see that $LIC_{D_1,\overline{z}}$ is stressed in (11), compared to the LIC of D_2.

4 Results

To measure the three-level confidence voting technique's accuracy and performance, we aimed to use a dataset widely available on the Internet. Furthermore, we wanted to prove the expected performance in a manner not only reproducible, but immediately comparable as well. Hence, we decided to test the method on the dataset of ACM SIGKDD's KDD Cup 2008, the oldest and most famous among the data mining competitions [1, 2]. Before going into details about the results, we present this dataset [3], provided by Siemens Medical Solutions USA.

4.1 KDD Cup 2008

The competition's main task was to detect breast cancer at an early stage from X-ray images of the breast. This typically consists of 4 X-ray images, 2 images per breast that are taken from different directions, called MLO and CC. The data is preprocessed in 2 steps.

1. Candidate generation from suspicious regions of the X-ray images.
2. Feature extraction, that computes descriptive features for each candidate to represent them in the form of numerical values.

Our task is to classify candidates to be able to differentiate between malignant cancers from the rest of the candidates based on the given feature vector. The rate of prevalence is extremely low, only 0.5-1% of the patients have breast cancer. The entries are judged by the area under the FROC curve, in the clinically relevant region 0.2-0.3 false positives per image. The participants had to return a confidence score for every candidate that indicates the confidence of their classifier that the candidate is malignant.

The training data consists of 102294 candidates from 118 malignant, and 1594 normal patients, each described with 117 features, and also an information segment about it's source image, patient, candidate's coordinates on the image, and a class label indicating whether or not it is malignant. The features are computed from several standard image processing algorithms, but there are no additional proprietary features.

4.2 Comparison of Results

First of all, we're going to introduce the individual classifier that are used to form ensembles and demonstrate their individual performance in the above mentioned classification challenge. 17 classifiers are tested including Neural Network (IDs beginning with 4), Support Vector Machine (IDs beginning with 5), Logistic Regression (IDs beginning with 1 or 3) and Decision Tree (IDs beginning with 2) algorithms. The ID 1 was assigned to the classifier with the best individual performance (later marked by GEC**), thus we can see that it was a logistic regression model. The goodness of the individual classifiers is shown in Table 4.

These confidence values were provided by the KDD Cup's evaluation method, which is based on the area under FROC, that is calculated from the LIC values of the validation partition of the dataset. The method' details are introduced on the web page of the competition [2]. So these individual classifier goodness indices

Table 4: Accuracy (GEC) of individual classifiers in the ensemble

ID(↓)	GEC	ID	GEC(↑)
1	76,79%	1	76,79%
21	55,91%	51	74,55%
22	55,90%	42	72,94%
31	71,98%	31	71,98%
32	63,91%	46	71,93%
33	62,06%	34	71,00%
34	71,00%	47	69,64%
35	69,06%	35	69,06%
41	59,17%	44	68,43%
42	72,94%	52	67,93%
43	56,81%	45	65,77%
44	68,43%	32	63,91%
45	65,77%	33	62,06%
46	71,93%	41	59,17%
47	69,64%	43	56,81%
51	74,55%	21	55,91%
52	67,93%	22	55,90%

are practically the GEC values. First, we are to demonstrate the performance potential in using only the GEC measure for voting, and finally, all three levels are combined and accuracy is tested as the two explicit indices (GEC, LEC) get integrated (Subsection 3.4), still on local implicit basis, forming our Three-Level Voting (TLV) strategy.

Given 17 individual classifiers, this would mean a tremendous amount of possible combinations. Hence, we decided to narrow down the problem space by constructing ensembles with only 2 members (classifier pairs). Evaluating only these pairs resulted 136 possible combinations. Several voting methods were considered, to give a wide range of possible comparison. The 136 ensembles are sorted by best ensemble accuracy reached among tested voting methods. The 5 best performances are listed in Table 5, referred also by ensemble members' IDs.

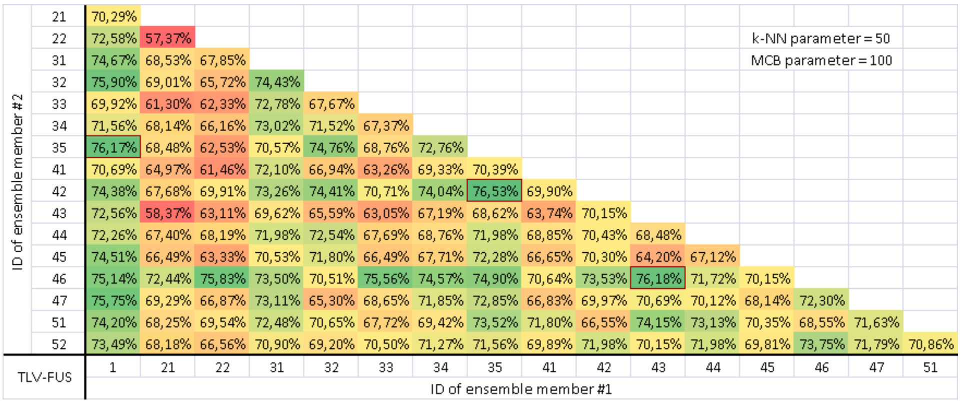
MAJ (majority voting), AVG (average), HC (highest confidence), MAX (maximum), MIN (minimum), GEC-AVG, TLV-SEL, TLV-FUS rows represent the voting methods. Majority voting is the only decision based method amongst the tested. Both classifiers have a vote, equivalent with their decision. In the end, the votes are averaged and the resulting ratio serves as the confidence value required in the evaluation. In the next three methods the combined values can be calculated with the $AVG(x_1, x_2)$, $MIN(x_1, x_2)$, and $MAX(x_1, x_2)$ functions, where x_i represents the LIC values of the two ensemble members.

Table 5: Performance of top 5 ensembles using different voting strategies

Ensemble	#1	#2	#3	#4	#5
ID1	1	1	1	1	1
GEC1	76,79%	76,79%	76,79%	76,79%	76,79%
ID2	35	32	31	42	52
GEC2	69,06%	63,91%	71,98%	72,94%	67,93%
Majority	34,59%	31,79%	38,49%	27,59%	52,32%
Avg	77,53%	76,96%	75,48%	76,38%	72,53%
HC	77,41%	78,88%	78,5%	78,48%	75,35%
Max	76,79%	76,79%	76,79%	76,79%	76,79%
Min	69,06%	65,93%	71,98%	72,95%	71,73%
GEC-AVG	77,53%	77,48%	75,7%	76,38%	72,42%
TLV-FUS	76,17%	75,9%	74,67%	74,38%	73,49%
TLV-SEL	79,63%	76,94%	77,08%	77,3%	78,38%
MaxMeth	2LV-SEL	HC	HC	HC	2LV-SEL
MaxGEC	79,63%	78,88%	78,5%	78,48%	78,38%
DiffMax	2,83%	2,09%	1,71%	1,69%	1,59%
OK	1	1	1	1	1

The highest confidence method selects the classifier, which was more confident in it's decision. So the output is the LIC value of the classifier with the highest confidence (the most distant from 0.5, considering *combined LIC*). GEC-AVG is almost the same as AVG, but here, the members are weighted by their GEC index before calculating the average of LIC values. The SEL and FUS versions of the TLV method stand for selection and fusion. Since we are combining classifier pairs, there's only 2 possible implementations of selection-fusion: pure selection (TLV-SEL) and pure fusion (TLV-FUS). Hence, parameter $t_{sel-fus}$ can be omitted. MaxMethod row contains the best method for the given ensemble (column) and MaxGec contains the best method's GEC value. DiffMax shows the precision compared to the best member's GEC value in the ensemble (GEC*). The OK flag value is equal to 1, if MaxGec is equal or higher than the GEC*.

In Table 5, we can see, that the best method amongst all was our TLV-SEL, that reached 79.63%, beating the GEC* by 2.83%. The GEC value of the best classifier amongst our 17 tested is marked by GEC**. Since in this case, the ensemble contained the best classifier of our 17 tested (thus GEC* = GEC**), our method was successful in increasing the overall performance compared to all our individual classifiers. The TLV-SEL algorithm is proved to be the best method also at #5 of the 136 ensembles besides #1, and the second best method at #3 and #4. At #2 it is only the forth, while GEC-AVG was second best, hence it is highly probable that the TLV parameters are far from perfect, otherwise GEC-AVG, that uses GEC, which is also a part of TLV could not be more accurate.



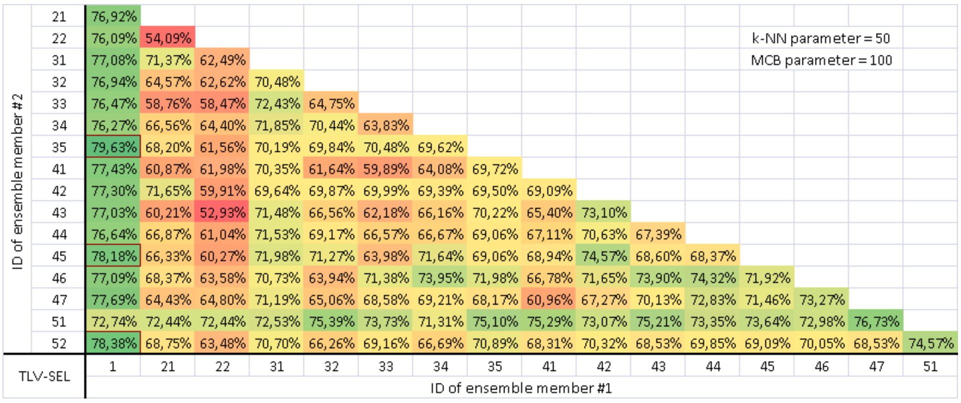
TLV-FUS	1	21	22	31	32	33	34	35	41	42	43	44	45	46	47	51
21	70,29%															
22	72,58%	57,37%														
31	74,67%	68,53%	67,85%													
32	75,90%	69,01%	65,72%	74,43%												
33	69,92%	61,30%	62,33%	72,78%	67,67%											
34	71,56%	68,14%	66,16%	73,02%	71,52%	67,37%										
35	76,17%	68,48%	62,53%	70,57%	74,76%	68,76%	72,76%									
41	70,69%	64,97%	61,46%	72,10%	66,94%	63,26%	69,33%	70,39%								
42	74,38%	67,68%	69,91%	73,26%	74,41%	70,71%	74,04%	76,53%	69,90%							
43	72,56%	58,37%	63,11%	69,62%	65,59%	63,05%	67,19%	68,62%	63,74%	70,15%						
44	72,26%	67,40%	68,19%	71,98%	72,54%	67,69%	68,76%	71,98%	68,85%	70,43%	68,48%					
45	74,51%	66,49%	63,33%	70,53%	71,80%	66,49%	67,71%	72,28%	66,65%	70,30%	64,20%	67,12%				
46	75,14%	72,44%	75,83%	73,50%	70,51%	75,56%	74,57%	74,90%	70,64%	73,53%	76,18%	71,72%	70,15%			
47	75,75%	69,29%	66,87%	73,11%	65,30%	68,65%	71,85%	72,85%	66,83%	69,97%	70,69%	70,12%	68,14%	72,30%		
51	74,20%	68,25%	69,54%	72,48%	70,65%	67,72%	69,42%	73,52%	71,80%	66,55%	74,15%	73,13%	70,35%	68,55%	71,63%	
52	73,49%	68,18%	66,56%	70,90%	69,20%	70,50%	71,27%	71,56%	69,89%	71,98%	70,15%	71,98%	69,81%	73,75%	71,79%	70,86%

ID of ensemble member #2

ID of ensemble member #1

k-NN parameter = 50
MCB parameter = 100

Figure 1: Performance of method TLV-FUS on 136 ensembles



TLV-SEL	1	21	22	31	32	33	34	35	41	42	43	44	45	46	47	51
21	76,92%															
22	76,09%	54,09%														
31	77,08%	71,37%	62,49%													
32	76,94%	64,57%	62,62%	70,48%												
33	76,47%	58,76%	58,47%	72,43%	64,75%											
34	76,27%	66,56%	64,40%	71,85%	70,44%	63,83%										
35	79,63%	68,20%	61,56%	70,19%	69,84%	70,48%	69,62%									
41	77,43%	60,87%	61,98%	70,35%	61,64%	59,89%	64,08%	69,72%								
42	77,30%	71,65%	59,91%	69,64%	69,87%	69,99%	69,39%	69,50%	69,09%							
43	77,03%	60,21%	52,93%	71,48%	66,56%	62,18%	66,16%	70,22%	65,40%	73,10%						
44	76,64%	66,87%	61,04%	71,53%	69,17%	66,57%	66,67%	69,06%	67,11%	70,63%	67,39%					
45	78,18%	66,33%	60,27%	71,98%	71,27%	63,98%	71,64%	69,06%	68,94%	74,57%	68,60%	68,37%				
46	77,09%	68,37%	63,58%	70,73%	63,94%	71,38%	73,95%	71,98%	66,78%	71,65%	73,90%	74,32%	71,92%			
47	77,69%	64,43%	64,80%	71,19%	65,06%	68,58%	69,21%	68,17%	60,96%	67,27%	70,13%	72,83%	71,46%	73,27%		
51	72,74%	72,44%	72,44%	72,53%	75,39%	73,73%	71,31%	75,10%	75,29%	73,07%	75,21%	73,35%	73,64%	72,98%	76,73%	
52	78,38%	68,75%	63,48%	70,70%	66,26%	69,16%	66,69%	70,89%	68,31%	70,32%	68,53%	69,85%	69,09%	70,05%	68,53%	74,57%

ID of ensemble member #2

ID of ensemble member #1

k-NN parameter = 50
MCB parameter = 100

Figure 2: Performance of method TLV-SEL on 136 ensembles

The algorithm uses two main parameters, the k index of the k-NN method, and the t_{MCB} threshold. Our technique's accuracy is highly dependent on these parameters. First, we tuned these two parameters using all 136 ensembles. This resulted a local optimum of $t_{MCB} = 100$, $k = 50$ that performed well on all ensembles. We included the results gathered by parameter setting in Table 5 presented previously. Considering computational expenses, we decided to run k-NN in a faster, approximative manner. This is done by searching k nearest neighbors only in a randomly chosen 1% of the training data points (approximatly 1000 data points). Meanwhile we kept all positive training samples in, to improve the rate of prevalence. The results of this setting is presented in Figure 1 for TLV-FUS, and in Figure 2 for TLV-SEL. The accuracy of the ensembles are demonstrated also by the fill color of the cell. The top three ensembles are marked by red bordering. The best ensemble for these settings was the pair of ID1 and ID35.

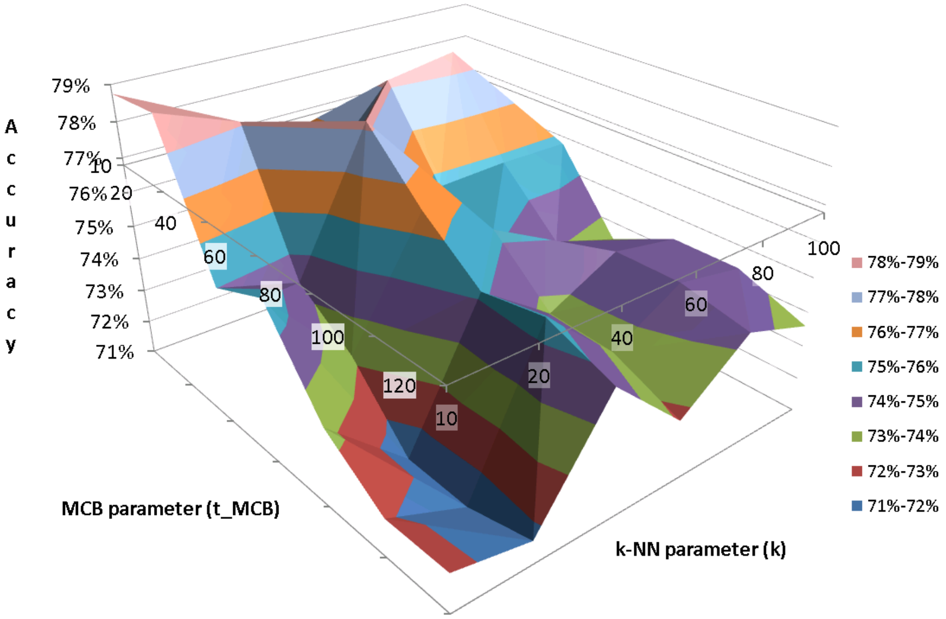
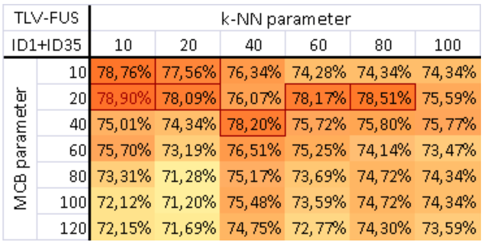


Figure 3: Trends in accuracy of ensemble ID1+ID35, taken as function of parameter settings



TLV-FUS		k-NN parameter					
ID1+ID35		10	20	40	60	80	100
MCB parameter	10	78,76%	77,56%	76,34%	74,28%	74,34%	74,34%
	20	78,90%	78,09%	76,07%	78,17%	78,51%	75,59%
	40	75,01%	74,34%	78,20%	75,72%	75,80%	75,77%
	60	75,70%	73,19%	76,51%	75,25%	74,14%	73,47%
	80	73,31%	71,28%	75,17%	73,69%	74,72%	74,34%
	100	72,12%	71,20%	75,48%	73,59%	74,72%	74,34%
	120	72,15%	71,69%	74,75%	72,77%	74,30%	73,59%

Figure 4: Numeric accuracy values of ensemble ID1+ID35, taken as function of parameter settings

Afterwards, we searched the optimal parameter settings considering this pair of classifiers (ensemble ID1+ID35). This time, we applied k-NN on 10% of the training points plus positive samples to increase k-NN precision. Figure 3 demonstrates the trends in accuracy of method TLV-FUS on a 3D surface, while Figure 4 is the same result presented in 2D with numeric values of accuracy assigned to each parameter setting. These are plotted also for method TLV-SEL in Figures 5 and 6. The floor of the 3D figures represents the GEC*. On the 2D figures, red bordering is applied where the accuracy of the ensemble at that parameter setting is greater then the GEC* hence the combination successfully improved performance. Furthermore,

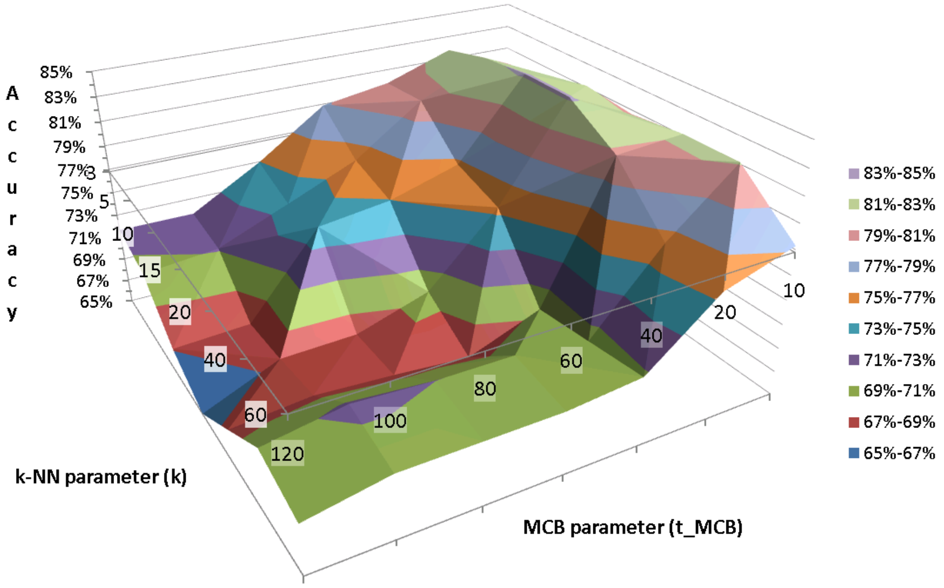
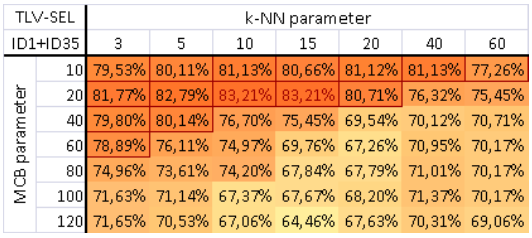


Figure 5: Trends in accuracy of ensemble ID1+ID35, taken as function of parameter settings



TLV-SEL		k-NN parameter						
ID1+ID35		3	5	10	15	20	40	60
MCB parameter	10	79,53%	80,11%	81,13%	80,66%	81,12%	81,13%	77,26%
	20	81,77%	82,79%	83,21%	83,21%	80,71%	76,32%	75,45%
	40	79,80%	80,14%	76,70%	75,45%	69,54%	70,12%	70,71%
	60	78,89%	76,11%	74,97%	69,76%	67,26%	70,95%	70,17%
	80	74,96%	73,61%	74,20%	67,84%	67,79%	71,01%	70,17%
	100	71,63%	71,14%	67,37%	67,67%	68,20%	71,37%	70,17%
	120	71,65%	70,53%	67,06%	64,46%	67,63%	70,31%	69,06%

Figure 6: Numeric accuracy values of ensemble ID1+ID35, taken as function of parameter settings

cells containing the highest accuracy are marked by red text. By method TLV-FUS, this accuracy is 78.9% at parameters ($k = 10; t_{MCB} = 20$), while by using TLV-SEL, it is 83.21% at parameters ($k = 10, 15; t_{MCB} = 20$). The performance improvement compared to GEC* (76.79%) is *2.11%* by TLV-FUS and *6.42%* by TLV-SEL. The method that reached second best ensemble accuracy considering all 136 ensembles was HC with 78.88% in Ensemble #2 (in Ensemble #1 HC only reached 77.41%). TLV-FUS managed to surpass the best performance of HC by 0.02%, while TLV-SEL overshadows it by *4.33%*.

We separated an independent, labeled data partition for final evaluation purposes, to be able to test the real accuracy of the best, optimized ensemble, with tuned parameters. For comparison, we also reevaluated the accuracy of our individ-

Table 6: Accuracy of our technique evaluated on an independent dataset

$t_{sel-fus}$	k	t_{MCB}	GEC^{TLV}	$GECDiff^{**}$	$GECDiff^{HC}$
fusion	10	20	78.91%	0.12%	-1.66%
selection	10	20	84.04%	5.25%	3.47%
selection	15	20	84.36%	5.57%	3.79%

ual classifiers (GEC^{TLV}). The best individual classifier reached a GEC of 78.79% (still Classifier ID1: GEC^{**}). The second best ensemble accuracy reached 80.57% (HC method, Ensemble #2: GEC^{HC}) on this independent partition. While the three best ensembles that use our algorithm among ensembles built of Classifier ID1 and Classifier ID35 are presented in Table 6, detailed with parameter settings, where $GECDiff^{**} = GEC^{TLV} - GEC^{**}$ and $GECDiff^{HC} = GEC^{TLV} - GEC^{HC}$.

5 Conclusion

In out paper, by using the confidence triplet of "general accuracy", "self confidence", and "expertise on the particular field" at classifying data objects, we created a novel meta-classifier stage classifier ensemble algorithm. This fusion-selection method is able to outdo the best of the combined classifiers' accuracy by more than 5% and also overshadows the second best meta-classifier stage algorithm by more than 3%. Hence, we experimentally proved that our algorithm has the potential to significantly increase classification performance compared to individual classifiers and other voting techniques. Considering future work, there are still numerous aspects that can be improved. Since in this paper only one ensemble was subjected to parameter optimization, which was selected by local optimization, it is very probable that a global optimization can reveal even more potential. There is the 3 dimensional space of (1) which individual classifier pairs to include in the ensemble, (2) and (3) setting the two main parameters. Furthermore, considering not only classifier pairs as ensembles, the third parameter of our technique is also present, instead of just dividing the space into the two segments of selection and fusion. Hence (4) setting the selection-fusion threshold. And finally, our algorithm will be tested on other well-known datasets as well.

References

[1] The 14th acm sigkdd international conference on knowledge discovery and data mining. http://www.sigkdd.org/kdd2008/, 2008.

[2] Kdd cup 2008 on mining medical data. http://www.kddcup2008.com/, 2008.

[3] The original data set of kdd cup 2008 competition on mining medical data. http://adatbanyaszat.tmit.bme.hu/ fozocsaba/kdd2008/, 2008.

[4] Breiman, L. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[5] Breiman, L. Arcing classifiers. *Annals of Statistics*, 26(3):801–849, 1998.

[6] Breiman, L. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[7] Freund, Y. and Schapire, R. E. Experiments with a new boosting algorithm. In *Thirteenth International Conference on Machine Learning*, pages 148–156, San Francisco, 1996. Morgan Kaufmann.

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

[9] Ho, T.K. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20(8):832–844, Aug. 1998.

[10] Kuncheva, L. I. *Combining Pattern Classifiers. Methods and Algorithms.* John Wiley and Sons, 2004.

[11] Kuncheva, L. I. Diversity in multiple classifier systems (editorial). *Information Fusion*, 6(1):3–4, 2004.

[12] Kuncheva, L. I. and Rodríguez, J. J. Classifier ensembles with a random linear oracle. *IEEE Transactions on Knowledge and Data Engineering*, 19(4):500–508, 2007.

[13] Kuncheva, L. I. and Rodríguez, J. J. An experimental study on rotation forest ensembles. In *7th International Workshop on Multiple Classifier Systems, MCS 2007, volume 4472 of LNCS*, pages 459–468. Springer, 2007.

[14] Long, P.M. and Vega, V.B. Boosting and microarray data. *Machine Learning*, 52:31–44, 2003.

[15] Margineantu, D. D. and Dietterich, T. G. Pruning adaptive boosting. In *Proc. 14th Int'l Conf. Machine Learning*, pages 211–218, 1997.

[16] Melville, P., Shah, N., Mihalkova, L., and Mooney, R.J. Experiments with ensembles with missing and noisy data. In *Proc Fifth Int'l Workshop Multiple Classifier Systems*, pages 293–302, 2004.

[17] Oza, N. C. Boosting with averaged weight vectors. In *Proc Fourth Int'l Workshop Multiple Classifier Systems (MCS 2003)*, 2003.

[18] Piatetsky-Shapiro, Gregory. Kdnuggets poll: Which analysis types you plan to do more in 2008. http://www.kdnuggets.com/polls/, 2007.

[19] Rodríguez, J. J., Kuncheva, L. I., and Alonso, C. J. Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1619–1630, 2006.

[20] Webb, G. I. Multiboosting: A technique for combining boosting and wagging. *Machine Learning*, 40(2):159–196, 2000.

[21] Y.S. Huang, C.Y. Suen. A method of combining multiple experts for the recognition of unconstrained handwritten numerals. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(1):90–94, 1995.