Reliable Probabilistic Classification with Neural Networks

Harris Papadopoulos

Frederick Research Center, 7-9 Filokyprou St., Palouriotisa, Nicosia 1036, Cyprus

Computer Science and Engineering Department, Frederick University, 7 Y. Frederickou St., Palouriotisa, Nicosia 1036, Cyprus

Abstract

Venn Prediction (VP) is a new machine learning framework for producing well-calibrated probabilistic predictions. In particular it provides well-calibrated lower and upper bounds for the conditional probability of an example belonging to each possible class of the problem at hand. This paper proposes five VP methods based on Neural Networks (NNs), which is one of the most widely used machine learning techniques. The proposed methods are evaluated experimentally on four benchmark datasets and the obtained results demonstrate the empirical well-calibratedness of their outputs and their superiority over the outputs of the traditional NN classifier.

Keywords: Venn Prediction, Neural Networks, Probabilistic Classification, Multiprobability Prediction

1. Introduction

Machine learning techniques are becoming increasingly popular for solving all kinds of problems that cannot be solved with conventional tools. They have been applied to a great variety of problems and fields with very good results. However, most machine learning techniques do not provide any indication about the uncertainty of each of their predictions, which would have been very beneficial for most applications and especially for risk sensitive settings such as medical diagnosis [1]. An indication of the likelihood of each prediction being correct notifies the user of a system about how much he can rely on each prediction and enables him to take more informed decisions.

A solution to this problem was given by a recently developed machine learning theory called *Conformal Prediction* (CP) [2]. CP can be used for extending traditional machine learning algorithms and developing methods (called Conformal Predictors) whose predictions are guaranteed to satisfy a given level of confidence without assuming anything more than that the data are independently and identically distributed (i.i.d.). More specifically, CPs produce as their predictions a set containing all the possible classifications needed to satisfy the required confidence level. To date many different CPs have been developed, see e.g. [3–11], and have been applied successfully to a variety of important problems such as the early detection of ovarian cancer [12], the classification of leukaemia subtypes [13], the recognition of hypoxia electroencephalograms (EEGs) [14], the recognition of gestures [15], the prediction of plant promoters [16], the diagnosis of acute abdominal pain [17], the assessment of the risk of complications following a coronary drug eluting stent procedure [18], the assessment of stroke risk [19] and the estimation of effort for software projects [20]. The CP framework has also been extended to additional problem settings such as active learning [21] and change detection in data streams [22].

Email address: h.papadopoulos@frederick.ac.cy (Harris Papadopoulos)

To be published in Neurocomputing (doi:10.1016/j.neucom.2012.07.034)

This paper focuses on an extension of the original CP framework, called Venn Prediction (VP), which can be used for making *multiprobability predictions*. In particular multiprobability predictions are a set of probability distributions for the true classification of the new example. This set can be summarized by lower and upper bounds for the conditional probability of the new example belonging to each one of the possible classes[1]. The resulting bounds are guaranteed to contain well-calibrated probabilities (up to statistical fluctuations). Again, like with CPs, the only assumption made for obtaining this guaranty is that the data are i.i.d.

The VP framework has until now been combined with the k-nearest neighbours algorithm in [2] and [23] and with Support Vector Machines in [24]. A Venn Predictor based on Neural Networks (NNs) was first proposed in [25] for binary classification problems. This work is extended here by developing five Venn Predictors based on NNs for multiclass problems, for which NNs have more than one output neurons. The choice of NNs as basis for the proposed methods was made for two main reasons: (a) their popularity among machine learning techniques for almost any type of application, see e.g. [26–31], and (b) that they can produce probabilistic outputs which can be compared with those produced by the Venn Predictors. The experiments performed examine on one hand the empirical well-calibratedness of the probability bounds produced by the proposed methods and on the other hand compare them with the probabilistic outputs of the traditional NN classifier.

The rest of this paper starts with an overview of the Venn Prediction framework in the next section, while in Section 3 it details the proposed Neural Network Venn Prediction methods. Section 4 presents the experiments performed on four benchmark datasets and reports the obtained results. Finally, Section 5 gives the conclusions and future directions of this work.

2. The Venn Prediction Framework

This section gives a brief description of the Venn prediction framework; for more details the interested reader is referred to [2]. We are given a training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$ of examples[2], where each $x_i \in \mathbb{R}^d$ is the vector of attributes for example i and $y_i \in \{Y_1, \ldots, Y_c\}$ is the classification label of that example. We are also given a new unclassified example x_{l+1} and our task is to predict the probability of this new example belonging to each class $Y_j \in \{Y_1, \ldots, Y_c\}$ based only on the assumption that all $(x_i, y_i), i = 1, 2, \ldots$ are generated independently by the same probability distribution (i.i.d.).

The main idea behind Venn prediction is to divide all examples into a number of categories based on their similarity and calculate the probability of x_{l+1} belonging to each class $Y_j \in \{Y_1, \ldots, Y_c\}$ as the frequency of Y_j in the category that contains it. However, as we don't know the true class of x_{l+1}, we assign each one of the possible classification labels to it in turn and for each assigned classification label Y_k we calculate a probability distribution for the true class of x_{l+1} based on the examples

$$\{(x_1, y_1), \ldots, (x_l, y_l), (x_{l+1}, Y_k)\}. \tag{1}$$

To divide each set (1) into categories we use what we call a *Venn taxonomy*. A Venn taxonomy defines a number of categories and a rule based on which each example is assigned to one of these categories; formally a category is a multiset of examples. In effect similar examples should be assigned to the same category so that the resulting probability distribution for each assumed classification label Y_k will depend on the examples that are most similar to (x_{l+1}, Y_k).

Typically each taxonomy is based on a traditional machine learning algorithm, called the *underlying algorithm* of the Venn predictor. The output of this algorithm for each attribute vector $x_i, i = 1, \ldots, l+1$ after being trained either on the whole set (1), or on the set resulting after removing the pair (x_i, y_i) from (1), is used to assign (x_i, y_i) to one of a predefined set of categories. For example, a Venn taxonomy that

[1] It should be noted that moving from multiprobability predictions to the corresponding lower and upper bounds entails some loss of information, since the set of probability predictions for each class is replaced by their maximum and minimum values.

[2] The "training set" is in fact a multiset, as it can contain some examples more than once.

can be used with every traditional algorithm puts in the same category all examples that are assigned the same classification label by the underlying algorithm. At this point it is important to emphasize the difference between the classes of the problem and the categories of a Venn taxonomy. Even though this example taxonomy consists of a category corresponding to each classification label, these categories are assigned examples based on the output classification label of the underlying algorithm and not on the true class to which each example belongs. Therefore the category corresponding to a given classification label Y_k will contain the examples that the underlying algorithm "believes" to belong to class Y_k, which are not necessarily the same as the examples that actually do belong to that class since the underlying algorithm might be wrong in some cases. Of course other Venn taxonomies can be defined that depend on more information obtained from the underlying algorithm rather than just the output classification label. Four new Venn taxonomies for multiclass Neural Networks are defined in the next section.

After partitioning (1) into categories using a Venn taxonomy, the category T_{new} containing the new example (x_{l+1}, Y_k) will be nonempty as it will contain at least this one example. Then the empirical probability of each classification label Y_j in this category will be

$$p^{Y_k}(Y_j) = \frac{|\{(x^*, y^*) \in T_{new} : y^* = Y_j\}|}{|T_{new}|}. \tag{2}$$

This is a probability distribution for the label of x_{l+1}. After assigning all possible classification labels to x_{l+1} we get a set of probability distributions that compose the multiprobability prediction of the Venn predictor $P_{l+1} = \{p^{Y_k} : Y_k \in \{Y_1, \ldots, Y_c\}\}$. As proved in [2] the predictions produced by any Venn predictor are automatically well-calibrated multiprobability predictions. This is true regardless of the taxonomy of the Venn predictor. Of course the taxonomy used is still very important as it determines how efficient, or informative, the resulting predictions are. We want the diameter of multiprobability predictions and therefore their uncertainty to be small, since saying that the probability of a given classification label for an example is between 0.8 and 0.9 is much more informative than saying that it is between 0 and 0.9. We also want the predictions to be as close as possible to zero or one, indicating that a classification label is highly unlikely or highly likely respectively.

The maximum and minimum probabilities obtained for each classification label Y_j define the interval for the probability of the new example belonging to Y_j:

$$\left[\min_{k=1,\ldots,c} p^{Y_k}(Y_j), \max_{k=1,\ldots,c} p^{Y_k}(Y_j)\right]. \tag{3}$$

To simplify notation the lower bound of this interval for a given class Y_j will be denoted as $L(Y_j)$ and the upper bound will be denoted as $U(Y_j)$. The Venn predictor outputs the class $\hat{y} = Y_{j_{best}}$ as its prediction where

$$j_{best} = \arg\max_{j=1,\ldots,c} \overline{p(Y_j)}, \tag{4}$$

and $\overline{p(Y_j)}$ is the mean of the probabilities obtained for Y_j:

$$\overline{p(Y_j)} = \frac{1}{c}\sum_{k=1}^{c} p^{Y_k}(Y_j). \tag{5}$$

This prediction is accompanied by the interval

$$[L(\hat{y}), U(\hat{y})] \tag{6}$$

as the probability interval of it being correct. The complementary interval

$$[1 - U(\hat{y}), 1 - L(\hat{y})] \tag{7}$$

gives the probability that $\hat{y}$ is not the true classification label of the new example and it is called the *error probability interval*.

3. Venn Prediction with Neural Networks

This section describes the proposed Neural Network based Venn Prediction methods. The NNs used were fully connected feed-forward networks with a single hidden layer consisting of units with a hyperbolic tangent activation function. Their output layer consisted of units with a softmax activation function [32], which made all their outputs lie between zero and one and their sum equal to one. They were trained with the scaled conjugate gradient algorithm [33] minimizing cross-entropy error (log loss):

$$CE = -\sum_{i=1}^{N}\sum_{j=1}^{c} t_i^j \log(o_i^j), \tag{8}$$

where N is the number of examples, c is the number of possible classes, $o_i^1, \ldots, o_i^c$ are the outputs of the network for example i and $t_i^1, \ldots, t_i^c$ is the binary form of the true classification label y_i of example i, that is

$$t_i^j = \begin{cases} 1, & \text{if } y_i = Y_j, \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

As a result the outputs of these NNs can be interpreted as probabilities for each class and can be compared with those produced by the proposed methods.

As explained in Section 2 the difference between alternative Venn Prediction methods is the taxonomy they use to divide examples into categories. Here five different Venn taxonomies are defined which allocate examples into categories based on the outputs $o_i^1, \ldots, o_i^c$ of the NN for each example i after being trained on the extended set (1).

Algorithm 1: Neural Networks Venn Predictor

Input: training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$, new example x_{l+1}, possible classes $\{Y_1, \ldots, Y_c\}$.
for $k = 0$ **to** c **do**
  Train the NN on the extended set $\{(x_1, y_1), \ldots, (x_l, y_l), (x_{l+1}, Y_k)\}$;
  Supply the input patterns $x_1, \ldots, x_{l+1}$ to the trained NN to obtain the outputs $o_1, \ldots, o_{l+1}$;
  for $i = 0$ **to** $l + 1$ **do**
    Assign (x_i, y_i) to the corresponding category T_m of the Venn taxonomy V_1, V_2, V_3, V_4, or V_5 according to the NN outputs $o_i^1, \ldots, o_i^c$;
  end
  Find the category T_{new} that contains (x_{l+1}, Y_k);
  for $j = 0$ **to** c **do**
    $p^{Y_k}(Y_j) := \frac{|\{(x^*, y^*) \in T_{new} : y^* = Y_j\}|}{|T_{new}|}$;
  end
end
for $j = 0$ **to** c **do**
  $\overline{p(Y_j)} := \frac{1}{c}\sum_{k=1}^{c} p^{Y_k}(Y_j)$;
end
Output:
  Prediction $\hat{y} = \arg\max_{j=1,\ldots,c} \overline{p(Y_j)}$;
  The probability interval for $\hat{y}$: $[\min_{k=1,\ldots,c} p^k(\hat{y}), \max_{k=1,\ldots,c} p^k(\hat{y})]$.

The first Venn taxonomy, which will be denoted as V_1, is the simple taxonomy that assigns two examples to the same category if their maximum outputs correspond to the same class. This produces c categories, one for each possible class of the problem. This is a taxonomy that can be used with any traditional classifier as underlying algorithm of the Venn Predictor. The remaining four taxonomies defined below were developed in this work especially for being used with Neural Networks as the underlying algorithm of the Venn Predictor. These taxonomies take into account more information about the actual values of the NN

outputs, rather than just which one is the maximum output, and divide examples more effectively into more than c categories. Consequently the taxonomy categories are smaller than those of V_1 and consist of examples that are more similar to each other, resulting in more accurate probabilistic outputs.

The second taxonomy, which will be denoted as V_2, further divides the examples in each category of taxonomy V_1 into two smaller categories based on the value of their maximum output. It is expected that the higher the value of the maximum output for an example, the higher the chance of the corresponding class being the correct one. Therefore the examples of each category of taxonomy V_1 are divided into those with maximum output above a high threshold θ and those with maximum output below θ. This produces $2c$ categories. In principle θ can be set to any value between $\frac{1}{c}$ and 1, which is the range of possible values for the maximum output of the NN, however values in the upper half of this range are most appropriate since typically the maximum outputs of NN are relatively high. Here θ is set to 0.75 for all experiments with this taxonomy.

The third taxonomy, which will be denoted as V_3, again further divides each category of taxonomy V_1 into two smaller categories, but this time the division depends on the value of the second highest output of the examples. It is expected that the higher the value of the second highest output of an example, which corresponds to the most likely classification after the predicted one, the lower the chance of the class corresponding to its maximum output being the correct one. Therefore the examples of each category of taxonomy V_1 are divided into those with second highest output above a low threshold θ and those with second highest output below θ. This again produces $2c$ categories. In this case θ can be set to any value between 0 and 0.5, as the second highest output cannot be higher than 0.5. Here it is set to 0.25 for all experiments with this taxonomy.

The fourth taxonomy, which will be denoted as V_4, again further divides each category of taxonomy V_1 in two, but this time the division depends on the difference between the highest and second highest outputs. In effect this difference takes into account both how high the maximum output of the example is and how low the second highest output is. The bigger this difference is for an example, the higher the chance of the class corresponding to its maximum output being the correct one. Therefore the examples of each category of taxonomy V_1 are divided into those with difference between its two highest outputs above a threshold θ and those with difference below θ. Like V_2 and V_3 this taxonomy also consists of $2c$ categories. In this case θ can be set to any value between 0 and 1, however values in the middle of this range are most appropriate as very small or very big differences are very unusual. Here θ is set to 0.5 for all experiments with this taxonomy.

The fifth and last taxonomy, which will be denoted as V_5, assigns two examples to the same category if their outputs that are above a given low threshold θ correspond to the same set of classes. In effect this taxonomy considers all outputs above θ as likely and puts the examples that have the same likely classifications into the same category. In principle this taxonomy consists of 2^c categories, but most of them are almost always empty as having more than 2 outputs above θ is extremely unusual. In this case θ can be set to a value between 0 and 0.5, as a $\theta \geq 0.5$ would never have more than one outputs as likely. Here θ is set to 0.25 for all experiments with this taxonomy.

Using these taxonomies the examples are divided into categories for each assumed classification label $Y_k \in \{Y_1, \ldots, Y_c\}$ of x_{l+1} and the process described in Section 2 is followed for calculating the outputs of the Neural Network Venn Predictor (NN-VP). Algorithm 1 presents the complete NN-VP algorithm.

4. Experiments and Results

Experiments were performed on four datasets from the UCI Machine Learning Repository [34], which has been widely used as a source for benchmark datasets in testing machine learning algorithms:

- **Teaching Assistant Evaluation**, which is concerned with the evaluation of 151 Teaching Assistant (TA) assignments over three regular semesters and two summer semesters at the Statistics Department of the University of Wisconsin-Madison. The 151 TA assignments are described by 5 attributes and are divided into 3 score classes of roughly equal size: low, medium, high.

- **Glass Identification**, which is concerned with the identification of glass types based on their oxide content, motivated by criminological investigations. The dataset consists of 214 glasses of 6 different types (classes). The number of glasses of each type ranges from 76 to 9. Each glass is described by 9 attributes.
- **Ecoli**, which is concerned with the prediction of the localization sites of proteins. It consists of 336 instances described by 7 attributes and divided into 8 classes. The number of examples in each class ranges from 143 to just 2 and 91% of the examples belong to 4 out of the 8 classes.
- **Vehicle Silhouettes**, which was gathered at the Turing Institute, Glasgow, Scotland. It is concerned with the classification of vehicle silhouettes into 4 types of vehicles. There are 846 instances, each described by 18 attributes extracted from the corresponding silhouette.

In an effort to make the evaluation and comparison presented here as general as possible the four datasets used correspond to different problem domains. It is also worth to note that two of the datasets have a class imbalance problem, which results in the corresponding tasks being somewhat more difficult.

The NNs used were fully connected feed-forward networks with a single hidden layer consisting of hyperbolic tangent units and an output layer consisting of softmax units. The number of hidden units used for each dataset was experimentally selected by applying the traditional NN classifier on 10 random divisions of each dataset into training and test sets consisting of 90% and 10% of the examples respectively. It should be noted that these randomly generated sets are different from the ones used for evaluating the performance of the NN-VP in the batch setting described in Subsection 4.2. Despite this difference there still remains an element of data snooping in the selection of the number of hidden units for each dataset, but this is in fact in favour of the traditional NN classifier, which is what the performance of NN-VP is compared against, since this was the classifier used for determining the number of hidden units. The selected number of hidden units for each dataset is reported in Table 1 along with the dataset characteristics.

All NNs were trained with the scaled conjugate gradient algorithm [33] minimizing cross-entropy error (8) and early stopping based on a validation set consisting of 30% of the corresponding training set. In an effort to avoid local minima each NN was trained 3 times with different random initial weight values and the one that performed best on the validation set was selected for being applied to the test examples. Before each training session all attributes were normalised setting their mean value to 0 and their standard deviation to 1.

Two sets of experiments were performed: on-line experiments, presented in Subsection 4.1, to demonstrate empirically that the probability bounds produced by NN-VP are well-calibrated and batch experiments, presented in Subsection 4.2, to compare the performance of NN-VP with that of the traditional NN classifier and assess the performances of the five different Venn taxonomies.

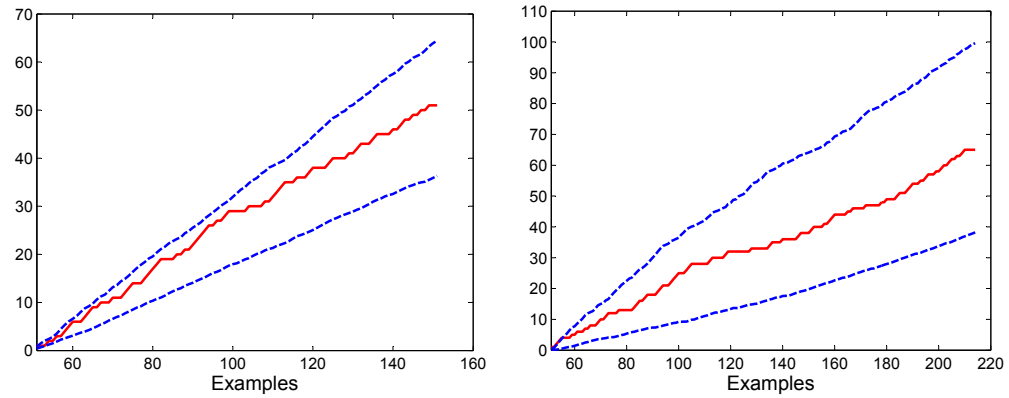
4.1. On-line Experiments

This subsection demonstrates the empirical well-calibratedness of NN-VP by applying it to the four datasets in the on-line mode. More specifically, starting with an initial training set consisting of 50 examples, each subsequent example is predicted in turn and then its true classification is revealed and it is added to the training set for predicting the next example.

Figure 1 shows the following three curves obtained by applying NN-VP with taxonomy V_1 on each dataset:

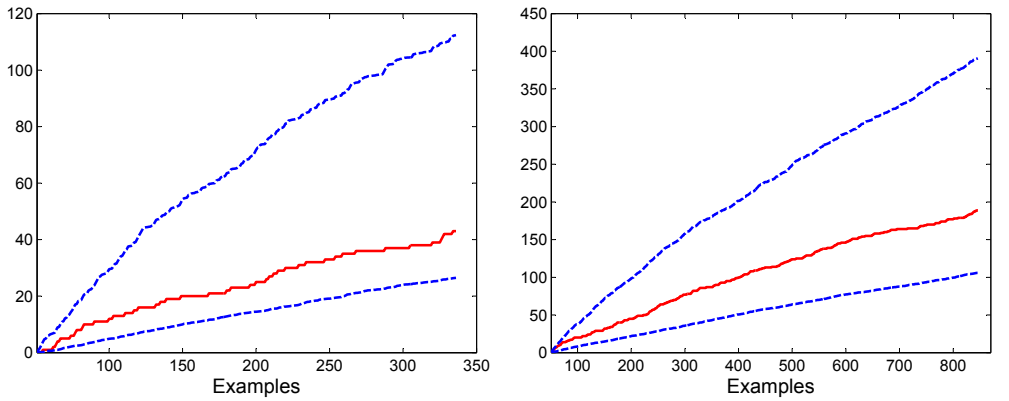
	TA Evaluation	Glass Identification	Ecoli	Vehicle Silhouettes
Examples	151	214	336	846
Attributes	5	9	7	18
Classes	3	6	8	4
Hidden Neurons	5	5	10	11

Table 1: Main characteristics and number of hidden neurons used for each data set.



(a) Teaching Assistant Evaluation

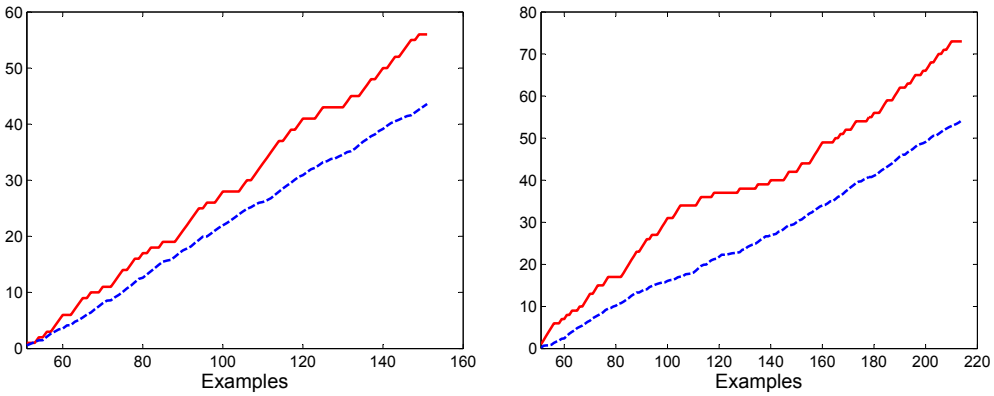
(b) Glass Identification



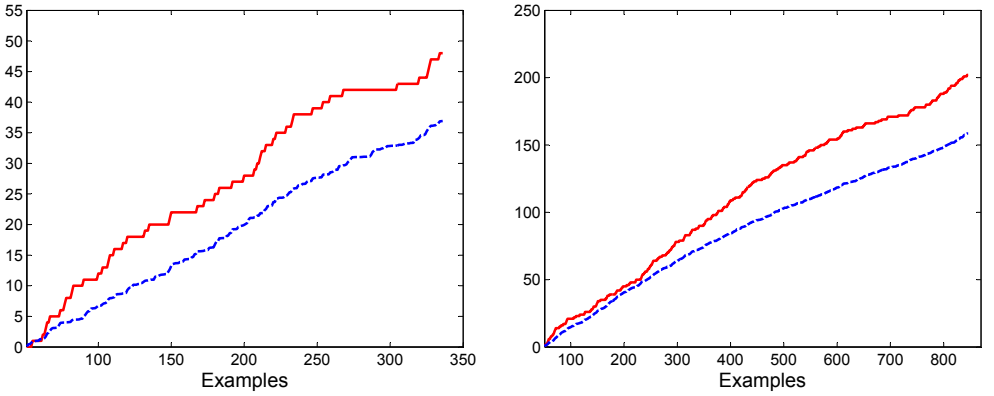
(c) Ecoli

(d) Vehicle Silhouettes

Figure 1: On-line performance of NN-VP with V_1 on the four datasets. Each plot shows the cumulative number of errors E_n with a solid line and the cumulative lower and upper error probability curves LEP_n and UEP_n with dashed lines.



(a) Teaching Assistant Evaluation (b) Glass Identification



(c) Ecoli (d) Vehicle Silhouettes

Figure 2: On-line performance of the traditional NN classifier on the four datasets. Each plot shows the cumulative number of errors E_n with a solid line and the cumulative error probability curve EP_n with a dashed line.

- the cumulative error curve

$$E_n = \sum_{i=1}^{n} err_i, \tag{10}$$

where $err_i = 1$ if the prediction $\hat{y}_i$ is wrong and $err_i = 0$ otherwise,

- the cumulative lower error probability curve (see (7))

$$LEP_n = \sum_{i=1}^{n} (1 - U(\hat{y}_i)) \tag{11}$$

- and the cumulative upper error probability curve

$$UEP_n = \sum_{i=1}^{n} (1 - L(\hat{y}_i)). \tag{12}$$

In effect the final cumulative errors are expected to lie between the final values of the cumulative upper and lower error probability curves up to statistical fluctuations. The four plots of Figure 1 confirm that the probability intervals produced by NN-VP are well-calibrated. In fact the cumulative errors are always included inside the cumulative upper and lower error probability curves produced by the NN-VP. The same is true for the plots obtained with the other four Venn taxonomies, which are very similar to the ones shown in this figure and are presented in Appendix A.

The analogous plots generated by applying the traditional NN classifier to the four datasets are shown in Figure 2. In this case the cumulative error curve (10) for each NN is plotted together with the cumulative error probability curve

$$EP_n = \sum_{i=1}^{n} |1 - \hat{p}_i|, \tag{13}$$

where $\hat{p}_i$ is the probability given by the NN for example i belonging to its predicted classification $\hat{y}_i$:

$$\hat{p}_i = \max_{j=1,\ldots,c} o_i^j. \tag{14}$$

In effect this curve is the sum of the probabilities of all classes except the predicted one for each example according to the NN. One would expect that this curve would be very near the cumulative error curve if the probabilities produced by the NN were well-calibrated. The plots of Figure 2 show that this is not the case. The NNs underestimate the true error probability in all cases since the cumulative error curve is much higher than the cumulative error probability curve. To check how misleading the probabilities produced by the NN are, the 2-sided p-value of obtaining a total number of errors E_N with the observed deviation from the expected errors EP_N given the probabilities produced by the NN was calculated for each dataset. The resulting p-values were 0.0087 for the TA Evaluation dataset, 0.00091 for the Glass Identification dataset, 0.026 for the Ecoli dataset and 0.000013 for the Vehicle Silhouettes dataset. The fact that three out of four of these p-values are below the 0.01 significance level demonstrates the need for probability intervals as opposed to single probability values as well as that the probabilities produced by NNs can be very misleading.

4.2. Batch Experiments

This subsection examines the performance of NN-VP in the batch setting and compares its results with those of the direct predictions made by the traditional NN classifier. For these experiments the four datasets were divided randomly into training and test sets consisting of 90% and 10% of their examples respectively. In order for the results not to depend on a particular division into training and test sets, 10 different random divisions were performed and all results reported here are over all 10 test sets.

Since NNs produce a single probabilistic output o^j for each possible classification Y_j, for NN-VP the values $\overline{p(Y_j)}$ corresponding to the estimate of NN-VP about the probability of each test example belonging

Table 2: Results of the traditional NN and the five NN-VPs on the TA Evaluation dataset

Method		Accuracy	CE	BS	REL
Traditional NN		45.33%	155.62	0.6323	0.0475
NN-VP	V_1	48.67%	156.27	0.6286	0.0283
	V_2	47.33%	155.67	0.6249	0.0294
	V_3	50.67%	152.98	0.6156	0.0335
	V_4	48.67%	154.88	0.6230	**0.0215**
	V_5	**52.67%**	**150.63**	**0.5997**	0.0274

to class Y_j were used for this performance evaluation and comparison. Consequently in this Subsection $\hat{o}_i^j$ will be used to denote the probabilistic output of the method in question for classification Y_j of example i, which in the case of the NN-VPs will correspond to $\overline{p_i(Y_j)}$. For reporting these results four quality metrics are used. The first is the accuracy of each classifier, which does not take into account the probabilistic outputs produced, but it is the most popular metric for assessing the quality of classifiers. The second is cross-entropy error (8), which is in fact the error minimized by the training algorithm of the NNs on the training set. The third metric is the Brier score [35]:

$$BS = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{c}(\hat{o}_i^j - t_i^j)^2, \tag{15}$$

where $t_i^1, \ldots, t_i^c$ is the binary form of the true classification label y_i (see (9)). The cross-entropy error, or log-loss, and the Brier score are the most popular quality metrics for probability assessments.

The Brier score can be decomposed into three terms interpreted as the uncertainty, reliability and resolution of the probabilities, by dividing the range of probability values into a number of intervals K and representing each interval $k = 1, \ldots, K$ by a 'typical' probability value r_k [36]. The reliability term of this decomposition measures how close the output probabilities are to the true probabilities and therefore reflects how well-calibrated the output probabilities are. This is the most important component of interest in this work as it evaluates how much one can rely on the probabilistic outputs produced by each method. Hence this is the fourth metric used here. It is defined in [36] as:

$$REL = \frac{1}{N}\sum_{k=1}^{K} n_k(r_k - \phi_k)^2, \tag{16}$$

where n_k is the number of $\hat{o}_i^j$, $i = 1, \ldots, N$ and $j = 1, \ldots, c$ in the interval k and ϕ_k is the percentage of these outputs for which $t_i^j = 1$, i.e. the example belongs to the corresponding class of the output. Here the number of categories K was set to 100 for the first three datasets and to 200 for the much larger Vehicle Silhouettes dataset.

Tables 2 to 5 present the results of the traditional NN and the five NN-VPs on each dataset. The values in bold indicate the best performance for each metric. The values reported in these tables show that all five VPs perform better than the traditional NN classifier against all metrics except the cross-entropy error. However, even in the case of the cross-entropy error metric for two out of the four datasets the best values are obtained by one of the VPs. Overall, the differences between the traditional NN classifier and the five VPs in the values of the cross-entropy error and Brier score metrics are relatively small ranging from an increase of 8.6% on the value of the traditional NN in the worst case to a decrease of 9.8% in the best case. The differences in terms of accuracy are a bit more important ranging from an improvement of 0.6% for the worst performing VP to an improvement of 16.2% for the best performing VP. More significant differences are observed in the values of the reliability metric where for the TA Evaluation, Glass Identification and Vehicle Silhouettes datasets the improvement of VPs ranges from 23.7% to 54.7%. This shows that even if we do not take into

Table 3: Results of the traditional NN and the five NN-VPs on the Glass Identification dataset

Method		Accuracy	CE	BS	REL
Traditional NN		57.62%	236.37	0.5697	0.0172
NN-VP	V_1	58.57%	217.71	0.5354	0.0119
	V_2	59.52%	216.00	0.5311	0.0124
	V_3	60.00%	217.14	0.5306	0.0110
	V_4	**60.48%**	**213.22**	**0.5245**	0.0109
	V_5	59.05%	216.35	0.5353	**0.0089**

Table 4: Results of the traditional NN and the five NN-VPs on the Ecoli dataset

Method		Accuracy	CE	BS	REL
Traditional NN		86.76%	**151.30**	0.2090	0.0061
NN-VP	V_1	88.53%	163.35	0.2023	**0.0059**
	V_2	88.53%	164.36	0.2040	0.0060
	V_3	89.12%	156.13	**0.1948**	0.0061
	V_4	88.53%	161.52	0.2011	0.0061
	V_5	**89.41%**	156.61	0.1980	**0.0059**

account the probabilistic intervals produced by the VPs and consider only the mean probabilities for each class, we can still achieve a considerable improvement over the reliability of the probabilities produced by the traditional NN classifier. In the case of the Ecoli dataset the reliability values of all methods are more or less the same probably because in this case the traditional NN classifier gives more reliable probabilities. Of course one cannot know if the same will be true for the particular problem and data he is working on. Moreover, even in such cases the probabilistic intervals of VPs are much more reliable since they are guaranteed to contain well-calibrated probabilities.

When comparing the performance of the five Venn taxonomies between each other, one can see that in most cases the new taxonomies defined here, V_2, V_3, V_4 and V_5, perform better than the simple taxonomy V_1. In terms of accuracy the best performance is obtained with V_4 for two out of the four datasets and with V_5 for the other two; however the differences between them are relatively small. It should also be noted that V_3 gives the second best accuracy for three out of the four datasets. Overall, if we average the performance of each metric over the four datasets, V_5 gives the best accuracy, V_3 gives the lowest cross-entropy error (which is lower than that of the traditional NN classifier), V_5 gives the lowest Brier score and V_4 gives the best reliability. This shows the advantage of using the four new Venn taxonomy definitions proposed in this work. The superior performance of the proposed definitions is due to the more effective partitioning of the examples into smaller categories and consequently the higher similarity between the examples in each category, which results in the probabilistic outputs of the VP being more accurate. As the performance of the proposed taxonomies varies across tasks, the most suitable taxonomy and the best threshold value θ for a particular task can be chosen by experimentation on the available training examples.

5. Conclusions

This paper presented five Venn Predictors based on Neural Networks. Unlike the traditional NN classifiers the proposed methods produce probability intervals for each of their predictions, which are well-calibrated under the general i.i.d. assumption. The experiments performed in the on-line setting demonstrated the well-calibratedness of the probability intervals produced by the NN-VPs and their superiority over the single probabilities produced by traditional NNs, which can be significantly different from the observed frequencies.

Table 5: Results of the traditional NN and the five NN-VPs on the Vehicle Silhouettes dataset

Method		Accuracy	CE	BS	REL
Traditional NN		80.35%	**356.56**	0.2620	0.0139
NN-VP	V_1	80.82%	373.85	0.2567	0.0106
	V_2	81.88%	371.81	0.2517	**0.0086**
	V_3	81.65%	369.92	0.2516	0.0099
	V_4	**82.12%**	368.48	**0.2494**	0.0088
	V_5	81.18%	380.11	0.2585	0.0106

Moreover, the comparison performed in the batch setting showed that even when one discards the interval information produced by the NN-VPs by taking the mean of their multiprobability predictions, these are still much more reliable in most cases than the probabilities produced by traditional NNs. Lastly, the batch setting experiments also showed that NN-VPs are more accurate.

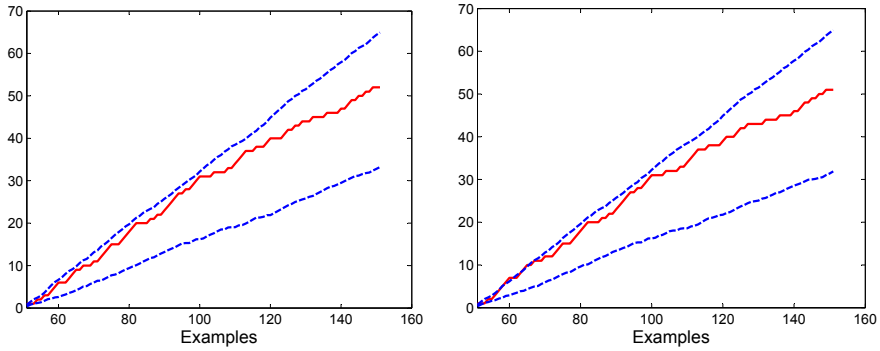
An important drawback of the VP approach is its computational inefficiency, which is a result of its transductive nature. Consequently an immediate future direction of this work is the development of an inductive VP approach based on the Inductive Conformal Prediction idea [5] in order to overcome this computational inefficiency problem when dealing with large datasets. In addition, the application of VP to the problem of osteoporosis risk assessment is currently being studied. Its application to other challenging real world problems and the evaluation of its results is also of great interest.

Acknowledgments.

The author is grateful to Professors V. Vovk and A. Gammerman for useful discussions. This work was supported by the European Regional Development Fund and the Cyprus Government through the Cyprus Research Promotion Foundation "DESMI 2009-2010" research contract TPE/ORIZO/0609(BIE)/24 ("Development of New Venn Prediction Methods for Osteoporosis Risk Assessment").

Appendix A. On-line Experiments with V_2, V_3, V_4 and V_5

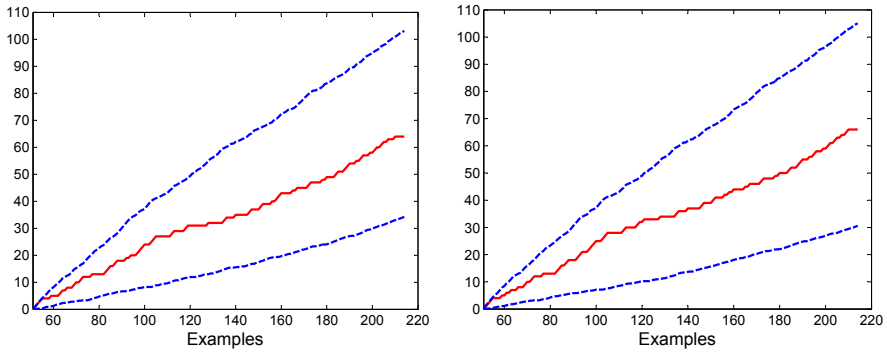
This Appendix presents in Figures A.3 to A.6 the plots produced when applying the NN-VP with taxonomies V_2, V_3, V_4 and V_5 in the on-line mode to the four datasets. Like in the plots obtained with taxonomy V_1, the cumulative errors are always included inside the cumulative upper and lower error probability curves produced by the NN-VP, which demonstrates that the probability intervals produced by NN-VP are well-calibrated.



(a) V_2 (b) V_3

(c) V_4 (d) V_5

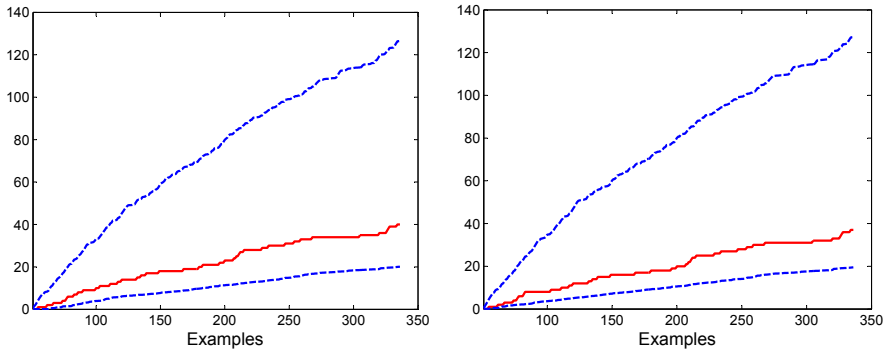
Figure A.3: On-line performance of NN-VP with V_2, V_3, V_4 and V_5 on the TA Evaluation dataset. Each plot shows the cumulative number of errors E_n with a solid line and the cumulative lower and upper error probability curves LEP_n and UEP_n with dashed lines.



(a) V_2 (b) V_3

(c) V_4 (d) V_5

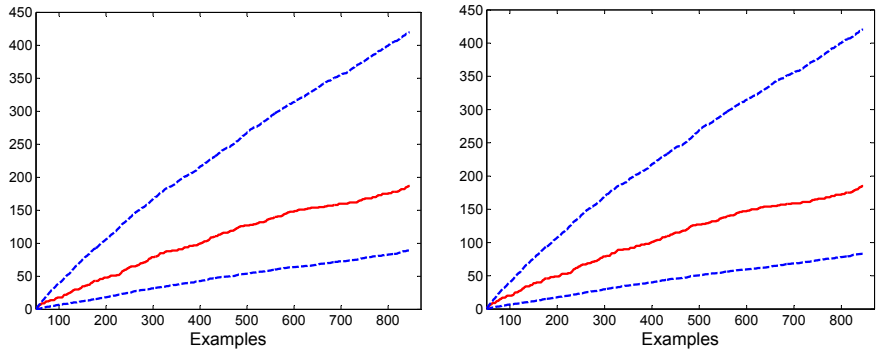
Figure A.4: On-line performance of NN-VP with V_2, V_3, V_4 and V_5 on the Glass Identification dataset. Each plot shows the cumulative number of errors E_n with a solid line and the cumulative lower and upper error probability curves LEP_n and UEP_n with dashed lines.



(a) V_2 (b) V_3

(c) V_4 (d) V_5

Figure A.5: On-line performance of NN-VP with V_2, V_3, V_4 and V_5 on the Ecoli dataset. Each plot shows the cumulative number of errors E_n with a solid line and the cumulative lower and upper error probability curves LEP_n and UEP_n with dashed lines.



(a) V_2 (b) V_3

(c) V_4 (d) V_5

Figure A.6: On-line performance of NN-VP with V_2, V_3, V_4 and V_5 on the Vehicle Silhouettes dataset. Each plot shows the cumulative number of errors E_n with a solid line and the cumulative lower and upper error probability curves LEP_n and UEP_n with dashed lines.

References

[1] H. Holst, M. Ohlsson, C. Peterson, L. Edenbrandt, Intelligent computer reporting 'lack of experience': a confidence measure for decision support systems, Clinical Physiology 18 (1998) 139–147.

[2] V. Vovk, A. Gammerman, G. Shafer, Algorithmic Learning in a Random World, Springer, New York, 2005.

[3] A. Lambrou, H. Papadopoulos, A. Gammerman, Reliable confidence measures for medical diagnosis with evolutionary algorithms, IEEE Transactions on Information Technology in Biomedicine 15 (2011) 93–99.

[4] I. Nouretdinov, T. Melluish, V. Vovk, Ridge regression confidence machine, in: Proceedings of the 18th International Conference on Machine Learning (ICML'01), Morgan Kaufmann, San Francisco, CA, 2001, pp. 385–392.

[5] H. Papadopoulos, Inductive Conformal Prediction: Theory and application to neural networks, in: P. Fritzsche (Ed.), Tools in Artificial Intelligence, InTech, Vienna, Austria, 2008, pp. 315–330. URL http://www.intechopen.com/download/pdf/pdfs_id/5294.

[6] H. Papadopoulos, H. Haralambous, Reliable prediction intervals with regression neural networks, Neural Networks 24 (2011) 842–851.

[7] H. Papadopoulos, K. Proedrou, V. Vovk, A. Gammerman, Inductive confidence machines for regression, in: Proceedings of the 13th European Conference on Machine Learning (ECML'02), volume 2430 of *LNCS*, Springer, 2002, pp. 345–356.

[8] H. Papadopoulos, V. Vovk, A. Gammerman, Regression conformal prediction with nearest neighbours, Journal of Artificial Intelligence Research 40 (2011) 815–840. URL http://dx.doi.org/10.1613/jair.3198.

[9] K. Proedrou, I. Nouretdinov, V. Vovk, A. Gammerman, Transductive confidence machines for pattern recognition, in: Proceedings of the 13th European Conference on Machine Learning (ECML'02), volume 2430 of *LNCS*, Springer, 2002, pp. 381–390.

[10] C. Saunders, A. Gammerman, V. Vovk, Transduction with confidence and credibility, in: Proceedings of the 16th International Joint Conference on Artificial Intelligence, volume 2, Morgan Kaufmann, Los Altos, CA, 1999, pp. 722–726.

[11] S. Bhattacharyya, Confidence in predictions from random tree ensembles, in: Proceedings of the 11th IEEE International Conference on Data Mining (ICDM 2011), Springer, 2011, pp. 71–80.

[12] A. Gammerman, V. Vovk, B. Burford, I. Nouretdinov, Z. Luo, A. Chervonenkis, M. Waterfield, R. Cramer, P. Tempst, J. Villanueva, M. Kabir, S. Camuzeaux, J. Timms, U. Menon, I. Jacobs, Serum proteomic abnormality predating screen detection of ovarian cancer, The Computer Journal 52 (2009) 326–333.

[13] T. Bellotti, Z. Luo, A. Gammerman, F. W. V. Delft, V. Saha, Qualified predictions for microarray and proteomics pattern diagnostics with confidence machines, International Journal of Neural Systems 15 (2005) 247–258.

[14] J. Zhang, G. Li, M. Hu, J. Li, Z. Luo, Recognition of hypoxia EEG with a preset confidence level based on EEG analysis, in: Proceedings of the International Joint Conference on Neural Networks (IJCNN 2008), part of the IEEE World Congress on Computational Intelligence (WCCI 2008), IEEE, 2008, pp. 3005–3008.

[15] I. Sprinkhuizen-Kuyper, L. Vuurpijl, Y. van Pinxteren, Reliable gesture recognition with transductive confidence machines, in: H. Dai, J. N. Liu, E. Smirnov (Eds.), Reliable Knowledge Discovery, Springer, 2012, pp. 183–200.

[16] I. A. Shahmuradov, V. V. Solovyev, A. J. Gammerman, Plant promoter prediction with confidence estimation, Nucleic Acids Research 33 (2005) 1069–1076.

[17] H. Papadopoulos, A. Gammerman, V. Vovk, Reliable diagnosis of acute abdominal pain with conformal prediction, Engineering Intelligent Systems 17 (2009) 115–126.

[18] V. N. Balasubramanian, R. Gouripeddi, S. Panchanathan, J. Vermillion, A. Bhaskaran, R. M. Siegel, Support vector machine based conformal predictors for risk of complications following a coronary drug eluting stent procedure, in: Proceedings of the IEEE Conference on Computers in Cardiology 2009, pp. 5–8.

[19] A. Lambrou, H. Papadopoulos, E. Kyriacou, C. S. Pattichis, M. S. Pattichis, A. Gammerman, A. Nicolaides, Assessment of stroke risk based on morphological ultrasound image analysis with conformal prediction, in: Proceedings of the 6th IFIP International Conference on Artificial Intelligence Appications and Innovations (AIAI 2010), volume 339 of *IFIP AICT*, Springer, 2010, pp. 146–153.

[20] H. Papadopoulos, E. Papatheocharous, A. S. Andreou, Reliable confidence intervals for software effort estimation, in: Proceedings of the 2nd Workshop on Artificial Intelligence Techniques in Software Engineering (AISEW 2009), volume 475 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2009. URL http://ceur-ws.org/Vol-475/AISEW2009/22-pp-211-220-208.pdf.

[21] S.-S. Ho, H. Wechsler, Transductive confidence machine for active learning, in: Proceedings of the International Joint Conference on Neural Networks 2003, volume 2, pp. 1435–1440.

[22] S.-S. Ho, H. Wechsler, A martingale framework for detecting changes in data streams by testing exchangeability, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (2010) 2113–2127.

[23] M. Dashevskiy, Z. Luo, Reliable probabilistic classification and its application to internet traffic, in: Proceedings of the 4th international conference on Intelligent Computing (ICIC 2008): Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues, volume 5226 of *LNCS*, Springer, 2008, pp. 380–388.

[24] C. Zhou, I. Nouretdinov, Z. Luo, D. Adamskiy, L. Randell, N. Coldham, A. Gammerman, A comparison of venn machine with platts method in probabilistic outputs, in: Proceedings of the 7th IFIP International Conference on Artificial Intelligence Appications and Innovations (AIAI 2011), volume 364 of *IFIP AICT*, Springer, 2011, pp. 483–490.

[25] H. Papadopoulos, Reliable probabilistic prediction for medical decision support, in: Proceedings of the 7th IFIP International Conference on Artificial Intelligence Appications and Innovations (AIAI 2011), volume 364 of *IFIP AICT*, Springer, 2011, pp. 265–274.

[26] G. C. Anastassopoulos, L. S. Iliadis, Ann for prognosis of abdominal pain in childhood: Use of fuzzy modelling for

convergence estimation, in: Proceedings of the 1st International Workshop on Combinations of Intelligent Methods and Applications, pp. 1–5.
[27] D. Mantzaris, G. Anastassopoulos, L. Iliadis, K. Kazakos, H. Papadopoulos, A soft computing approach for osteoporosis risk factor estimation, in: Proceedings of the 6th IFIP International Conference on Artificial Intelligence Appications and Innovations (AIAI 2010), volume 339 of *IFIP AICT*, Springer, 2010, pp. 120–127.
[28] C. S. Pattichis, C. Christodoulou, E. Kyriacou, M. S. Pattichis, Artificial neural networks in medical imaging systems, in: Proceedings of the 1st MEDINF International Conference on Medical Informatics and Engineering, pp. 83–91.
[29] L. S. Iliadis, F. Maris, An artificial neural network model for mountainous water-resources management: The case of cyprus mountainous watersheds, Environmental Modelling and Software 22 (2007) 1066–1072.
[30] H. Haralambous, H. Papadopoulos, 24-hour neural network congestion models for high-frequency broadcast users, IEEE Transactions on Broadcasting 55 (2009) 145–154.
[31] L. S. Iliadis, S. Spartalis, S. Tachos, Application of fuzzy t-norms towards a new artificial neural networks' evaluation framework: A case from wood industry, Information Sciences 178 (2008) 3828–3839.
[32] J. S. Bridle, Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition, in: F. F. Soulie, J. Herault (Eds.), Neuralcomputing: Algorithms, Architectures and Applications, Springer, Berlin, 1990, pp. 227–236.
[33] M. F. Moller, A scaled conjugate gradient algorithm for fast supervised learning, Neural Networks 6 (1993) 525–533.
[34] A. Frank, A. Asuncion, UCI machine learning repository, 2010.
[35] G. W. Brier, Verification of forecasts expressed in terms of probability, Monthly Weather Review 78 (1950) 1–3.
[36] A. H. Murphy, A new vector partition of the probability score, Journal of Applied Meteorology 12 (1973) 595–600.