Off-Line Learning with Transductive Confidence Machines: an Empirical Evaluation

Stijn Vanderlooy[1], Laurens van der Maaten[1], and Ida Sprinkhuizen-Kuyper[2]

[1] MICC-IKAT, Universiteit Maastricht, P.O. Box 616,
6200 MD Maastricht, The Netherlands,
{s.vanderlooy,l.vandermaaten}@micc.unimaas.nl
[2] NICI, Radboud University Nijmegen, P.O. Box 9104,
6500 HE Nijmegen, The Netherlands
i.kuyper@nici.ru.nl

Abstract. The recently introduced transductive confidence machines (TCMs) framework allows to extend classifiers such that they satisfy the calibration property. This means that the error rate can be set by the user prior to classification. An analytical proof of the calibration property was given for TCMs applied in the on-line learning setting. However, the nature of this learning setting restricts the applicability of TCMs. In this paper we provide strong empirical evidence that the calibration property also holds in the off-line learning setting. Our results extend the range of applications in which TCMs can be applied. We may conclude that TCMs are appropriate in virtually any application domain.

1 Introduction

Machine-learning classifiers are common in many real-life applications. Many of these applications are characterized by high error costs, indicating that incorrect classifications can have serious consequences. It is therefore desired to have classifiers that output reliable classifications. One way to achieve this is to complement each classification with a confidence value. Classifications with a low confidence value are not reliable and should be handled with caution.

For some classifiers (such as the naive Bayes classifier) a measure of confidence is readily available, but for many other classifiers this is not the case. The recently introduced transductive confidence machines (TCMs) framework allows for an efficient way to provide confidence values produced by virtually any classifier [8,17]. The essential property of TCMs is that their error rate is controlled by the user prior to classification. For example, if the user specifies an error rate of 0.05, then at most 5% of the classifications made by a TCM are incorrect. This property is called the calibration property and has been proven to hold in the on-line learning setting. However, this learning setting restricts the applicability of TCMs. In the paper we investigate to what extent the calibration property holds in the off-line learning setting. We investigate this by means of a systematic empirical evaluation of TCMs using six different classifiers on various real-world datasets.

The remainder of the paper is organized as follows. Section 2 defines the learning setting that we consider. Section 3 explains TCMs and the calibration property. It also provides implementations of six classifiers in the TCM framework. Section 4 investigates to what extent the calibration property holds in the off-line learning setting. Section 5 provides a final discussion on TCMs. Section 6 concludes that TCMs satisfy the calibration property in the off-line learning setting.

2 Learning Setting

We consider the supervised machine-learning setting. The instance space is denoted by $\mathcal{X}$ and corresponding label space by $\mathcal{Y}$. An example is of the form $z = (x, y)$ with $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. The symbol $\mathcal{Z}$ will be used as a compact notation for $\mathcal{X} \times \mathcal{Y}$. Training data are considered as a sequence of examples:

$$S = (x_1, y_1), \ldots, (x_n, y_n) = z_1, \ldots, z_n \ , \tag{1}$$

where each example is generated by the same unknown probability distribution P over $\mathcal{Z}$. We assume that this distribution satisfies the *exchangeability assumption*. This assumption states that the joint probability of a sequence of random variables is invariant under any permutation of the indices of these variables. In other words, the information that the z_i's provide is independent of the order in which they are collected. Formally, we write:

$$P(z_1, \ldots, z_n) = P(z_{\pi(1)}, \ldots, z_{\pi(n)}) \ , \tag{2}$$

for all permutations π on the set $\{1 \ldots, n\}$.[3]

We apply a classifier in the *off-line learning setting* (batch setting): the classifier is learned on training data and subsequently used to classify instances one-by-one. The true labels of instances are not returned. This is in contrast to the *on-line learning setting* where the true label of each instance is provided after prediction. The classifier is then retrained after each prediction since new information is available. Clearly, the on-line learning setting restricts the applicability of classifiers since any form of feedback can be very expensive.

3 Transductive Confidence Machines

Traditionally, classifiers assign a single label to an instance. In contrast, transductive confidence machines (TCMs) are allowed to assign a set of labels to each instance. Such a *prediction set* contains multiple labels if there is uncertainty in the true label of the instance [7,8,17]. Subsection 3.1 explains the construction of prediction sets. Subsection 3.2 discusses the calibration property. Subsection 3.3 outlines six practical implementations of TCMs.

[3] Note that exchangeable random variables are identically distributed and not necessarily independent from each other. Therefore, identically and independently distributed (iid) random variables are also exchangeable. The exchangeability assumption is thus weaker (i.e., more general) than the iid assumption.

3.1 Construction of Prediction Sets

To construct a prediction set for an unlabeled instance x_{n+1}, TCMs operate in a transductive manner. Each possible label $y \in \mathcal{Y}$ is tried as a label for instance x_{n+1}. In each try we form the example $z_{n+1} = (x_{n+1}, y)$ and add it to S. Then we measure how likely it is that the resulting sequence is generated by the underlying distribution P. To this end, each example in the *extended sequence*:

$$(x_1, y_1), \ldots, (x_n, y_n), (x_{n+1}, y) = z_1, \ldots, z_{n+1} \ , \tag{3}$$

is assigned a nonconformity score by means of a nonconformity measure. This measure defines how nonconforming an example is with respect to other available examples. We require that it is irrelevant in which order the nonconformity scores of the examples are calculated (due to the exchangeability assumption).

Definition 1. *A nonconformity measure is a measurable mapping:*

$$A : \mathcal{Z}^{(*)} \times \mathcal{Z} \to \mathbb{R} \cup \{\infty\} \ , \tag{4}$$

with output indicating how nonconforming an example is with respect to all other examples. The symbol $\mathcal{Z}^{(*)}$ denotes the set of all bags of elements of $\mathcal{Z}$. A bag is denoted by $\wr \cdot \int$.

Definition 2. *Given a sequence of examples $z_1, \ldots, z_{n+1}$ with $n \geq 1$, the nonconformity score of example z_i $(i = 1, \ldots, n)$ is defined as:*

$$\alpha_i = A(\wr z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_{n+1} \int, z_i) \ , \tag{5}$$

and the nonconformity score of example z_{n+1} is defined as:

$$\alpha_{n+1} = A(\wr z_1, \ldots, z_n \int, z_{n+1}) \ . \tag{6}$$

Nonconformity scores can be scaled arbitrarily by multiplying with a fixed non-zero number. Therefore, to know how nonconforming the created example z_{n+1} is in the extended sequence, the nonconformity score α_{n+1} is compared to all other α_i $(i = 1, \ldots, n)$.

Definition 3. *Given a sequence of nonconformity scores $\alpha_1, \ldots, \alpha_{n+1}$ with $n \geq 1$, the p-value of label y assigned to unlabeled instance x_{n+1} is defined as:*

$$p_y = \frac{|\{i = 1, \ldots, n+1 : \alpha_i \geq \alpha_{n+1}\}|}{n+1} \ . \tag{7}$$

If the p-value is close to its lower bound $1/(n+1)$, then example z_{n+1} is very nonconforming. The closer the p-value is to its upper bound 1, the more conforming example z_{n+1} is. Hence, the p-value indicates how likely it is that the tried label for an unlabeled instance is in fact the true label. A TCM outputs the set of labels with p-values above a predefined significance level ϵ.

Definition 4. *A transductive confidence machine determined by some nonconformity measure is a function that maps each sequence of examples $z_1, \ldots, z_n$ with $n \geq 1$, unlabeled instance x_{n+1}, and significance level $\epsilon \in [0,1]$ to the prediction set:*

$$\Gamma^{\epsilon}(z_1, \ldots, z_n, x_{n+1}) = \{y \in \mathcal{Y} \mid p_y > \epsilon\} \quad . \tag{8}$$

There may be situations in which many training examples have nonconformity score equal to the score of example z_{n+1}. The p-value is then large, but caution is needed since many examples are equally nonconforming, making it impossible to discriminate between them. To alleviate this problem, a randomized version of the p-value has been proposed [17, p. 27].

Definition 5. *Given a sequence of nonconformity scores $\alpha_1, \ldots, \alpha_{n+1}$ with $n \geq 1$, the randomized p-value of label y assigned to unlabeled instance x_{n+1} is defined as:*

$$p_y^{\tau} = \frac{|\{i = 1, \ldots, n+1 : \alpha_i > \alpha_{n+1}\}| + \tau |\{i = 1, \ldots, n+1 : \alpha_i = \alpha_{n+1}\}|}{n+1} \quad , \tag{9}$$

with τ a random number uniformly sampled from $[0,1]$ for instance x_{n+1}.

Definition 6. *A randomized transductive confidence machine determined by some nonconformity measure is a function that maps each sequence of examples $z_1, \ldots, z_n$ with $n \geq 1$, unlabeled instance x_{n+1}, uniformly distributed random number $\tau \in [0,1]$, and significance level $\epsilon \in [0,1]$ to the prediction set:*

$$\Gamma^{\epsilon,\tau}(z_1, \ldots, z_n, x_{n+1}) = \left\{y \in \mathcal{Y} \mid p_y^{\tau} > \epsilon\right\} \quad . \tag{10}$$

A randomized TCM treats the borderline cases $\alpha_i = \alpha_{n+1}$ more carefully. Instead of increasing the p-value with $1/(n+1)$, the p-value is increased with a random amount between 0 and $1/(n+1)$. In the following, we employ randomized TCMs, although for brevity we simply call them TCMs.

3.2 Calibration Property

In the on-line learning setting, TCMs have been proven to satisfy the *calibration property* [17, p. 20-22 & p. 193]. This property states that the long run error rate of a TCM with significance level ϵ equals ϵ:

$$\limsup_{n \to \infty} \frac{Err_n^{\epsilon}}{n} = \epsilon \quad , \tag{11}$$

with Err_n^{ϵ} the number of prediction sets that do not contain the true label, given the first n prediction sets.[4] The idea of the proof is to show that the sequence of prediction outcomes (i.e., whether the prediction set contains the true label or not) is a sequence of independent Bernoulli random variables with parameter ϵ.

[4] In case of non-randomized TCMs, the equality sign in (11) is replaced by the $\leq$ sign.

From (11) follows that the significance level has a frequentist interpretation as the limiting frequency of errors. It allows to control the number of errors prior to classification. The calibration property holds regardless of which nonconformity measure is used.

In the off-line learning setting there theoretically exists a small probability that TCMs are not well-calibrated (the training data is kept fixed, and therefore the prediction outcomes are not independent) [17, p. 111]. Section 4 investigates empirically whether TCMs are well-calibrated in the off-line learning setting.

3.3 Implementations

This subsection shows that virtually any classifier can be plugged into the TCM framework. Nonconformity measures are formulated for the following six classifiers: (1) k-nearest neighbour, (2) nearest centroid, (3) linear discriminant, (4) naive Bayes, (5) kernel perceptron, and (6) support vector machine. Although the nonconformity measures are based on specific classifier characteristics, they can readily be applied to similar classifiers. In addition, they provide clear insights in how to define new nonconformity measures.

The implementation of TCMs based on linear discriminant, kernel perceptron, and support vector machine considers binary classification tasks. This is due to the nature of these classifiers. We denote the binary label space as $\mathcal{Y} = \{-1, +1\}$. Extensions to multilabel learning are well-known and therefore not discussed in the paper. We implemented TCMs that can incrementally learn and decrementally unlearn a single instance, hereby keeping time complexity low. Pseudo codes of these efficient implementations are found in a technical report [16].

***k*-Nearest Neighbour** The k-nearest neighbour classifier (k-NN) classifies an instance by means of majority vote among the labels of the k nearest neighbours ($k \geq 1$) [4]. An example is nonconforming when it is far from nearest neighbours with identical label and close to nearest neighbours with different label.

A nonconformity measure can model this as follows. Given example $z_i = (x_i, y_i)$, define an ascending ordered sequence $D_i^{y_i}$ with distances from instance x_i to its k nearest neighbours with label y_i. Similarly, let $D_i^{-y_i}$ contain ordered distances from instance x_i to its k nearest neighbours with label different from y_i. The nonconformity score is then defined as:

$$\alpha_i = \frac{\sum_{j=1}^{k} D_{ij}^{y_i}}{\sum_{j=1}^{k} D_{ij}^{-y_i}} \ , \tag{12}$$

with subscript j representing the j-th element in a sequence [12]. Clearly, the nonconformity score is monotonically increasing when distances to the k nearest neighbours with identical label increase and/or distances to the k nearest neighbours with different label decrease.

Nearest Centroid The nearest centroid classifier (NC) learns a Voronoi partition on the training data. It assumes that examples cluster around a class centroid. An example is nonconforming when it is far from the class centroid of its label and close to the class centroids of other labels. Therefore, the nonconformity score of example $z_i = (x_i, y_i)$ can be defined as the distance from x_i to the class centroid of y_i relative to the minimum distance from x_i to all other class centroids [2]. Formally, we write:

$$\alpha_i = \frac{d(\mu_{y_i}, x_i)}{\min_{y \neq y_i} d(\mu_y, x_i)} \ , \tag{13}$$

with μ_y the class centroid of label y which is defined as:

$$\mu_y = \frac{1}{|C_y|} \sum_{i \in C_y} x_i \ , \tag{14}$$

with C_y the set of indices of instances with label y.

Linear Discriminant The linear discriminant classifier (LDC) learns a separating hyperplane by maximizing the between scatter of instances with different labels while minimizing the within scatter of instances with identical labels [6]. Instances close to the hyperplane are classified with low confidence since a small change in the hyperplane can result in a different classification of nearby instances. Therefore, a natural nonconformity score of example $z_i = (x_i, y_i)$ is the signed perpendicular distance from x_i to the hyperplane:

$$\alpha_i = -y_i \left(\langle w, x_i \rangle + b \right) \ , \tag{15}$$

with w and b the normal vector and intercept of the hyperplane, and $\langle \cdot, \cdot \rangle$ the inner product. If a classification is correct, then the nonconformity score is negative. A larger distance to the hyperplane represents more confidence in a correct classification, and consequently a lower nonconformity score is obtained. If a classification is incorrect, then the nonconformity score is positive and monotonically increasing with larger distances to the hyperplane.

Naive Bayes The naive Bayes classifier (NB) is a probabilistic classifier that applies Bayes theorem with independence assumptions [5]. A valid nonconformity score is large if the label of an instance is strange under the Bayesian model [17, p. 102]. We use the following as nonconformity score of example $z_i = (x_i, y_i)$:

$$\alpha_i = 1 - \mathbb{P}(y_i) \ , \tag{16}$$

with $\mathbb{P}(y_i)$ the conditional probability of label y_i that is estimated from the training data and instance x_i, i.e., $\mathbb{P}(\cdot)$ is the posterior label distribution computed by the naive Bayes classifier.[5]

[5] It is tempting to believe that the probabilities $\mathbb{P}(\cdot)$ are confidence values. However, it has been verified that these probabilities are overestimated in case of an incorrect prior, e.g., classifying with a probability of 0.7 does not mean that the true label is predicted 70% of the time [10].

Kernel Perceptron The kernel perceptron (KP) learns a separating hyperplane by updating a weight vector in a high-dimensional space during training [9]. The weight vector represents the normal vector and intercept of the hyperplane. The expansion of the weight vector in dual form is:

$$w = \sum_{i=1}^{n+1} \lambda_i y_i \Phi(x_i) \ , \tag{17}$$

with λ_i the dual variable for instance x_i and Φ the mapping to the high-dimensional space. It is easily verified that λ_i encodes the number of times that instance x_i is incorrectly classified during training [15, p. 241-242]. Hence, the nonconformity score of example $z_i = (x_i, y_i)$ can be defined as $\alpha_i = \lambda_i$ [10]. However, such a nonconformity score is not valid in the sense that the KP solution depends on the ordering of the training examples. Different KP runs result in different nonconformity scores. In our experiments we show that this violation of the exchangeability assumption does not have any effect in practice.

Support Vector Machine The support vector machine (SVM) finds a separating hyperplane with maximum margin using pairwise inner products of instances mapped to a high-dimensional space. The inner products are efficiently computed using a kernel function. The maximum margin hyperplane is found by solving a quadratic programming problem in dual form [15, Ch. 7].

In this optimization problem, the Lagrange multipliers $\lambda_1, \lambda_2, \ldots, \lambda_{n+1}$ associated with examples $z_1, \ldots, z_{n+1}$ take values in the domain $[0, C]$ with C the SVM error penalty. Examples with $\lambda_i = 0$ lie outside the margin and at the correct side of the hyperplane. Examples with $0 < \lambda_i < C$ also lie at the correct hyperplane side, but on the margin. Examples with $\lambda_i = C$ can lie inside the margin and at the correct side of the hyperplane, or they can lie at the incorrect side of the hyperplane. Clearly, larger Lagrange multipliers represent more nonconformity and therefore they are valid nonconformity scores, i.e., we define $\alpha_i = \lambda_i$ as the nonconformity score of example $z_i = (x_i, y_i)$ [13,14].

4 Experiments

The previous section discussed technical properties and practical implementations of TCMs. This section empirically investigates whether the calibration property holds when TCMs are applied in the off-line learning setting. We performed experiments with TCMs on a number of benchmark datasets. Subsection 4.1 briefly describes the datasets that we used. Subsection 4.2 outlines the experimental setup. Subsection 4.3 presents the results of the experiments.

4.1 Benchmark Datasets

In the following, we denote the aforementioned TCM implementations by the classifier name and the prefix TCM, e.g., TCM-kNN is the TCM based on the k-NN nonconformity measure.

We tested the six TCMs on ten well-known binary datasets from the UCI benchmark repository [11]. The datasets are: `heart statlog`, `house votes`, `ionosphere`, `liver`, `monks1`, `monks2`, `monks3`, `pima`, `sonar`, and `spect`. Some datasets such as `liver` and `sonar` are known to be highly non-linear. For these non-linear datasets, it is especially challenging to verify if TCM-LDC satisfies the calibration property. The `monks` datasets are datasets for which distance-based classifiers can have difficulties [3].

As a preprocessing step, all instances with missing feature values are removed as well as duplicate instances. Features are standardized to have zero mean and unit variance to remove possible effects caused by features with different orders of magnitude.

4.2 Experimental Setup

The classifiers TCM-kNN, TCM-KP, and TCM-SVM require the selection of one or more parameters. We performed model selection by applying a ten-fold cross validation process that was repeated for five times. The chosen parameter values are those for which the number of prediction sets with multiple labels is minimized for significance levels in the domain $[0, 0.2]$.[6] The number of nearest neighbours for TCM-kNN is restricted to $k = 1, 2, \ldots, 10$. For TCM-SVM and TCM-KP we tested polynomial and Gaussian kernels with exponent values $e = 1, 2, \ldots, 10$ and bandwidth values $\sigma = 0.001, 0.01, 0.03, 0.06, 1, 1.6$ respectively. The SVM error penalty C is kept fixed at value 10.

Once the parameter values are chosen, TCMs are applied in the off-line learning setting with ten-fold cross validation. To ensure that results are independent of the order of examples in the training folds, the experiments were repeated five times with random permutations of the data. We report the average performance of all experiments and test folds.

The performance of TCMs is measured by two key statistics. First, the percentage of prediction sets that do not contain the true label is measured. This is the *error rate* measured as a percentage. Second, we measure *efficiency* to indicate how useful the prediction sets are. Efficiency is given by the percentages of three types of prediction sets. The first type are prediction sets with one label. These prediction sets are called certain predictions. Second, uncertain predictions correspond to prediction sets with two labels and indicate that both labels are likely to be correct. Third, prediction sets can also be empty. Clearly, certain predictions are preferred.

4.3 Results

In this section we report our empirical results of off-line TCMs on the ten benchmark datasets. To visualize performance of a TCM, we follow the convention as

[6] The conclusions based on our experiments do not depend on the chosen parameter values. Other values simply result in more prediction sets with multiple labels.

defined in [17]. Results are shown as graphs indicating four values for each significance level: (1) percentage of incorrect predictions, (2) percentage of uncertain predictions, (3) percentage of empty predictions, and (4) percentage of incorrect predictions that are allowed at the significance level. The first value represents the error rate as a percentage, while the second and third values represent efficiency.[7] The line connecting the percentage of incorrect predictions allowed at each significance level is called the *error calibration line*. As an example, Fig. 1 shows the result of applying TCM-kNN and TCM-NC on the `ionosphere` dataset. Graphs of all TCMs and datasets are found in a technical report [16]. In the following we first focus our attention to the calibration property, then we give some remarks about efficiency.

TCMs satisfy the calibration property if the percentage of incorrect predictions at each significance level lies on the error calibration line. From Fig. 1 follows that the corresponding TCMs are well-calibrated up to neglectable statistical fluctuations (the empirical error line can hardly be distinguished from the error calibration line). For example, at $\epsilon = 0.05$ approximately 5% of the prediction sets do not contain the true label. Table 1 verifies the calibration property for all TCMs and datasets by reporting the average deviation between empirical errors and the the error calibration line for $\epsilon = 0, 0.01, \ldots, 0.50$. We do not consider significance levels above 0.5 since these result in classifiers for which more than 50% of the prediction sets do not contain the true label. Deviations are given in percentages and are almost zero, indicating that TCMs satisfy the calibration property when they are applied in the off-line learning setting. Note that we included datasets for which some classifiers have difficulties to achieve a low error rate (Subsection 4.1). Even for these datasets and classifiers, Table 1 reports deviations that are almost zero.

To measure efficiency we note that the percentage of uncertain predictions is 100% when $\epsilon = 0$ since the computed prediction sets contain all labels. We allow for more incorrect predictions when the significance level is set to a higher value. Therefore, the percentage of uncertain predictions monotonically decreases with higher significance levels. How fast this decline goes depends on the performance of the classifier plugged into the TCM framework. This means that k-NN performs significantly better than NC on the `ionosphere` dataset, as illustrated by Fig. 1. The percentage of empty predictions starts to occur at approximately the significance level for which there are no more uncertain predictions. The percentage of empty predictions monotonically increases after this significance level, moving closer to the error calibration line to eventually lie on this line. To summarize efficiency for the `ionosphere` dataset, we consider four significance levels that we believe to be of interest in many practical situations: $\epsilon = 0.20, 0.15, 0.10, 0.05$. For these significance levels, Table 2 reports means and standard deviations for the percentage of incorrect, certain, and empty predictions of all six TCMs. Of course, Table 2 again verifies that the calibration

[7] The percentage of certain predictions is trivially derived from the reported percentages of the other types of prediction sets. Note that the percentage of empty predictions is at most the percentage of incorrect predictions.

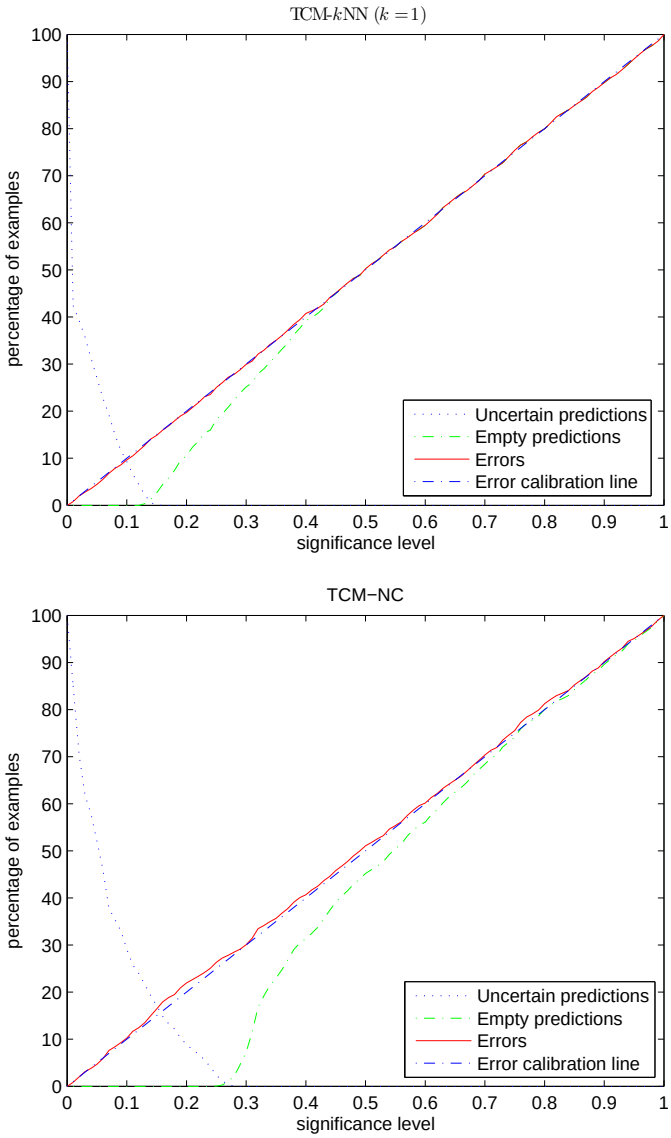

Fig. 1. Results of two TCMs applied on the `ionosphere` dataset in the off-line learning setting: (a) TCM-kNN and (b) TCM-NC.

Table 1. The deviations between empirical errors and the error calibration line. Values are reported as percentages.

	TCM-kNN	TCM-NC	TCM-LDC	TCM-NB	TCM-KP	TCM-SVM
heart statlog	0.34	0.59	0.35	0.20	0.25	0.31
house votes	0.33	0.27	0.38	0.29	0.53	0.28
ionosphere	0.21	0.81	0.31	0.28	0.33	0.38
liver	0.62	1.35	0.35	0.43	0.47	0.23
monks1	0.98	1.02	0.40	0.60	0.26	0.40
monks2	0.49	1.29	0.46	0.29	0.27	0.36
monks3	0.32	0.51	0.22	0.52	0.21	0.45
pima	0.21	0.28	0.13	0.16	0.16	0.16
sonar	0.59	1.09	0.38	0.32	0.46	0.67
spect	0.35	1.06	0.36	0.58	0.51	0.61

property holds. The reported standard deviations may not seem that small. However, the number of instances in a single test fold is small for the `ionosphere` dataset (35 test instances). All values correspond to our discussion of efficiency. Efficiency results for the other datasets are similar and presented in a technical report [16].

5 Discussion

This section elaborates more on the difference between randomized and non-randomized TCMs, and on the meaning of empty prediction sets.

In our experiments with non-randomized TCMs, we found that the line connecting the empirical errors of a non-randomized TCM-SVM is a step function that tends to stay below the error calibration line (results not shown, see [16] for an example). The reason for this observation is as follows. There are two possible scenarios when a new example is added to the training examples. First, the new example may be a support vector. The difference between the randomized p-value and the non-randomized p-value is then small since the number of support vectors with equal nonconformity score is only a small fraction of the available examples. Second, the new example may be a non-support vector. The randomized p-value is then significantly smaller than the non-randomized p-value since all non-support vectors have equal nonconformity score. This implies that the non-randomized TCM-SVM will compute less empty prediction sets than the randomized TCM-SVM. Therefore, the empirical error line becomes a step function since empty prediction sets are counted as errors. A similar reasoning holds for the difference between a non-randomized TCM-KP and a randomized TCM-KP. For the remaining TCM implementations, a non-randomized version did not led to significantly different results than a randomized version. Indeed, when the nonconformity scores take values in a large domain, then the difference between non-randomized and randomized TCMs is neglectable.

Table 2. Results of the six TCMs applied on the `ionosphere` dataset in the off-line learning setting.

classifier	% error		% certain		% empty	
ϵ	mean	std	mean	std	mean	std
TCM-kNN						
0.20	19.71	7.66	89.43	5.75	10.57	5.75
0.15	14.86	7.05	97.26	3.96	2.63	3.99
0.10	9.66	5.59	90.69	6.34	0.00	0.00
0.05	4.46	4.25	72.97	8.62	0.00	0.00
TCM-NC						
0.20	21.94	7.86	91.14	4.87	0.00	0.00
0.15	15.40	6.80	82.80	5.87	0.00	0.00
0.10	10.23	6.00	70.86	6.88	0.00	0.00
0.05	4.69	4.27	48.00	8.79	0.00	0.00
TCM-LDC						
0.20	19.71	6.86	87.60	5.96	12.34	5.96
0.15	14.69	6.38	93.71	3.74	5.43	3.75
0.10	10.00	5.24	95.31	3.86	0.11	0.57
0.05	5.14	4.32	81.71	6.76	0.00	0.00
TCM-NB						
0.20	19.88	8.20	95.42	4.20	4.57	4.20
0.15	14.74	7.12	93.82	4.91	0.00	0.00
0.10	9.71	5.88	83.82	7.67	0.00	0.00
0.05	4.80	4.18	71.82	8.62	0.00	0.00
TCM-KP						
0.20	20.11	6.88	88.86	5.51	11.09	5.59
0.15	14.74	5.90	96.11	3.20	2.06	2.77
0.10	8.80	5.22	89.83	5.39	0.00	0.00
0.05	5.37	4.57	70.40	10.01	0.00	0.00
TCM-SVM						
0.20	20.06	8.67	81.14	8.26	18.86	8.26
0.15	15.31	7.48	86.06	6.97	13.31	7.11
0.10	10.29	6.85	77.03	5.91	7.20	5.42
0.05	5.31	4.55	52.34	9.51	2.69	3.48

Empty prediction sets indicate that the classification task has become too easy: we can afford the luxury of refusing to make a prediction. Thus, empty prediction sets are a tool to satisfy the calibration property for high significance levels. In fact, the significance level for which empty prediction sets start to arise is approximately equal to the error rate of the classifier when it is not plugged into the TCM framework. To avoid empty predictions, TCMs can be modified to include the label with highest p-value into the prediction set, even though this p-value can be smaller than or equal to the significance level. In this situation, the percentage of empirical errors will also become a step function below the error calibration line since an empty prediction set was previously counted as an error. The significance level now gives an upper bound on the error rate, although we do not know how tight this bound is. The resulting TCMs are called *forced TCMs* and they are said to be conservatively well-calibrated [1].

6 Conclusions

In this paper we focused on the applicability and validity of transductive confidence machines (TCMs) applied in the off-line learning setting. TCMs allow to make predictions such that the error rate is controlled a priori by the user. This property is called the calibration property. An analytical proof of the calibration property exists when TCMs are applied in the on-line learning setting. However, this learning setting restricts the applicability of TCMs.

We provided an extensive empirical evaluation of TCMs applied in the off-line learning setting. Six TCM implementations with different nonconformity measures were applied on ten well-known benchmark datasets. From the results of our experiments we may conclude that TCMs satisfy the calibration property in the off-line learning setting, hereby strongly extending the range of tasks in which they can be applied. TCMs have a significant benefit over conventional classifiers for which the error rate cannot be controlled by the user prior to classification, especially in tasks where reliable instance classifications are desired.

Since TCMs have now been shown to be widely applicable and well-calibrated in virtually any application domain, our future work focuses on efficiency. We noticed that the chosen nonconformity measure affects efficiency while it does not violate the upper bound on the error rate. Our next goal is to minimize the size of the computed prediction sets, especially in case of multilabel learning. We believe that this can be achieved with a new nonconformity measure. Our interest is a measure that is independent of the specific TCM implementation and that is designed to provide a confidence value on nonconformity scores too.

Acknowledgments

The first author is supported by the Dutch Organization for Scientific Research (NWO), ToKeN programme, grant nr: 634.000.435. The second author is supported by NWO, CATCH programme, grant nr: 640.002.401.

References

1. Tony Bellotti. *Confidence Machines for Microarray Classification and Feature Selection*. PhD thesis, Royal Holloway University of London, London, UK, February 2006.
2. Tony Bellotti, Zhiyuan Luo, Alex Gammerman, Frederick Van Delft, and Vaskar Saha. Qualified predictions for microarray and proteomics pattern diagnostics with confidence machines. *International Journal of Neural Systems*, 15(4):247–258, 2005.
3. Enrico Blanzieri and Francesco Ricci. Probability based metrics for nearest neighbor classification and case-based reasoning. In Klaus-Dieter Althoff, Ralph Bergmann, and Karl Branting, editors, *3rd International Conference on Case-Based Reasoning and Development (ICCBR 1999)*, pages 14–28, Seeon Monastery, Germany, July 27-30 1999. Springer.
4. Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.
5. Pedro Domingos and Michael Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29(2):103–130, 1997.
6. Ronald Fisher. The use of multiple measurements in taxonomics problems. *Annals of Eugenics*, 7:178–188, 1936.
7. Alex Gammerman and Vladimir Vovk. Prediction algorithms and confidence measures based on algorithmic randomness theory. *Theoretical Computer Science*, 287(1):209–217, 2002.
8. Alex Gammerman, Vladimir Vovk, and Vladimir Vapnik. Learning by transduction. In Gregory Cooper and Serafin Moral, editors, *14th Conference on Uncertainty in Artificial Intelligence (UAI 1998)*, pages 148–155, Madison, WI, USA, July 24-26 1998. Morgan Kaufmann.
9. Roni Khardon, Dan Roth, and Rocco Servedio. Efficiency versus convergence of boolean kernels for on-line learning algorithms. *Journal of Artificial Intelligence Research*, 24:341–356, 2005.
10. Thomas Melluish, Craig Saunders, Ilia Nouretdinov, and Vladimir Vovk. Comparing the Bayes and typicalness frameworks. In Luc De Raedt and Peter Flach, editors, *12th European Conference on Machine Learning (ECML 2001)*, pages 360–371, Freiburg, Germany, September 5-7 2001. Springer.
11. David Newman, Seth Hettich, Cason Blake, and Christopher Merz. UCI repository of machine learning databases, 1998.
12. Kostas Proedrou, Ilia Nouretdinov, Vladimir Vovk, and Alex Gammerman. Transductive confidence machines for pattern recognition. Technical Report 01-02, Royal Holloway University of London, London, UK, 2001.
13. Craig Saunders, Alex Gammerman, and Vladimir Vovk. Transduction with confidence and credibility. In Thomas Dean, editor, *16th International Joint Conference on Artificial Intelligence (IJCAI 1999)*, pages 722–726, Stockholm, Sweden, July 31 - August 6 1999. Morgan Kaufmann.
14. Craig Saunders, Alex Gammerman, and Vladimir Vovk. Computationally efficient transductive machines. In Toshio Okamoto, Roger Hartley, Kinshuk, and John Klus, editors, *11th International Conference on Algorithmic Learning Theory (ICALT 2000)*, pages 325–333, Madison, WI, USA, August 6-8 2000. IEEE Computer Society Press.
15. John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge, UK, 2004.

16. Stijn Vanderlooy, Laurens van der Maaten, and Ida Sprinkhuizen-Kuyper. Off-line learning with transductive confidence machines: an empirical evaluation. Technical Report MICC-IKAT 07-03, Universiteit Maastricht, Maastricht, The Netherlands, 2007.
17. Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, New York, NY, USA, 2005.