Transductive Confidence Machines for Pattern Recognition

Kostas Proedrou, Ilia Nouretdinov, Volodya Vovk, and Alex Gammerman

Department of Computer Science, Royal Holloway, University of London
Egham, Surrey TW20 0EX, England
{konstant,ilia,vovk,alex}@cs.rhul.ac.uk

Abstract. We propose a new algorithm for pattern recognition that outputs some measures of "reliability" for every prediction made, in contrast to the current algorithms that output "bare" predictions only. Our method uses a rule similar to that of nearest neighbours to infer predictions; thus its predictive performance is close to that of nearest neighbours, while the measures of confidence it outputs provide practically useful information for individual predictions.

1 Introduction

Current machine learning algorithms usually lack measures that can give an indication of how "good" the predictions are. Even when such measures are present they have certain disadvantages, such as:

- They cannot be applied to individual test examples.
- They often are not very useful in practice (PAC theory).
- They often rely on strong underlying assumptions (Bayesian methods).

In our case none of these disadvantages are present. Our only assumption is that data items are produced independently by the same stochastic mechanism (iid assumption), our measures of confidence are applicable to individual examples, while experimental results show that they produce good results for benchmark data sets (and so potentially are useful in practice). The iid assumption that we make is a very natural one for most applications of pattern recognition, as it only implies that

- all our examples are produced by the same underlying probability distribution and
- they are produced independently of each other; so the order in which they appear is not relevant.

Many algorithms have been proposed in the past, both in the Bayesian and in the PAC settings, that provide additional information of the "quality" of the predictions.

Bayesian algorithms usually provide useful confidence values but when the underlying distribution is not known these values are "misleading". Experiments

T. Elomaa et al. (Eds.): ECML, LNAI 2430, pp. 381–390, 2002.

in (Melluish et al., 2001) have shown that in Bayesian algorithms, when the underlying probability distribution of the examples is not known, the deviation from the expected percentage of misclassified examples is too large to give any practical meaning to the confidence values. For example, we expect that from all examples with a confidence value of 90% the percentage of those wrongly classified will be close to 10%. Bayesian algorithms instead, can produce a much higher percentage of error at the above confidence level; in experiments in (Melluish et al., 2001) this error is between 20% and 40%.

PAC theory doesn't make any assumptions about the underlying probability distribution, but its results are often not useful in practice. To demonstrate crudeness of the usual PAC bounds, we reproduce an example from (Nouretdinov, Vovk et al., 2001). Littlestone and Warmuth's theorem stated in (Cristianini et al., 2000) is one of the tightest results of PAC theory, but still usually does not give practically meaningful results. The theorem states that for a two-class Support Vector classifier f the probability of mistakes is

$$err(f) \leq \frac{1}{l-d}\left(d \ln \frac{el}{d} + \ln \frac{l}{\delta}\right)$$

with probability $1-\delta$, where l is the training size and d is the number of Support Vectors. For the US Postal Service (USPS) database (described below and in Vapnik, 1998, Section 12.2), the error bound given by that theorem for one out of ten classifiers is close to

$$\frac{1}{7291-274} 274 \ln \frac{7291e}{274} \approx 0.17,$$

even if we ignore the term $\ln \frac{l}{\delta}$ (274 is the average number of support vectors for polynomials of degree 3, which give the best predictive performance; see Table 12.2 in Vapnik, 1998). Since there are ten classifiers, the upper bound on the total probability of mistakes becomes 1.7, which is not helpful at all.

Our prediction method is based on the so called algorithmic theory of randomness. A description of this theory is the subject of Section 2. Then, in Section 3, we describe our algorithm, and in the next section we give some experimental and comparison results for our algorithm as applied to the USPS and other data sets.

Our algorithm follows the transductive approach, as for the classification of every new example it uses the whole training set to infer a rule for that particular example only. In contrast, in the inductive approach a general rule is derived from the training set and then applied to each training example. For this reason we shall call our algorithm transductive confidence machine for nearest neighbours (TCM-NN). It is also possible to use the inductive approach to obtain confidence measures for our predictions (see e.g. (Papadopoulos et al., 2002) for an example of how to obtain confident predictions in the case of regression using the inductive approach).

2 Algorithmic Theory of Randomness

According to classical probability theory if we toss a fair coin n times, all sequences $\{0,1\}^n$ will have the same probability $\frac{1}{2^n}$ of occuring. We would be much more surprised, however, to see a sequence like $111111111\ldots1$ than a sequence like $011010100\ldots1$. The classical approach to probabilty theory can only give probabilites of different outcomes, but cannot say anything about the typicalness of sequences.

Intuitively, sequences that don't seem to have any specific pattern in their elements would be more typical than sequences in which one can easily find regularities. An important result of the theory of algorithic randomness is that there exists a universal method of finding regularities in data sequences.

This result is due to Martin-Löf, who was the first to introduce the notion of a randomness test. A slightly modified definition of Martin-Löf's test[1] states that a function $t : Z^* \to [0,1]$ is a test for randomness with respect to a class of probability distributions Q in Z if

- for all $n \in \mathbb{N}$, for all $s \in [0,1]$ and for all probability distributions P in Q,

$$P^n\{z \in Z^n : t(z) \leq s\} \leq s, \tag{1}$$

- t is semi-computable from above.

Here Z is a space that possesses some computability properties; in our application, Z is the set of all possible examples.

Every randomness test creates a series of nested subsets. Each subset is associated with a number s that bounds the value $t(z)$ that the test takes. We can expect that every randomness test will detect only some of the non-random patterns occuring in each sequence. Martin-Löf proved, however, that we can merge all such tests to obtain a universal test for randomness[2]. Such a test would be able to find all non-random patterns in a sequence of elements. Unfortunately, universal tests for randomness are not computable. Thus we have to approximate them using valid (in the sense of satisfying (1)) non-universal tests.

In the next section, we will give a valid randomness test for finite sequences of real numbers produced under the iid assumption and we shall use what we call a strangeness measure to map each example into a single real value in order to utilize that test for obtaining confident predictions using the nearest neighbours algorithm.

[1] The definition stated here is equivalent with Martin-Löf's original definition; the only difference being the use of the 'direct scale' (randomness values from 0 to 1), instead of the 'logarithmic scale' (randomness values from 0 to $+\infty$).

[2] A proof of the existence of universal randomness tests can be found in (Li & Vitányi, 1997), Chapter 2.4.

3 Nearest Neighbours and Randomness

3.1 Formal Setting of the Problem

We have a training set $\{(x_1, y_1), \ldots, (x_m, y_m)\}$, of m elements, where $x_i = (x_i^1, \ldots, x_i^n)$ is the set of feature values for example i and y_i is the classification for example i, taking values from a finite set of possible classifications, which we identify as $\{1, 2, \ldots, c\}$. We also have a test set of r examples similar to the ones in the training set, only this time the actual classifications are withheld from us. Our goal is to assign to every test example one of the possible classifications. For every classification we also want to give some confidence measures, valid in the sense of (1), that will enable us to gain more insight in the predictions that we make.

3.2 Nearest Neighbours Transductive Confidence Machine

Let us denote the sorted sequence (in ascending order) of the distances of example i from the other examples with the same classification y as D_i^y. Also, D_{ij}^y will stand for the jth shortest distance in this sequence and D_i^{-y} for the sorted sequence of distances containing examples with classification different from y. We assign to every example a measure called the individual strangeness measure. This measure defines the strangeness of the example in relation to the rest of the examples. In our case the strangeness measure for an example i with label y is defined as

$$\alpha_i = \frac{\sum_{j=1}^{k} D_{ij}^y}{\sum_{j=1}^{k} D_{ij}^{-y}}, \tag{2}$$

where k is the number of neighbours used. Thus, our measure for strangeness is the ratio of the sum of the k nearest distances from the same class to the sum of the k nearest distances from all other classes. This is a natural measure to use, as the strangeness of an example increases when the distance from the examples of the same class becomes bigger or when the distance from the other classes becomes smaller.

Now let us return to algorithmic randomness theory. In (Melluish et al., 2001) it is proved that the function

$$p(\alpha_{\text{new}}) = \frac{\#\{i : \alpha_i \geq \alpha_{\text{new}}\}}{m+1}, \tag{3}$$

where α_{new} is the strangeness value for the test example (assuming there is only one test example, or that the test examples are processed one at a time), is a valid randomness test in the iid case. The proof takes advantage of the fact that since our distribution is iid all permutations of a sequence have the same probability of occuring. If we have a sequence $\alpha_1, \ldots, \alpha_m$ and a new element α_{new} is introduced then α_{new} can take any place in the new (sorted) sequence with the same probability, as all permutations of the new sequence are equiprobable.

Thus, the probability that α_{new} is among the j largest occurs with probability of at most $\frac{j}{m+1}$.

The values taken by the above randomness test will be called p-values. The p-value for the sequence $\{\alpha_1, \ldots, \alpha_m, \alpha_{\text{new}}\}$, where $\{\alpha_1, \ldots, \alpha_m\}$ are the strangeness measures for the training examples and α_{new} is the strangeness measure of a new test example with a possible classification assigned to it, is the value $p(\alpha_{\text{new}})$. We can now give our algorithm.

TCM-NN Algorithm

Choose k, the number of nearest neighbours to be used
for $i = 1$ to m do
  Find and store D_i^y and D_i^{-y}
end for
Calculate alpha values for all training examples
for $i = 1$ to r do
  Calculate the dist vector as the distances of the new example
  from all training examples
  for $j = 1$ to c do
    for every training example t classified as j do
      if $D_{tk}^j > \text{dist}(t)$ recalculate the alpha value of example t
    end for
    for every training example t classified as non-j do
      if $D_{tk}^{-j} > \text{dist}(t)$ recalculate the alpha value of example t
    end for
    Calculate alpha value for the new example classified as j
    Calculate p-value for the new example classified as j
  end for
  Predict the class with the largest p-value
  Output as confidence one minus the 2nd largest p-value
  Output as credibility the largest p-value
end for

For each possible classification of a test example we construct the sequence of strangeness values of the training set augmented by the strangeness value of the new test example[3]. The prediction for each example is the classification that gives the most typical completion of the sequence of strangeness measures of the training set under the iid assumption.

Each prediction is accompanied by two other measures. The most important of them is the confidence measure. Since, by equation (1), the second largest

[3] Note that some of the strangeness values of the training set may be different for different test examples or different possible classifications assigned to a test example. In this sense our algorithm is transductive, as the training set is being reused for each test example.

p-value is an upper bound on the probability that the excluded classifications will be correct, the confidence measure indicates how likely the predicted classification is the correct one. The credibility measure gives the typicalness of the predicted classification. This value indicates how well suited the training set is for the classification of a particular test example. Low credibility would mean that the test example is strange with respect to the training examples, e.g. trying to classify a letter using a training set that consists of digits.

In principle, we would want for each prediction all p-values to be close to 0, apart from the one that gives the correct classification, that we would want to be close to 1.

4 Experimental Results

The standard comparison criterion in classification problems is the percentage of incorrectly classified examples. Here we shall also use a second one. We fix a specific significance level δ, say 1%, and we accept as possible classifications the ones whose p-value is above that level. In this way we can determine how many test examples can be classified with a confidence of at least $1 - \delta$.

We have tested our algorithm on the following datasets:

- USPS. It consists of handwritten digits from $0 - 9$. The training set consists of 7291 examples and the test set of 2007 examples. Each example has 256 attributes (pixels) that describe the given digit. All data were pre-processed as follows. As any image from the USPS data set was represented as 256 numbers $(x_1, ..., x_{256})$, we replaced it by $(y_1, ..., y_{256})$, where

$$y_i = \frac{x_i - S}{D}, S = \frac{\sum_{i=1}^{256} x_i}{256},$$

$$D = \sqrt{\frac{\sum_{i=1}^{256}(x_i - S)^2}{256}}$$

  The aim of this preprocessing is to normalise the level of brightness. After the preprocessing, the mean value of each image becomes 0 and the standard deviation becomes 1.
- Satellite. These are 6435 satellite images(4435 for train and 2000 for test). The classification task is to identify between 6 different soil conditions that are represented in the images.
- Shuttle. The classes of this dataset are the appropriate actions that should be taken under certain conditions(described by 9 attributes) in a space shuttle. There are 43500 train examples, 14500 test examples and 7 different classes.
- Segment. 2310 outdoor images described by 9 attributes each. The classifications are : brick-face, sky, foliage, cement, window, path, grass.

The last three datasets are used in the Statlog project (King et al., 1995). For comparison purposes we followed the same testing procedure. For the satellite

Table 1. Comparison of the error rate of TCM-NN with other learning algorithms

Dataset	Algorithm							
	C4.5	CART	NB	k-nn	CASTLE	Discrim	Neural	TCM
Satellite	15.1	13.8	30.7	**9.4**	19.4	17.1	13.9	10.6
Shuttle	**0.04**	-	4.55	0.44	3.77	4.83	4.9	0.11
Segment	4	4	26	7.7	11.2	11.6	-	**3.68**

Table 2. Comparison of the error percentage of TCM-NN with other algorithms on the USPS dataset

Learning Algorithms	Nearest Neighbours	TCM-NN	Support Vector Machine	Five layer Neural Network
% of error	4.29%	4.29%	4.04%	5.1%

and shuttle datasets we used the same training and test set, while for the segment one we used 10 fold cross-validation.

In Table 1 we compare the performance of our algorithm[4] with 7 others, all taken from the Statlog project, on the satellite, shuttle and segment datasets. The algorithms are two decision tree algorithms, C4.5 and CART, the Naive Bayes classifier (NB), the k-nearest neighbours algorithm, a Bayesian network algorithm (CASTLE), a linear discriminant algorithm (Discrim) and a back-propagation neural network. Two of the values from Table 1 are missing as these results are not mentioned in (King et al., 1995).

Table 2 contains experimental and comparison results on the USPS dataset. The error percentage for the Five Layer Neural Network was obtained from (Vapnik, 1998), while for the other three algorithms the results were produced by the authors on the same training and test set. It is clear from both tables that TCM-NN's performance is almost as good as the performance of the best algorithms for all datasets used.

Next, in Figure 1 we compare the error rate of TCM with that of the original nearest neighbours algorithm on the USPS dataset using a different number of neighbours each time. Though the perfromace of both algorithms is decreasing as the number of neighbours is increasing it seems that TCM is more robust as its error rate is increasing much slower.

When the second comparison criterion is used, our algorithm makes 'region' predictions (outputs a set of classifications) instead of point predictions. For a specified significance level δ the correct classification will be in the predicted set of classifications with a probability of at least $1 - \delta$, since the set of rejected classifications can occur with probability of at most δ. In Figure 2 we demonstrate this relationship between error classification and confidence level using 50 random instances of the USPS dataset.

[4] We normally use one nearest neighbour for testing TCM-NN. When this is not the case, the number of neighbours used will be stated explicitly.

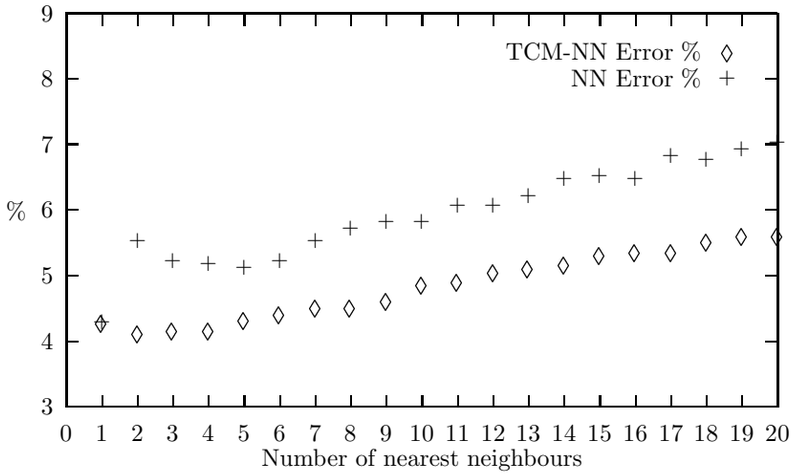

Fig. 1. Error percentage of TCN-NN and NN on the USPS dataset using 1-20 nearest neighbours

In Table 3 we detail the results of 'region' predictions for significance levels of 1% and 5%, giving the percentage of examples for which the predicted set contains one label, more than one label and no labels. For the shuttle and USPS datasets we predict a set containing one classification for 99.17% and 94.77% of the examples respectively with great certainty (confidence of 99% or more). We can also note that as the overall error rate is increasing the number of examples that can be given a single classification is decreasing. Since greater error percentages mean more difficult classification problems it is natural that more examples will be assigned more than one possible classifications.

Finally, the last column in Table 3 gives the percentage of examples of the 'one class' column that were correctly classified. These percentages are very close and in most cases higher than the corresponding confidence levels; thus indicating the practical usefulness of TCM's confidence measure[5].

5 Conclusion

The TCM-NN algorithm presented here has the advantage of giving probabilistic measures for each individual prediction that we make. In this way we gain more insight into how likely a correct classification is for an example when given a specific training set. Furthermore, the percentage of errors of TCM-NN seems to be as good as that of other learning algorithms.

[5] Note that choosing a smaller significance level doesn't necessarily guarantee a greater rate of success, as we only consider the examples that are assigned one classification. The former holds only when we consider all test examples (see Figure 2).

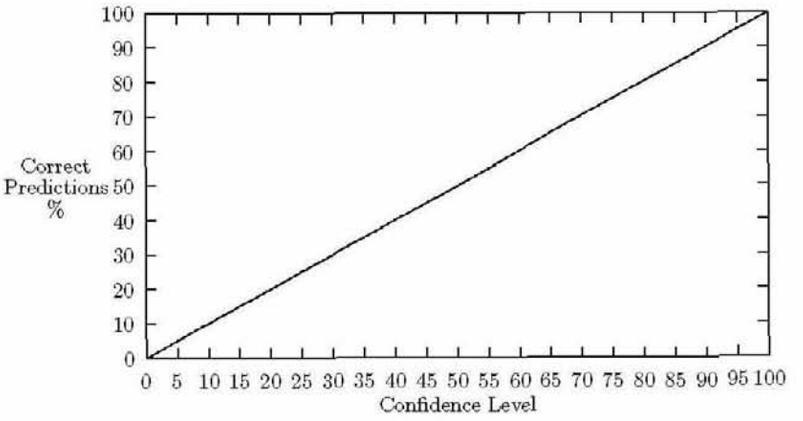

Fig. 2. Percentage of correct 'region' predictions for different confidence levels using 50 random instances of the USPS dataset

The scheme we have proposed can be used on top of every classification algorithm and not only nearest neighbours, by defining the individual strangeness measure (2) in a different way. For example, the method can be applied to the Support Vector Machine algorithm using as a strangeness measure the distance of each example from the hyper-plane that separates the different classes. Finally, as an approximation to the universal test defined in Section 2 we have used the

Table 3. TCM-NN Performance. The column "One class" gives the number of examples for which a confident prediction is made, the column "$\geq$ 2 classes" gives the number of examples for which two or more possible classifications were not excluded at the given significance level, and the column "No class" gives the number of examples for which all possible classifications were excluded at the given significance level. The last column shows the percentage of correct predictions for the examples we could confidently predict at each significance level

Dataset	Level	One class	≥ 2 classes	No class	Correct Predictions
USPS	1%	94.77%	5.23%	0%	97.9%
	5%	92.38%	0%	7.62%	98.4%
Satellite	1%	64.4%	35.6%	0%	98.8%
	5%	86.5%	13.5%	0%	94.9%
Shuttle	1%	99.17%	0%	0.83%	99.9%
	5%	94.99%	0%	5.01%	99.9%
Segment	1%	90.04%	9.96%	0%	100%
	5%	97.36%	0%	2.64%	99.1%

statistical p-test (3). It remains an open problem though whether one can find valid tests for randomness (under the general iid assumption) that are better approximations to the universal tests for randomness than the one used here.

Acknowledgements

This work was partially supported by EPSRC through grants GR/L35812 ("Support Vector and Bayesian learning algorithms"), GR/M14937 ("Predictive complexity: recursion-theoretic variants"), and GR/M16856 ("Comparison of Support Vector Machine and Minimum Message Length methods for induction and prediction"). We are grateful to the Program Committee for useful comments.

References

1. Cristianini, N., & Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-based Methods*. Cambridge: Cambridge University Press.
2. Fraser, D. A. (1976). *Non-parametric methods in statistics*. New York: Wiley.
3. King, R. D., Feng, C., & Sutherland, A. (1995). *Statlog: Comparison of classification algorithms on large real-world problems*. Applied Artificial Intelligence, 9(3), pp 259–287.
4. Li, M., & Vitányi, P. (1997). *An introduction to Kolmogorov complexity and its applications* (2nd edn.). New York: Springer.
5. Melluish, T., Saunders, C., Nouretdinov, I., & Vovk, V. (2001). *Comparing the Bayes and typicalness frameworks*. In *Proceedings of ECML'2001.*
6. Nouretdinov, I., Melluish, T., Vovk V. (2001). *Ridge Regression Confidence Machine*. In *Proceedings of the 18th International Conference on Machine Learning.*
7. Nouretdinov, I., Vovk, V., Vyugin, M., & Gammerman, A. (2001). *Pattern recognition and density estimation under the general iid assumption*. In *Proceedings of the Fourteenth Annual Conference on Computational Learning Theory and Fifth European Conference on Computational Learning Theory.*
8. Papadopoulos, H., Proedrou, K., Vovk, V., Gammerman, A. (2002). *Inductive Confidence Machines for Regression.*. In *Proceedings of ECML'2002.*
9. Vapnik, V. (1998). *Statistical Learning Theory*. New York: Wiley.
10. Vovk, V., & Gammerman, A. (2002). *Algorithmic Theory of Randomness and its Computer Applications*. Manuscript.