Proceedings of the 9th International Conference on Information Technology and Applications in Biomedicine, ITAB 2009, Larnaca, Cyprus, 5-7 November 2009

Evolutionary Conformal Prediction for Breast Cancer Diagnosis

A. Lambrou, H. Papadopoulos, A. Gammerman

Abstract—**Conformal Prediction provides a framework for extending traditional Machine Learning algorithms, in order to complement predictions with reliable measures of confidence. The provision of such measures is significant for medical diagnostic systems, as more informed diagnoses can be made by medical experts.**

In this paper, we introduce a Conformal Predictor based on Genetic Algorithms, and we apply our method on the Wisconsin Breast Cancer Diagnosis (WBCD) problem. We give results in which we show that our method is efficient, in terms of accuracy, and can provide useful confidence measures.

Index Terms— *Conformal Prediction, Genetic Algorithms, Confidence, Medical Diagnosis, Breast Cancer*

I. Introduction

Machine learning algorithms have been widely used for medical diagnostic systems. Such algorithms are trained to provide predictions for the diagnoses of new patients, based on past medical data. Most of the time the predictions are correct, but generally there is no indication on how accurate each prediction is likely to be. Conformal Prediction (CP) provides a framework for assigning reliable confidence measures to predictions. This is a considerable advantage, since confidence measures for medical diagnostic systems are of great importance [1].

In CP, the output is either the prediction for an instance together with a confidence measure and a credibility value, or a set of multiple alternative predictions. Such a set defines a "predictive region" which is needed for satisfying a desirable threshold of confidence. The confidence measures produced by Conformal Predictors are valid (for a proof, see [2]). Nevertheless, the performance of a CP relies on an underlying algorithm, which uses some classical Machine Learning technique. The underlying algorithm provides predictions, and CPs are built on top to complement the predictions with measures of confidence. Different versions of CPs have been developed using Artificial Neural Networks, *k*-Nearest Neighbors, and Support Vector Machines (described in [2]). Moreover, CPs have been applied to medical diagnostic problems in [3, 4, 5].

In this paper, we introduce rule-based Genetic Algorithms (GAs) as a method for building a CP, and we apply the resulting algorithm to the problem of breast cancer diagnosis. GAs are largely used as a Machine Learning method for providing predictions. Additionally, the evolved general rules, which represent a model, are human readable and can be understood by medical experts. We use the Wisconsin Breast Cancer Diagnosis (WBCD) data-set [6], which is popular in this domain. We conduct experiments on the data-set, and provide results that demonstrate the accuracy of our predictor and the usefulness of the confidence measures. The WBCD data-set was recorded at the University of Wisconsin Hospital, and contains features which are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass [7]. The cases can be classified as malignant or benign.

Work on the WBCD problem has been conducted in [7, 8, 9], where other Machine Learning methods have been used for providing predictions. Similarly, in [10], a GA combined with fuzzy systems is used for prediction and provision of possible confidence values. Nevertheless, the confidence values defined in [10] have no probabilistic interpretation.

II. Conformal Prediction

Conformal Predictors learn to make predictions from a training set of data. A training set with n instances is of the form $\{(x_1, y_1),...,(x_n, y_n)\}$, where x_i is a vector of real-valued attributes and y_i is a label given to the instance x_i.

For a new instance x_{n+1}, we wish to predict the label y_{n+1} (i.e. the class of the instance). In order to make a prediction, we try all possible labels for the new instance and we test for each case how likely the prediction is of being correct. Initially, we append the new instance x_{n+1} in our training set together with an assumed label y_{n+1}. We then calculate a non-conformity score for each instance in the extended training set. A non-conformity score indicates how different (or strange) an instance (x_i, y_i) is, compared to the other instances. The non-conformity scores are based on an underlying algorithm, and the way they are calculated in this paper is described in section III.

Based on the assumption that the instances are independently and identically distributed (i.i.d.), we measure how likely the extended training set (in fact, multiset) is of being i.i.d. using the p-value function:

Manuscript received July 10, 2009. This work was supported by the Cyprus Research Promotion Foundation through research contract PLHRO/0506/22 ("Development of New Conformal Prediction Methods with Applications in Medical Diagnosis").

A. Lambrou is with the Computer Learning Research Centre, Royal Holloway, University of London, Egham Hill, Egham, Surrey TW20 0EX, England. (phone: +44 1784 414024, e-mail: A.Lambrou@cs.rhul.ac.uk)

H. Papadopoulos is with the Computer Science and Engineering Department, Frederick University, 7 Y. Frederickou St., Palouriotisa, Nicosia 1036, Cyprus. (e-mail: H.Papadopoulos@frederick.ac.cy)

A. Gammerman is with the Computer Learning Research Centre, Royal Holloway, University of London, Egham Hill, Egham, Surrey TW20 0EX, England. (e-mail: A.Gammerman@cs.rhul.ac.uk)

978-1-4244-5379-5/09/$26.00 ©2009 IEEE

$$p((x_1,y_1),...,(x_{n+1},y_{n+1})) = \frac{\#\{i=1,...,n+1 : a_i \geq a_{n+1}\}}{n+1}, \quad (1)$$

which compares the non-conformity score a_{n+1} of (x_{n+1},y_{n+1}) with all the other non-conformity scores. For the wrong predictions y_{n+1} we expect that a_{n+1} will be significantly higher than the rest of the non-conformity scores. In such cases, we will get low p-values (i.e. the multiset is less likely to be i.i.d.), whereas for the correct prediction we expect a higher p-value (i.e. the multiset is more likely to be i.i.d).

Given an i.i.d. multiset, the p-value in (1) satisfies the following property:

$$\mathrm{P}\left(p((x_1,y_1),...,(x_{n+1},y_{n+1})) \leq \varepsilon\right) \leq \varepsilon, \quad (2)$$

where ε is a given significance level, such that $1-\varepsilon$ is a desirable confidence level. The property describes that when the multiset contains i.i.d. instances, the probability of the p-value of the multiset to be less than or equal to ε, is less than or equal to ε. Consequently, we may output a set of possible predictions (i.e. the "predictive region"), which contains all the predictions with p-values greater than the significance level. We always include the highest prediction in order to ensure that the "predictive region" will have at least one prediction:

$$S = \left\{y_{n+1} : p_{y_{n+1}} > \varepsilon\right\} \cup \left\{\arg\max_{y_{n+1}}(p_{y_{n+1}})\right\}, \quad (3)$$

where $p_{y_{n+1}}$ is the p-value of the extended multiset for the label y_{n+1} of the new instance x_{n+1}. Because of the property in (2), the probability of each set of predictions not containing the correct prediction will be less than or equal to ε. As a result, we have a predictor that will be correct $1-\varepsilon$ of the time, and thus we can say that we have $1-\varepsilon$ confidence for our predictions. Alternatively, the CP may output a single prediction, which is the prediction with the highest p-value, complemented with a confidence measure which is one minus the second highest p-value, and a credibility value which is the p-value of the prediction.

III. Genetic Algorithm Approach

We have implemented a ruled-based GA in order to evolve decision rules which can be used as models for calculating non-conformity scores. GAs are inspired by natural evolution: A population of decision rules (named "chromosomes") is evolved through generations using genetic-like operations, such as cross-over and mutation. At each generation, the rules are selected probabilistically based on their fitness, in order to generate off-springs and create the next generation. Each candidate is evaluated against an objective function in order to gain a fitness score. In a learning system, the objective function is the measure of the accuracy of a decision rule over a training set of instances.

In this work we use the "Pittsburgh" approach, where each "chromosome" in the population is a decision rule, and each decision rule is composed by R rules. We evolve a population for each class, and we pick the best decision rule from each final population in order to form our final model.

A. Decision rule representation

We use fuzzy decision rules for our GA implementation, since fuzzy rules can give degrees of membership and are useful for calculating non-conformity scores. In addition, fuzzy rules for GAs have been used in other studies [10, 11].

A fuzzy space is defined by l fuzzy-set membership functions. Fig. 1 depicts 5 triangular fuzzy-sets: Small (S); Small-Medium (SM); Medium (M); Medium-Large (ML); and Large (L). For every attribute $x_i \in [0,1]$, we calculate its membership to each of the 5 fuzzy-sets. Each decision rule requires the membership of each attribute j of an instance x_i (denoted as x_{ij}) to some fuzzy-sets in the pre-defined fuzzy space. For example, a decision rule of 2 attributes can be the following statement:

IF [(x_{i1}= *S, x_{i1}= SM*) *and* (x_{i2} = *ML*)] *or*
IF [(x_{i1}= *S*) *and* (x_{i2} = *L*)] *then output = class* [*weight*].

The first rule in the decision rule requires the memberships of attribute x_{i1} to the S and SM fuzzy-sets, and for attribute x_{i2} the membership to the ML fuzzy-set. The above decision rule can then be represented in binary bits as:

$$\underbrace{11000}_{r_1}\underbrace{00010}_{r_2}\underbrace{10000}_{r_1}\underbrace{00001}_{r_2},$$

where each r_j is a "gene" of the "chromosome" containing l bits, and the k-th bit in each "gene" corresponds to the k-th fuzzy-set in the fuzzy space. Logical operators are not encoded in the "chromosome", since they are fixed: "genes" are connected with "and", while rules are connected with "or". The output class is also not encoded, since the algorithm evolves a population for each class separately. We calculate the compatibility degree β_{r_j} of a real-valued attribute x_{ij} to "gene" r_j as the sum of the required memberships:

$$\beta_{r_j}(x_{ij}) = \sum_{k=1}^{l}[\mu_{r_{jk}}(x_{ij}) \times r_{j_k}]. \quad (4)$$

Further, given an instance x_i with m real-valued attributes and the "genes" of rule r in the "chromosome", we define the compatibility degree of the attribute j to the rule, as:

$$\beta_r(x_i) = \min\left\{\beta_{r_j}(x_{ij}) : j = 1,...,m\right\}, \quad (5)$$

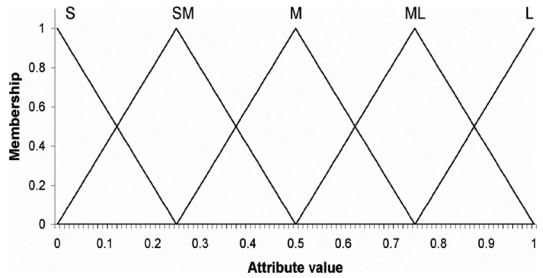

Fig. 1. Triangular fuzzy-set membership functions within a fuzzy-space for real-valued attributes of the range [0, 1].

where the minimum function is the equivalent *fuzzy-and* operation. Finally, we define the output of a decision rule D containing R rules, given instance x_i, as the weight:

$$w_D^c(x_i) = \max\left\{\beta_{D_r}(x_i) : r = 1,..., R\right\}, \tag{6}$$

where c is the class of the decision rule, and the maximum function is the equivalent *fuzzy-or* operation.

B. Genetic operations

We use two point variable size cross-over between two "chromosomes" in order to generate two off-springs. In two point cross-over we randomly select two points in the "chromosomes" and we create the off-springs by swapping the bits of the parents from the two points.

Mutation is a simple flip operation to a random bit in a "chromosome". The mutation operation is greatly important in GAs, since populations otherwise tend to converge to local maxima, due to the cross-over operation.

C. Objective function

Our objective function measures the accuracy of a decision rule based on a training set of instances. To calculate the accuracy of a decision rule, we evaluate the decision rule against all training instances and we find the weight value returned for each instance.

For the instances with the same class as the decision rule, such that $y_i=c$ we apply the following formulae to calculate the True Positives (*TP*) and False Negatives (*FN*):

$$TP = \sum_{i=1,..,n+1:y_i=c} \sqrt{w_D^c(x_i)}\,, \tag{7}$$

$$FN = \sum_{i=1,..,n+1:y_i=c} \sqrt{1 - w_D^c(x_i)}\,. \tag{8}$$

For the examples with a different class from the decision rule, such that $y_i \neq c$, we apply the same formulae (7) and (8) for False Positives (*FP*) and True Negatives (*TN*) respectively. The fitness score of the decision rule after processing all the training set is defined as

$$fitness = \frac{TP}{TP + FN} + \frac{TN}{FP + TN}. \tag{9}$$

D. Non-conformity measures

The fittest decision rules evolved from the GA can generate non-conformity scores for given instances. The most natural way to define a non-conformity measure is to reverse the weight function of the decision rule:

$$a_i = 1 - w_D^c(x_i)\,, \tag{10}$$

where D is the corresponding (evolved) decision rule for the class given to the instance x_i.

Other non-conformity measures can be defined with slightly more sophisticated calculations. Generally, we would like to use as much information we can derive from the decision rules. Therefore, we may include the weights returned by all of the decision rules which are defined for each class. Moreover, we have re-defined our weight function in (6) to include the quality of each rule:

$$W_D^c(x_i) = \max\left\{\beta_{D_r}(x_i) Q_{D_r} : r = 1,..., R\right\}. \tag{11}$$

We calculate the quality Q of each rule over the training set as the proportion of the compatibility degrees of the instances that have the same label as the rule, over the sum of the compatibility degrees of all the instances. Our second definition of a non-conformity measure is then as follows:

$$a_i = \frac{\sum_{c=1,..,y,c\neq u} W_D^c(x_i)}{W_D^u(x_i)\gamma}, \tag{12}$$

where W_D^u is the weight of the decision rule for the class of the given instance x_i, and γ is a chosen constant which adjusts the level of sensitivity to changes of W_D^u.

Our third definition of a non-conformity measure uses the same information as in (12), but with a slight modification:

$$a_i = \left[\sum_{c=1,..,y,c\neq u} W_D^c(x_i)\right] - W_D^u(x_i)\gamma\,. \tag{13}$$

E. The Conformal Predictor

Our GA evolves decision rules which can calculate non-conformity scores for our CP. We can apply the CP method and provide predictions with confidence levels, or provide sets of predictions for instances, given desirable error rates.

The method is as follows: We append a new instance we would like to predict in the training set, giving it a possible label. We apply the GA using the extended training set and we get a decision rule for each class. Then, for each instance in the training set, we calculate a non-conformity score, based on the evolved decision rules, using (10), (12) or (13). We find the p-value of the extended training set and we store it in order to compare it with the rest of the p-values. After

we have tried every possible label for the new instance and have generated our p-values, we use the method described in section II to output predictions with confidence measures.

IV. Experimental Setting and Results

We have conducted experiments on the WBCD data-set which contains 699 instances with 9 integer valued attributes for each instance. From the 699 instances, we have discarded 16 cases which contain unknown values, thus the results shown in this section are on the remaining 683 instances. Moreover, we have normalized the data to real-valued attributes within the range [0, 1].

We apply ten-fold cross validation on the data-set. That is, we split the data-set into ten equally sized blocks, and at each fold *i*, we leave out the *i*-th block as the test-set. In this section, we show the average results of the ten folds.

Our GA evolved variable sized decision rules with a maximum of 4 rules in each decision rule. This maximum number of rules seems to be sufficient for this data-set as explained in [10], and was chosen in order to limit the computational time required by the GA. The parameters we used for our GA are: population size=100; generations=100; crossover rate=0.8; selection rate=0.8, mutation rate=0.01. Each "chromosome" in the initial population is created by randomly setting each of its bits to 0 or 1 with the probability of 1s set to 0.9. We have identified from empirical results that our GA converges faster when the rate of 1s is higher, since the initial decision rules become more general.

In Table I, we compare the accuracy of our GA with the best results from other work which has been conducted on the WBCD data-set. Our goal is not to provide better accuracy; rather we aim to show the confidence measures we provide for the predictions. From the results in Table I, we can confirm that our GA implementation is accurate enough, giving 97.20% accuracy on average.

Table I
Accuracy Comparison with Other Methods

Method	Accuracy (%)
Setiono and Liu [8]	97.21
Taha and Ghosh[9]	96.19
Pena Reyes and Sipper [10]	97.8
Our GA	97.20

In Table II, we show the certainty and the error rates of the CP with non-conformity measures (12) and (13), given four confidence levels: 99%; 98%; 95%; and 90%. The certainty is measured in terms of how many "predictive regions" contain only a single prediction (i.e. only a single p-value is above a given significance level for the new instance we wish to predict). We do this, in order to test the efficiency of our confidence measures, since we would like to have as many certain predictions as possible, given high confidence levels.

When using the non-conformity measure in (13), we get 100% of certain predictions for the confidence level of 90%, and for 95% confidence we get 99.5% certain predictions. At confidence level 98% we still have 94.8% certainty, and for 99% confidence, we have 82.4% certainty. The error rates (which are for the "predictive regions" that did not contain a correct prediction) confirm the validity of our CP, since for a given confidence level $1-\varepsilon$ the error rate is near ε.

Table II
Certainty and Error Rates at Different Confidence Levels

Non-conformity Measure	Confidence Level	Certainty	Error
(12) with $\gamma=1$	99%	81.7%	0.7%
	98%	94.0%	1.7%
	95%	97.8%	4.2%
	90%	98.7%	6.8%
(13) with $\gamma=1.5$	99%	82.4%	1.1%
	98%	94.8%	1.7%
	95%	99.5%	4.8%
	90%	100%	9.0%

V. Conclusion

In this paper we have developed a new Conformal Prediction method based on Genetic Algorithms, and we have applied our method on a real-world problem for medical diagnosis. We have demonstrated in the results that our approach is accurate and can provide efficient results with high confidence levels.

References

[1] H. Holst, M. Ohlsson, C. Peterson, and L. Edenbrandt, "Intelligent computer reporting 'lack of experience': a confidence measure for decision support systems", in *Clinical Physiology*, 1998, pp. 139-147.

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

[3] H. Papadopoulos, A. Gammerman and V. Vovk, "Confidence Predictions for the Diagnosis of Acute Abdominal Pain", In Proc. *5th IFIP Conference on Artificial Intelligence Applications & Innovations,* AIAI 2009, pp. 175-184.

[4] T. Bellotti, Z. Luo, A. Gammerman, "Reliable classification of childhood acute leukaemia from gene expression data using confidence machines", in Proc. *IEEE International Conference on Granular Computing*, 2006, pp. 148–153.

[5] A. Gammerman, I. Nouretdinov, B. Burford, A. Chervonenkis, V. Vovk, and Z. Luo, "Clinical mass spectrometry proteomic diagnosis by conformal predictors." *Statistical applications in genetics and molecular biology*, vol. 7, no. 2, 2008.

[6] A. Asuncion and D.J. Newman, *UCI Machine Learning Repository*, Irvine, CA: University of California, School of Information and Computer Science. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html.

[7] O. L. Mangasarian and W. H. Wolberg, "Cancer diagnosis via linear programming", in *SIAM News*, vol. 23, num 5, 1990, pp 1 & 18.

[8] R. Setiono, "Extracting rules from pruned neural networks", in *Artificial Intelligence in Medicine,* vol. 8, issue 1, 1996, pp. 37-51.

[9] I. Taha, and J. Ghosh, "Evaluation and ordering of rules extracted from feed forward networks", in Proc. *IEEE International Conference on Neural Networks*, 1997, pp. 221-226.

[10] C. Pena Reyes, and M. Sipper, "A fuzzy-genetic approach to breast cancer diagnosis", in *Artificial Intelligence in Medicine*, vol. 17, issue 2, 1999, pp. 131-155.

[11] H. Ishibuchi, and T. Nakaskima, "Improving the performance of fuzzy classifier systems for pattern classification problems with continuous attributes". *IEEE Trans. on Industrial Electronics*, vol. 46, no. 6, 1999, pp. 1057-1068.