Decision Tree Extraction from Trained Neural Networks

Darren Dancey and **Dave McLean** and **Zuhair Bandar**

Intelligent Systems Group
Department of Computing and Mathematics,
Manchester Metropolitan University,
Chester Street, Manchester, M1 5GD.
United Kingdom.
d.dancey@mmu.ac.uk

Abstract

Artificial Neural Networks (ANNs) have proved both a popular and powerful technique for pattern recognition tasks in a number of problem domains. However, the adoption of ANNs in many areas has been impeded, due to their inability to explain how they came to their conclusion, or show in a readily comprehendible form the knowledge they have obtained.

This paper presents an algorithm that addresses these problems. The algorithm achieves this by extracting a Decision Tree, a graphical and easily understood symbolic representation of a decision process, from a trained ANN. The algorithm does not make assumptions about the ANN's architecture or training algorithm; therefore, it can be applied to any type of ANN. The algorithm is empirically compared with Quinlan's C4.5 (a common Decision Tree induction algorithm) using standard benchmark datasets. For most of the datasets used in the evaluation, the new algorithm is shown to extract Decision Trees that have a higher predictive accuracy than those induced using C4.5 directly.

Introduction

The two main approaches to machine learning have been Artificial Neural Networks(ANNs) and symbolic learning algorithms. ANNs characteristically produce models that are capable of generalizing to previously unseen data (prediction). However, ANN's do not explicitly reveal the reasoning behind their decisions. Conversely, symbolic learning methods, do not generalize as well as ANNs, but present the explanation behind their reasoning explicitly. This paper presents a method that extracts a symbolic representation from the knowledge embedded within an ANN. Therefore combining the predictive accuracy of an ANN with the advantage of an explicit explanation provided by a symbolic model.

Artificial Neural Networks

The field of Artificial Neural Networks consists of a large collection of models and techniques originally inspired by biological nervous systems such as the human brain. ANNs are based around a number of individual models of neurons (figure 1) arranged in a network. These artificial neurons accept a number of *weighted* inputs and process these inputs to produce an output. It is the value of these weights that determine the function of the ANN. Using the backpropagation algorithm (Rumelhart, Hinton, & Williams 1986), Multilayer Perceptrons (MLPs) are able to learn non-linear mappings. It is this type of model that will be used throughout this paper. A typical two layer MLP is shown in figure 2.

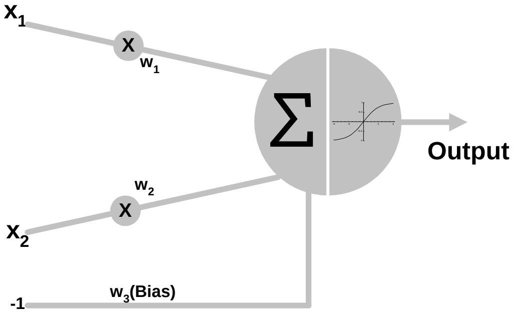

Figure 1: An Artificial Neuron

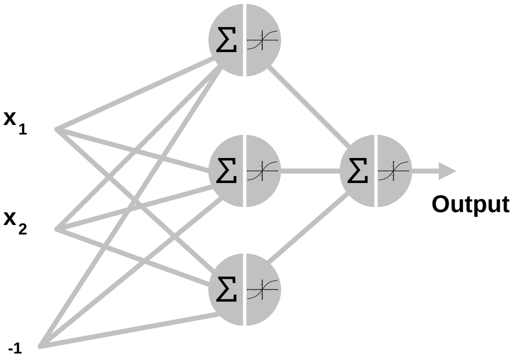

Figure 2: A two layer Multilayer Perceptron

Decision Trees

Decision Trees are one of the most widely used classifier models(Michie, Spiegelhalter, & Taylor 1994). Decision

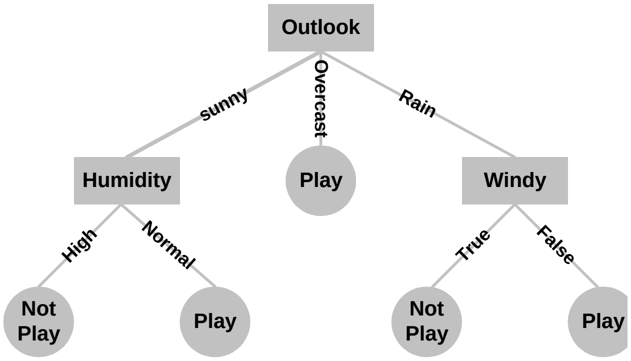

Figure 3: Decision for Quinlan's Play–Not Play Example

Trees are directed acyclic graphs consisting of nodes and connections (edges) that illustrate decision rules. Each non-terminal node has a splitting test associated with it, which splits the data into mutually exclusive subsets. The terminal nodes called leaves represent a classification. A Decision Tree for the Quinlan's classic 'play/not play tennis' example(Quinlan 1986) is shown in figure 3.

To make a decision using a Decision Tree start at the root node and follow the tree down the branches, according to the tests for the instance being classified, until a leaf node representing the class is reached. Although Decision Trees are very simple to understand, the method of creating a decision tree from examples is a nontrivial task, in fact, it has been shown to be NP complete(Hyafil & Rivest 1976).

Rule Extraction From Multilayer Perceptrons

Multilayer Perceptron's (MLP's) greatest weakness is their lack of transparency. Unlike decision trees, which show their reasoning explicitly, MLPs hide their knowledge in the complex interrelationships of their weights. This means that although MLPs often provide excellent models for prediction, they provide no insight into the relationships between input values and output values that the model may have found(Andrews, Diederich, & Tickle 1995). For example, Rothwell(2002) has created an ANN that can classify a persons responses as either deceptive or truthful, using clues in their nonverbal behaviour (eye moments, shrugs etc) but although the ANN has good predictive accuracy it does not reveal the relationships it has found between nonverbal behaviour and deception.

The aim of rule extraction is to reduce the complexity of an ANN into a more easily understood symbolic form. These rules can then be analyzed for trustworthiness for safety critical systems or used to provide insights into the relationships found by the ANN.

There have been two main approaches to extracting rules from trained ANNs decompositional and pedagogical(Craven & Shavlik 1994a). The decompositional approach examines the individual weights of the underlying ANN. This approach is typified by the KT algorithm(Fu 1995). The second approach to rule extraction is the pedagogical approach. This approach is typified by the Trepan algorithm(Craven & Shavlik 1994b). This approach treats the ANN like a 'black box', and uses a symbolic learning algorithm to 'learn' the rules which represent the mapping the ANN has found.

ExTree

ExTree is an algorithm(figure 5) for extracting Decision Trees from trained ANNs. ExTree is an example of the pedagogical approach to rule extraction. ExTree uses Craven's querying and sampling method (Craven & Shavlik 1995), but unlike Craven's Trepan, which uses MofN based splits(Murthy 1995), ExTree uses standard splitting tests like CART and C4.5.

The standard Decision Tree induction algorithms have the limitation that the selection of the splitting test is based on fewer and fewer instances as the tree grows downwards. Therefore, the splitting tests that are near the bottom of the tree are often poorly chosen because they are based on less data. ExTree alleviates this problem by generating new instances then querying the ANN (which acts as an oracle) with the newly created instances. ExTree can then select a splitting test based on the newly created instances as well as the original dataset.

ExTree requires a trained ANN to act as an oracle. In the next section ExTree is applied to trained MLPs but ExTree could be as easily applied to other ANN types such as trained Radial Basis Function networks or even other pattern recognition techniques which are opaque. ExTree does not require the ANN to use a special training algorithm or architecture only that it maps the input space to 1 of K classes. Once a trained ANN is available ExTree proceeds in a similar manner to Decision Tree induction algorithms recursively splitting the tree by finding the best feature to split on.

Split Types

ExTree considers two types of tests: for discrete features ExTree creates a branch for each possible value of the feature, for continuous numeric features a binary split is made with two outcomes $A \leq Z$ and $A > Z$. The threshold value Z is determined by first sorting the set of instances on the value of feature A. For a set with m unique values for feature A there will be $m - 1$ possible split points that could partition the set into two. ExTree chooses a split point halfway between the bounding values.

Split Selection Measure

To determine which one of the possible splits to use, ExTree uses a modification of Information Gain. Information Gain has a bias towards selecting tests with many outcomes. Quinlan(1999) proposed a modification to Information Gain giving Information Gain Ratio. Gain Ratio is determined by dividing the Information Gain by the Information gained solely by splitting the data into the number of outcomes resulting from the test. The information gained by arbitrarily

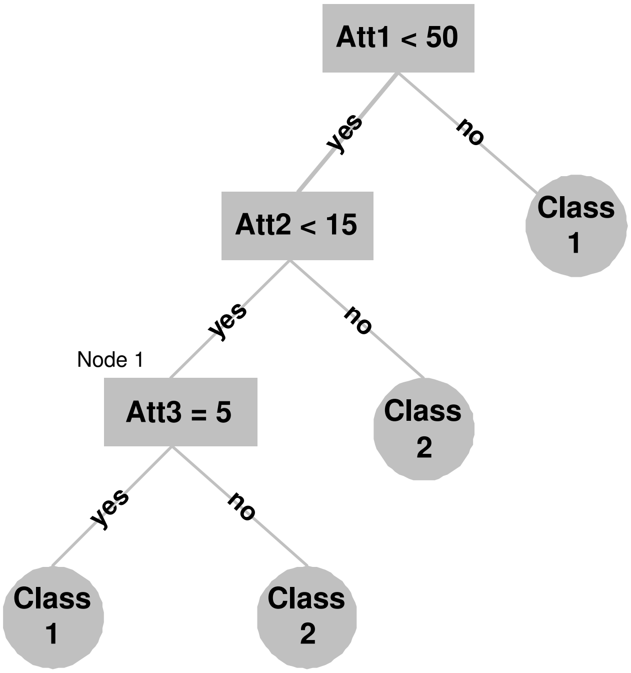

Figure 4: A Decision Tree demonstrating a constraint

splitting a set S into n subsets is given by

$$\text{split info}(X) = \sum_{i=1}^{n} \frac{|S_i|}{|S|} \times \log_2 \left(\frac{|S_i|}{|S|} \right). \qquad (1)$$

The gain ratio of test X can thus be calculated as

$$\text{gain ratio}(X) = \frac{\text{gain}(X)}{\text{split info}(X)}. \qquad (2)$$

Oracle Querying

As previously stated the advantage of pedagogical approaches such as ExTree is that new instances can be created and classified by the ANN. ExTree is able to create these new instances by maintaining a set of constraints which flows down the tree with the training instances. These constraints specify what conditions an instance must have satisfied to have reached a node as determined by the splitting tests above. For example, new instances created at node 1 in figure 4 must satisfy the constraints: $Att1 < 50$ and $Att2 < 15$. Given these constraints, new instances can created by sampling linearly in the area of input space delimited by the constraints. Currently ExTree makes an extra 100 extra instances at ever split point but ideally the number of extra instances created would be adjusted to suite the dataset.

Pruning

ExTree only stops growing the tree when the set of instances reaching a node all belong to the same class or the instances can not be split any further. For the majority of datasets which contain noise this will lead to overfitting. ExTree uses a form of post-pruning to create smaller trees that should generalize better and be more comprehendible. Before training, 33% of the training data is set-aside as a validation set. ExTree uses the pruning method of subtree replacement. Starting at the leaves and working back towards the root, each subtree is tested using the validation set to determine whether the replacement would be beneficial. If the tree with the replacement has a lower error then the subtree is replaced.

```
Extree( dataset $S$, constraints $Const$ )
BEGIN
NewInstances :=
      Create $N$ new instances constrained
by $Const$;
FOR each instance in NewInstances
   Label instance using ANN
$S$ := $S$ + NewInstances;
IF all $S$ belongs to $C_k$ THEN
   label node as leaf $C_k$
9     RETURN
ELSE
   Find Best Split $S*$
   Split the $S$ into subsets $S_{1..}$S$_n$according
to $S$*
   FOR each subset$S_i$
   BEGIN
      IF the number of instances in
subset is 0
      THEN mark node as dominating class
of parent
ELSE IF node is a mixture of classes
   Create new Constraint $Const_i$ from $Const$,
   ExTree(oracle,$S_i$,$Const_i$)
END
```

Figure 5: ExTree Algorithm

Empirical Evaluation of Extree

ExTree was evaluated using benchmark machine learning datasets from the well known UCI machine learning repository(Blake & Merz 1998). The predictive performance of a trained MLPs and C4.5 induced Decision Trees were compared on number of datasets. Nine datasets from those which the MLP outperformed the C4.5 Decision Tree were randomly chosen to be used in this evaluation. The number of input features and number of classes for each dataset is given in table 1. The Balance scale dataset is an artificial dataset originally generated to model psychological experiments, all the others are real-world datasets originally collected in their respective fields and then donated to the UCI machine learning repository. The Hepatitis, Diabetes, Housing and Heart datasets consist of only numeric features. The Vote dataset consists of purely discrete data. The Labor and Colic datasets have a mixture of numeric and discrete features. The Housing dataset in its original form has a continuous output value, but for these experiments it has been

Dataset	num of features	num of classes
Balance-scale	4	3
Colic	8	2
Diabetes	24	2
Eucalyptus	19	5
Heart-statlog	13	2
Hepatitis	20	2
Housing	14	2
Labor	16	2
Vote	17	2

Table 1: Number of features and classes for datasets

DataSet	Neural(CE)	C4.5[2]	ExTree100
Balance-scale	89.60	77.92	78.60
Colic	82.61	81.52	81.79
Diabetes	75.91	72.53	76.04
Eucalyptus	62.09	60.60	57.20
Heart-statlog	83.33	71.11	78.15
Hepatitis	83.87	70.32	80.65
Housing	87.55	82.41	85.18
Labor	90.35	83.33	85.96
Vote	96.09	93.79	95.63
Mean	84.18	77.69	79.80

Table 2: Percentage of Instances Classified Correctly

transformed into a two class discrete problem of predicting whether the output value is above $20000.

To measure the performance of the algorithm, two standard statistical techniques were used: Stratified ten fold cross-validation(Stone 1974) was used to obtain a reliable measure of the predictive accuracy of the algorithm on the datasets and a Wilcoxon(Wilcoxon 1945) rank sign test was used to test whether the difference in accuracy between ExTree and C4.5 was statistically significant. In all the experiments the same ANN topology was used: a two-layer MLP, with five hidden nodes. All training was done using gradient descent with momentum to minimize a cross entropy (Van Ooyen & Nienhuis 1992) error function. The hidden nodes used the bipolar activation function. The nodes in the output layer used the softmax activation function[1]. Learning rate and momentum were set at 0.01 and 0.9 respectively. Performance of the ANNs could possibly be improved by optimizing the learning rate, momentum and architecture of each ANN to each of the individual datasets but because the purpose of this paper is to illustrate the validity of the ExTree approach to rule extraction this has not been done. To foster generalization, 33% of the training set was set aside to be used as an early stopping validation set. The input features were normalized to have a mean of 0, and a standard deviation of 1 for the ANN as is normal for MLP training(Demuth & Beale 2002; Haykin 1999). For purposes of comparison, predictive classification accuracy results were obtained for these datasets using an implementation of Quinlan's C4.5 algorithm[2]. The C4.5 implementation used the same validation set based pruning technique as ExTree to ensure that any differences in predictive accuracy were not due to the pruning techique used.

Table 2 shows the results obtained using 10-fold cross-validation. As expected the results confirm that ANNs do outperform C4.5. C4.5 does not make maximum use of the information present in the datasets. ExTree performed slightly better on average than C4.5 did. ExTree produced more accurate models on 8 of the 9 datasets. A Wilcoxon rank sum test showed that the difference between the C4.5 and Extree was significant $(p < 0.01)$. ExTree appeared to do particularly well on numerically dominated datasets with the largest improvement over C4.5 made on the Heart and Hepatitis datasets which consist of purely numeric features. A likely explanation for this improvement is that if the region of $\mathcal{X}$ where the optimal splitting point lies is sparsely represented in the dataset then C4.5 will be unlikely to find it, whereas ExTree will have sampled extra points in the region and so will be able to produce a more accurate estimate of the optimal splitting point. There are still large differences between many of results obtained by the ANN and ExTree which suggests that there is still much knowledge to be extracted. The ANN outperformed both ExTree and C4.5 by around 10% on the Balance scale dataset. This is almost certainly due to Decision Trees not being able to represent the mapping required by the balance scale dataset [3] This indicates that there will be ANNs that ExTree will be unable to extract sufficiently comprehensible rules from because Decision Trees are simply not powerful enough to represent the function that the ANN has learnt.

Conclusion

A method for extracting Decision Trees from trained Artificial Neural Networks regardless of the ANNs architecture and independent of its learning algorithm has been presented. It was found that the trees produced had better predictive accuracy than trees produced using the C4.5 based learning algorithm for eight of the nine datasets. The results obtained using the ExTree algorithm indicate that querying and sampling the ANN to induce a C4.5 like decision tree is a workable approach for a wide range of problem domains. The results showed that there were still large differences between the predictive accuracy of the underlying ANN and ExTree on some datasets. This suggests there is further knowledge to be extracted from the ANN. An obvious next step to achieving this would be to modify the number of new instances generated at the nodes (currently 100).

[1] The combination of softmax activation functions and a cross entropy error function has the advantage of allowing a probabilistic interpretation of the ANNs output(Ripley 1996)(Bishop 1995).

[2] It should be noted that this was not the 'official' C4.5 released by Quinlan but a C++ work-a-like implementation which shares much of the codebase of the ExTree implementation to ensure a fair comparison.

[3] Because balance scale is an artificial dataset the concept function is actually known: (Feature1 × Feature2) is equal, greater than or less than (Feature2 × Feature3)?

Preliminary experiments using an increased number of generated instances on a subset of the datasets used in this paper have indicated an improvement in predictive accuracy.

The results report in the last section used ANNs had not been optimized for the individual datasets. Optimizing the ANN topology would most likely increase the accuracy of the ANN which would in turn increase the accuracy of the Decision Tree extracted by ExTree.

References

Andrews, R.; Diederich, J.; and Tickle, A. B. 1995. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-Based Systems* 8(6):373–389.

Bishop, C. M. 1995. *Neural Networks for Pattern Recognition*. Oxford: Oxford University Press.

Blake, C., and Merz, C. 1998. UCI repository of machine learning databases. *University of California, Irvine, Dept. of Information and Computer Sciences*.

Craven, M. W., and Shavlik, J. W. 1994a. Using sampling and queries to extract rules from trained neural networks. In *Eleventh International Conference of Machine Learning*. San Francisco: Morgan Kaufmann.

Craven, M., and Shavlik, J. W. 1994b. Using sampling and queries to extract rules from trained neural networks. In *International Conference on Machine Learning*, 37–45.

Craven, M., and Shavlik, J. 1995. Extracting tree-structured representations of trained networks. In Touretzky; S., D.; Mozer; C., M.; Hasselmo; and E., M., eds., *Advances in Neural Information Processing Systems 8. Proceedings of the 1995 Conference.*, 24–30.

Demuth, H., and Beale, M. 2002. *Neural Network Toolbox*. Mathsoft.

Fu, L. 1995. Rule learning by searching on adapted nets. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, 373–389.

Haykin, S. 1999. *Neural networks : a comprehensive foundation*. Upper Saddle River, N.J.: Prentice Hall.

Hyafil, L., and Rivest, R. 1976. Constructing optimal binary decision trees is NP-complete. *Information Processing Letters* 5(1):15–17.

Michie, D.; Spiegelhalter, D.; and Taylor, C. 1994. *Machine learning, neural and statistical classification*. Ellis Horwood series in artificial intelligence. New York: Ellis Horwood.

Murthy, S. K. 1995. *On Growing Better Decision Trees from Data*. Ph.D. Dissertation, Johns Hopkins University.

Quinlan, J. R. 1986. Induction of decision trees. *Machine Learning* 1(1):81–106.

Quinlan, J. 1999. Simplifying decision trees. *International Journal of Human-Computer Studies* 51(2):497–510.

Ripley, B. D. 1996. *Pattern recognition and neural networks*. Cambridge ; New York: Cambridge University Press.

Rothwell, J. 2002. *Artificial Neural Networks For Psychological Profiling Using Multichannels Of Nonverbal Behaviour*. Ph.D. Dissertation, Manchester Metropolitan University.

Rumelhart, D.; Hinton, G.; and Williams, R. 1986. Learning internal representations by error propagation. In Rumelhart, D., and McCleland, J., eds., *Parallel Distributed Processing*, volume 1. Cambridge: MIT Press.

Stone, M. 1974. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society* 36:111–147.

Van Ooyen, A., and Nienhuis, B. 1992. Improving the convergence of the back-propagation algorithm. *Neural Networks* 5(3):465–71.

Wilcoxon, F. 1945. Individual comparisons by ranking methods. *Biometrics* 1:80–83.