Learning-based Rule-Extraction from Support Vector Machines

Nahla Barakat[1]
n.barakat@soharuni.edu.om

Joachim Diederich [1,2]
j.diederich@soharuni.edu.om

Faculty of Applied Sciences [1]
Natural Language Processing and Machine Learning Group
Sohar University
Sohar, PC311, Oman

School of Information Technology and Electrical Engineering [2]
The University of Queensland
Brisbane Q 4072, Australia

Abstract

In recent years, support vector machines (SVMs) have shown good performance in a number of application areas, including text classification. However, the success of SVMs comes at a cost – an inability to explain the process by which a learning result was reached and why a decision is being made. Rule-extraction from SVMs is important for the acceptance of this machine learning technology, especially for applications such as medical diagnosis. It is crucial for the users to understand how the system makes a decision. In this paper, a novel approach for rule-extraction from support vector machines is presented. This approach handles rule-extraction as a learning task, which proceeds in two steps. The first is to use the labeled patterns from a data set to train an SVM. The second step is to use the generated model to predict the label (class) for an extended data set or different, unlabeled data set. The resulting patterns are then used to train a decision tree learning system and to extract the corresponding rule sets. The output rule sets are verified against available knowledge for the domain problem (e.g. a medical expert), and other classification techniques, to assure correctness and validity of rules.

Key Words: SVM – Learning-based Rule Extraction algorithms – Medical diagnosis

1. Introduction

In recent years, support vector machines (SVMs) have shown good performance in a number of application areas [1]. However the learning capability of SVMs comes at a cost: an inherent inability to explain the process by which a learning result was reached. Hence, the situation is similar to neural networks [2], where the apparent lack of an explanation capability has led to various approaches aiming at extracting symbolic rules from neural networks. For SVMs to gain acceptance in areas such as medical diagnosis, it is desirable to offer an "explanation" capability.

Within the general area of rule-extraction from neural networks, two basic approaches are presented by authors such as Andrews et al. [3]: decompositional and learning-based. The distinguishing characteristic of the decompositional approach is that the focus is on extracting rules at the level of individual components of the underlying machine learning method. In neural networks, these are hidden and output units. The classification “learning-based” is given to those rule-extraction techniques that treat the underlying classifier as a "black box". Such techniques typically are used in conjunction with a learning algorithm with inherent explanation capability and the basic idea is to use the first classifier to generate examples for a second learning algorithm that generates rules as output.

In case of SVMs the situation is similar: Decompositional approaches can be based on the analysis of support vectors generated by the SVM while learning-based approaches learn what the SVM has learned. An example for learning-based rule-extraction from SVMs is Mitsdorffer [4].

The first part of this paper highlights the importance of rule-extraction algorithms. The second part reviews some learning-based rule-extraction algorithms from ANN, followed by some experiments with rule-extraction from SVMs. The remainder of the paper focuses on rule-extraction from SVMs and medical diagnosis, the validation of this approach, followed by a conclusion.

2. The Importance of Rule-Extraction Algorithms

2.1. Provision of an explanation component

The ability of symbolic AI systems to provide declarative representation of knowledge about the problem domain offers natural explanation for the decisions made by the system. Davis et al [5] argue that even limited explanation can positively influence system acceptance by the user. An explanation capability can also provide a check on the internal logic of the system as well as being able to give a novice insight into the problem [6]. ANN's & SVMs have no such declarative knowledge structures, and hence, are limited in providing an explanation component.

2.2. Overcome the problem of knowledge acquisition for symbolic AI systems

Constructing and debugging knowledge bases is the most difficult and time consuming task in building an expert system [7]. One of many motivations for introducing machine learning algorithms over the past decade was to overcome this 'knowledge acquisition' problem [8] [9]. Rule-extraction algorithms allow a trained ANN or SVM to be used as the basis for the construction of a knowledge/rule bases. Further, rule-extraction as part of a rule refinement method allows for the improvement of symbolic knowledge/rule bases that are incomplete or only partially correct.

2.3. Data exploration and the induction of scientific theories

Over time AI systems like ANNs and SVMs have proven to be extremely powerful tools for discovering previously unknown dependencies and relationships in data sets. As Craven & Shavlik [10] observe, *`a (learning) system may discover salient features in the input data whose importance was not previously recognised.'* The explanations given by rule-extraction algorithms significantly enhance the capabilities of AI systems to explore data and support the induction and generation of new theories.

2.4. Improving the generalisation of AI solutions

In spite of the good generalization ability of SVMs, in some cases, when a limited or unrepresentative data set is used in the training process, the generalisation may fail, even with evaluation methods such as cross-validation. A rule-extraction process is essential to anticipate or predict a set of circumstances under which generalisation failure can occur. Alternatively the user may be able to use the extracted rules to identify regions in input space which are not represented sufficiently in the existing training set data and supplement the data set accordingly.

3. Related Work:

3.1. Learning-based rule-extraction algorithms from ANN

3.1.1. Conjunctive rule-extraction algorithm: Carven & Shavlik's [10] approach involves viewing rule-extraction as a learning task where the target concept is the function computed by the network and the input features are simply the network's input features. The approach uses two different oracles that are able to answer queries about the concept being learned. The Examples oracle produces, on demand, training examples for the rule-learning algorithm. The Subset oracle takes two arguments: a class label and a conjunctive rule. Subset returns true if all of the instances that are covered by the rule are members of the given class, and false otherwise.

The algorithm for extracting conjunctive rules from trained neural networks is outlined in Table 1. The algorithm maintains a DNF expression for each class, repeatedly queries Examples, and then determines the class of each returned example. If an example is not covered by the current DNF expression for the class then it serves as the

basis for a new rule (i.e. a new term in the DNF expression). The new rule is initialized as the conjunction of all of the feature values of the returned example. This rule is then generalized by repeatedly dropping an antecedent (i.e. making one of the features undetermined) and then calling Subset to ascertain if the rule still agrees with the network. If Subset returns true, then the dropped antecedent is left off of the rule (i.e. the feature is left undetermined), otherwise it is put back on the rule) [2] [3].

Table 1: Rule-extraction by Learning

```
/* initialize rules for each class */
for each class c
        R_c : = 0
repeat
        e := Examples ( )
        c := classify(e)
        if e not covered by R_c then
                /* learn a new rule */
                r := conjunctive rule formed from e
                for each antecedent r_i of r
                        r':= r but with r_i dropped
                        if Subset (c; r') = true then r := r'
                R_c := R_c ∨ r
until stopping criterion met
```

3.1.2. The TREPAN Algorithm for Extracting Decision Trees: TREPAN algorithm by Craven & Shavlik [11], handles the task of extracting a comprehensible concept description from a trained network as an inductive learning problem. In this learning task, the target concept is the function represented by the network and the concept description produced by their learning algorithm is a decision tree that approximates the network. Since the target function is simply the concept represented by the network, the oracle uses the network to answer queries. TREPAN learns directly from training set (similar to CART and C4.5) but it is substantially different from those algorithms in terms of the oracle, tree expansion, split types, split selection and stopping criteria. The algorithm is outlined in Table 2:

Table 2: The TREPAN algorithm

```
TREPAN ( training – examples, features)
        Queue := 0                                          /* sorted queue of nodes to expand*/
        for each example E ε training – examples            /* use net to label examples*/
                class label for E := ORACLE (E)
        initialize the root of the tree T, as a leaf node
        put ⟨T , training – examples , { } ⟩ into Queue
        while Queue is not empty and size (T ) < tree – size-limit      /* expand a node*/
                remove node N from head of Queue
                examples_N : = example set stored with N
                constraints_N : = constraint set stored with N
                use features to build set of candidate splits
                use examples_N and calls to ORACLE ( constraints_N) to evaluate splits
                S: = best binary split
                Search for best m-of-n split, S′, using S as a seed
                make N an internal node with split S′
                for each outcome, s, of S′                  /* make children nodes*/
                        make C , a new child node of N
                        constraints_C : = constraints_N ∪ { S′ = s }
                        use calls to ORACLE (constraints_C ) to determine if C should remain a leaf
                        otherwise
                                examples_C : = members of examples_N with outcome s on split S′
                                put ⟨C , examples_C , constraints_C ⟩ into Queue
        return T
```

Craven & Shavlik [11] claim that the TREPAN algorithm produces comprehensible descriptions by extracting decision trees that accurately describe the network's concept representation.

3.2. Rule-Extraction from SVMs

Núñez et al [12] introduce an approach for rule-extraction from SVMs: The SVM+ prototype method. The basic idea of this method is to use the output decision function from an SVM and then use K-means clustering to determine prototype vectors for each class. These vectors are combined with support vectors to define an ellipsoid in the input space which are then mapped to if-then rules. This approach does not scale well: in case of a large number of patterns and an overlap between different attributes, the explanation capability suffers

4. Learning-based Rule-Extraction from SVMs: The Problem Domain

The problem here is the prediction of the onset of diabetes mellitus within 5 years by use of 8 variables present in the Pima Indian diabetic database[1]. Diabetes mellitus is a chronic disease which is associated with increased concentration of glucose in the blood which in turn damages many of the body's blood vessels and nerves. People with IGT[2] are at a substantially higher risk of developing diabetes than those with normal glucose tolerance. The other risk factors include obesity, family history of diabetes[3] etc.

4.1 The Problem

The Pima Indian diabetic database is a benchmark data set that has been used by many researchers to test the accuracy of different classification algorithms. The limitation of many of those algorithms is their inability to offer clear explanations of the concepts learned. For a medical application, it is crucial to understand the systems' decision in order to be confident that future diagnoses (predictions) are correct. In this section, we introduce a novel approach for rule-extraction from SVMs applied to a medical domain.

4.2 The Approach

A learning approach is adopted to extract rules from SVMs using two different data sets:

- A modified "Pima Indian diabetic" data set is used for SVM learning purposes, i.e. to build a model with acceptable accuracy. This is data set A.
- A second data set is generated with the same attributes but modified values to explore the generalisation behaviour of the SVM. That is, the SVM is used to get the class labels for this data set. Data set B is obtained.
- Data sets B & C (C is obtained by combining data sets A & B) are used to train various machine learning techniques with explanation capability. Thereby, rules are generated that represent the generalisation behaviour of the SVM.

4.3 The Experiment

The original "Pima Indian diabetic" data set has 768 patterns (500 non-diabetic and 268 diabetic); all of them are females of Pima Indian heritage with an age range 21-81 years. Each pattern has 9 attributes; risk factors are described in terms of 8 attributes plus a binary output class (tested positive or negative for diabetes).

The risk factors are:

Inputs:

1- Number of times pregnant
2- 2-hour OGTT plasma glucose
3- Diastolic blood pressure
4- Triceps skin fold thickness
5- 2 hour serum insulin
6- Body Mass Index (BMI)
7- Diabetes pedigree function
8- Age

All attributes are numeric,

Output:

9- Diagnosis (Diabetes onset within 5 years (0,1))

---

[1] Available at the UCI Machine Learning Repository

[2] IGT refer to levels of blood glucose concentration above normal range (OGTT 140-199), but below those which are diagnostic for diabetes(OGTT >=200)

[3] Diabetes Pedigree function

This is a database with no missing values; however, it includes many patterns that have zero values for Diastolic blood pressure, OGTT, BMI, Triceps skin fold thickness, and 2 Hour serum insulin, which are clinically insignificant. This in addition to some noise within the data set.

4.3.1 Training Examples: We selected a random sample of 496 patterns from the original data set, after removing all patterns with zero value for the attributes 2-4 plus patterns which are highly suspected to be noisy based on medical expertise. This leaves a data set A which has a considerable number of patterns with zero value for attribute 5. In total, data set A has 320 negative patterns (non-diabetic) and 176 positive patterns (diabetic) which preservers the distribution of original data set.

4.3.2 SVM Training: Data set A is used for SVM training. A variety of learning parameters have been tested, to reach an SVM model with an acceptable degree of accuracy, precision and recall. A linear SVM is sufficient and the results are shown in Table 3.

Table 3: Learning results for various machine learning techniques (Data set A).

	SVM			**Decision Tree (C5)**			**Best ANN Architecture (8-1-1) T.A Quasi - Newton**		
	Accuracy %	Precision %	Recall %	Accuracy %	Precision %	Recall %	Accuracy %	Precision %	Recall %
Training	83	87	60	89	82	88	83	82	68
Leave-one-out cross-validation	82	87	59	78	65	77	70		

4.3.3 SVM Generalisation: Generating data set B: The SVM model is used to predict the class for an additional 100 patterns with a different distribution of positive and negative examples compared to the training set[4]. The eight risk factors are used as input to the SVM which in turn assigns a target class to each pattern. The result of this step is data set B. Data set B is used to extract the rules learned by the SVM.

4.4 Results

4.4.1 Decision Tree Learning: The C5 algorithm is used to generate decision trees and rule sets from data sets A & B. To test the quality of extracted rules, 495- and 99-fold cross-validation is used for training purposes.

Comparing the decision trees indirectly generated from the SVM classifier (data set B) with the tree generated from the training data, it was noted that both trees show the most significant risk factors which are plasma glucose (OGTT), BMI, and pedigree function. It was also noted that the tree for data set A is deeper that for data set B. This difference can be attributed to the larger data set A.

Comparing the rule sets generated from the above mentioned trees with coverage 10% and confidence 0.6 (Table 4: Appendix 1), it can be demonstrated that the total number of rules is the same for both data sets. The difference in the number of rules is partially due to the difference in the distribution of positive and negative classes in data sets A & B. The same explanation is valid for the comparison of the results from data sets A & C (Table 5: Appendix 1).

4.4.2 Classification and Regression Tree: Comparing the rule sets generated by the Classification and Regression Trees for data sets A & B and using the same parameters as for C5 (10% coverage, and 0.6 confidence), it can be noticed that the number of rules for positive examples are the same for both data sets, while data set A has more rules for negative examples (Table 6: Appendix 1). Again this can be attributed to the difference in distribution of positive and negative examples between the data sets. The rules extracted by this technique for data set B are more relevant from the clinical point of view and are more consistent with the results obtained from the other classification techniques.

[4] This data set includes patterns excluded from the original data set (suspected to be noisy) plus random patterns and examples with a zero value for attribute 5. These zero values were replaced by the average, median or mode value for the attribute. Based on risk factors & clinical knowledge, data set B has 40 negative examples and 60 positive examples.

5. Conclusion

The decision trees and rule sets produced by C5 offer an explanation of the concepts learned by the SVM. The extracted rules are correct and valid from the medical point of view and consistent with clinical knowledge of diabetes risk factors. The results are also consistent with other techniques such as Classification and Regression Trees. In summary, this approach is valid for clinical applications.

References

[1] T. Joachims, Transductive Inference for Text Classification using Support Vector Machines, *International Conference on Machine Learning (ICML),* 1999.

[2] A.B. Tickle, R. Andrews, M. Golea, J. Diederich, The truth will come to light: directions and challenges in extracting the knowledge embedded within trained artificial neural network, *IEEE Transactions on Neural Networks 9,* 6 ,1998, PP.1057-1068.

[3] R. Andrews, J. Diederich, A.B. Tickle, A Survey and Critique of Techniques For Extracting Rules From Trained Artificial Neural Networks, *Knowledge Based Systems*, 8, 1995, pp 373-389.

[4] R. Mitsdorffer, J. Diederich, C. Tan, Rule-extraction from Technology IPOs in the US Stock Market, *ICONIP02*, Singapore, 2002.

[5] R. Davis, B.G.Buchanan, E. Shortcliff, Production Rules as a Representation for a Knowledge Based Consultation Program, *Artificial Intelligence* 8(1), 1977, pp15-45.

[6] S. Gallant, Connectionist Expert Systems, *Communications of the ACM*, 31(2), 1988, pp 152-169.

[7] S. Sestito, T. Dillon, *Automated Knowledge Acquisition*, Australia, Prentice Hall, 1994.

[8] K. Saito, R. Nakano, Medical Diagnostic Expert System Based on PDP Model, *IEEE International Conference on Neural Networks*, San Diego, 1, 1988, pp 255-262.

[9] S. Sestito, T. Dillon, Automated Knowledge Acquisition of Rules With Continuously Valued Attributes, *12th International Conference on Expert Systems and their Applications (AVIGNON'92),* France, 1992, pp. 645-656.

[10] M.W. Craven, J.W. Shavlik, Using Sampling and Queries to Extract Rules From Trained Neural Networks, , *the Eleventh International Conference of Machine Learning*, CA, 1994.

[11] M.W. Craven, J.W. Shavlik, Extracting Tree – Structured Representation of Trained Networks, *Advances in Neural Information Processing Systems*, Cambridge, MIT Press , 8, 1996.

[12] H. Núñez, C. Angulo, A. Catala, Rule-extraction from Support Vector Machines, *The European Symposium on Artificial Neural Networks*, Burges, ISBN 2- 930307-02-1, 2002, pp.107-112.

Appendix 1

C5	Data Set A (Training Data : 496 patterns)	Data Set B (SVM Data : 100 Patterns)
Rule Set Coverage 10% Confidence 0.6	**Rules for 0:** Rule #1 for 0: if Plasmaglo =< 141 and BMI =< 26.3 then -> 0 (87, 0.989) Rule #2 for 0: if Plasmaglo =< 127 and BMI > 26.3 and Age =< 28 then -> 0 (155, 0.903) **Rules for 1:** Rule #1 for 1: if Plasmaglo > 154 then -> 1 (68, 1.0) Default : -> 0	**Rules for 0:** Rule #1 for 0: if Plasmaglu =< 144 and BMI =< 35.9 and # times Prig > 1 then -> 0 (30, 0.933) **Rules for 1***:* Rule #1 for 1: if Plasmaglu =< 144 and BMI > 35.9 then -> 1 (20, 0.8) Rule #2 for 1: if Plasmaglu > 144 then -> 1 (38, 0.947) Default : -> 1

Table 4: C5 Rule sets for Data sets A & B

C5	Data Set B (SVM Data: 100 Patterns)	Data Set C (Training Data + SVM Data: 596 patterns)
Rule Set Coverage 10% Confidence 0.6	Rule Sets **Rules for 0:** Rule #1 for 0: if Plasmaglu =< 144 and BMI =< 35.9 and # times Prig > 1 then -> 0 (30, 0.933) **Rules for 1***:* Rule #1 for 1: if Plasmaglu =< 144 and BMI > 35.9 then -> 1 (20, 0.8) Rule #2 for 1: if Plasmaglu > 144 then -> 1 (38, 0.947) Default : -> 1	Rule Sets **Rules for 0:** Rule #1 for 0: if Plasmaglo =< 127 and Age =< 28 then -> 0 (221, 0.923) Rule #2 for 0: if Plasmaglo > 83 and Plasmaglo =< 127 and Age > 28 and BMI > 26.3 and BMI =< 44.5 and pedigreefun =< 0.561 then -> 0 (70, 0.757) **Rules for 1:** Rule #1 for 1: if Plasmaglo > 157 then -> 1 (81, 1.0) Default : -> 0

Table 5: C5 Rule sets for Data sets B & C

C & R Tree	Data Set A (Training Data: 496 patterns)	Data Set B (SVM Data: 100 Patterns)
Depth 4 Rule set Coverage 10% Confidence 0.6	**Rule Sets** **Rules for 0:** Rule #1 for 0: if Plasmaglo < 127.5 and Age < 28.5 and Tricepsskin < 29.5 then -> 0 (139, 0.978) Rule #2 for 0: if Plasmaglo < 127.5 and Age < 28.5 and Tricepsskin >= 29.5 then -> 0 (76, 0.829) Rule #3 for 0: if Plasmaglo < 108.5 and Age >= 28.5 and 2hourserum < 143 then -> 0 (64, 0.844) **Rules for 1:** Rule #1 for 1: if Plasmaglo >= 108.5 and Plasmaglo < 141.5 and Age >= 28.5 and BMI >= 26.8 then -> 1 (67, 0.642) Rule #2 for 1: if Plasmaglo >= 154.5 then -> 1 (68, 1.0) Default : -> 0	**Rule Sets** **Rules for 0:** Rule #1 for 0: if Plasmaglo < 144.5 and BMI < 36 and Age < 57 and pedigreefun < 0.884 then -> 0 (30, 1.0) Rule #2 for 0: if Plasmaglo < 144.5 and BMI < 36 and Age < 57 and pedigreefun >= 0.884 then -> 0 (10, 0.6) **Rules for 1:** Rule #1 for 1: if Plasmaglo >= 106.5 and Plasmaglo < 144.5 and BMI >= 36 and Tricepsskin < 45.5 then -> 1 (12, 1.0) Rule #2 for 1: if Plasmaglo >= 144.5 and pedigreefun >= 0.2 then -> 1 (35, 1.0) Default : -> 1

Table 6: C & R Tree Rule sets for Data sets A & B