Neural Data Analysis: Ensemble Neural Network Rule Extraction Approach and Its Theoretical and Historical Backgrounds

(Invited Paper)

Yoichi Hayashi

Department of Computer Science,
Meiji University,
Tama-ku, Kawasaki 214-8571, Japan
hayashiy@cs.meiji.ac.jp

Abstract. In this paper, we first survey the theoretical and historical backgrounds related to ensemble neural network rule extraction. Then we propose a new rule extraction method for ensemble neural networks. We also demonstrate that the use of ensemble neural networks produces higher recognition accuracy than do individual neural networks. Because the extracted rules are more comprehensible. The rule extraction method we use is the Ensemble-Recursive-Rule eX traction (E-Re-RX) algorithm. The E-Re-RX algorithm is an effective rule extraction algorithm for dealing with data sets that mix discrete and continuous attributes. In this algorithm, primary rules are generated, followed by secondary rules to handle only those instances that do not satisfy the primary rules, and then these rules are integrated. We show that this reduces the complexity of using multiple neural networks. This method achieves extremely high recognition rates, even with multiclass problems.

Keywords: Ensemble, neural network, rule extraction, Re-RX algorithm, Ensemble method, Recursive neural network rule extraction.

1 Introduction

In this paper, we first survey the nature of artificial neural networks (ANNs), the origin of neural network rule extraction, incorporation of fuzziness in neural network rule extraction, the theoretical foundation of neural network rule extraction, the computational complexity of neural network rule extraction, neuro-fuzzy hybridization, rule extraction from neural network ensembles, and the background of neural network ensembles. Then we describe the three objectives of this paper.

We propose the Ensemble-Recursive-Rule Extraction (E-Re-RX) algorithm, which extracts comprehensible rules [13], [37]. In the E-Re-RX algorithm, the Re-RX algorithm [9] is an effective rule extraction algorithm for data sets that comprise both discrete and continuous attributes. The extracted rules maintain the high recognition capabilities of a neural network while expressing highly comprehensible rules.

1.1 Nature of Artificial Neural Networks

ANNs attempt to replicate the *computational* power of biological neural networks and thereby endow machines with some of the *cognitive abilities* possessed by biological organisms. However, an impediment to more widespread acceptance of ANNs is the absence of the capability to explain to the user, in a human-comprehensible form, how the network arrives at a particular decision. That is, it is very difficult to discuss the *knowledge* encoded within the "*black box*" of NNs. Recently, widespread activities have attempted to revisit this situation by extracting the embedded knowledge in trained ANNs in the form of symbolic rules [14].

As a biologically inspired analytical technique, NNs have the capacity to learn and model complex nonlinear relationships. Theoretically, multi-layered feedforward NNs are universal approximators and, as such, have an excellent ability to approximate any nonlinear mapping to any degree of accuracy [6], [40]. They do not require *a priori* models to be assumed or *a priori* assumptions to be made on the properties of data [8].

Generally, ANNs consider a fixed topology of neurons connected by links in a predefined manner. These connection weights are usually initialized by small random values. *Knowledge-based networks* constitute a special class of ANNs that consider crude domain knowledge to generate the initial network architecture, which is later refined in the presence of training data. [14]

Recently, some attempts have been made to improve the efficiency of neural computation by using knowledge-based nets. Such nets help to reduce the searching space and time while the network traces the optimal solution. In such a situation, one can extract causal factors and functional dependencies from the data domain for initially encoding the ANN and later extracting refined rules from the trained network [50].

The term *rule generation* encompasses both rule extraction and rule refinement. Note that *rule extraction* here refers to extracting knowledge from the ANN while using network parameters in the process. *Rule refinement*, in contrast, pertains to extracting refined knowledge from the ANN that was initialized by using crude domain knowledge. Rules learned and interpolated for fuzzy reasoning and fuzzy control can also be considered under rule generation. In a wider sense, rule generation includes the extraction of domain knowledge (e.g., for the initial encoding of an ANN) by using nonconnectionist tools such as fuzzy sets and rough sets. [14]

1.2 Origin of Neural Network Rule Extraction [14]

Here we first consider the layered connectionist model by Gallant [15] and Saito and Nakano [16] for rule extraction in the medical domain. The inputs and outputs consist of crisp variables in all cases. Generally, the symptoms are represented by the input nodes, and the diseases and possible treatments correspond to intermediate and/or output nodes. The multilayer network described by Saito and Nakano was applied to the detection of *headache*. Headache patients respond to a questionnaire regarding the perceived symptoms and these constitute the input to the network.

In 1988, the model by Gallant [15], dealing with *sacrophagal* problems, uses a linear discriminant network (with no hidden nodes) that is trained by the *simple pocket algorithm.*

Gallant's model [15] incorporates inferencing and forward chaining, confidence estimation, backward chaining, and explanation of conclusions by IF-THEN rules. To generate a rule, the attributes with greater inference strength (magnitude of connection weights) are selected and a conjunction of the more significant premises is formed to justify the output concept.

Rule extraction techniques generally fall into two categories [44]: direct approaches and indirect approaches. We take the standpoint that indirect approaches are more promising.

In an indirect approach, a predictive model is built from training data, and rules are extracted from the model. ANNs and Support Vector Machines (SVMs) are two of the most popular algorithms used to build predictive models. Rule extraction from ANNs has been investigated by many researchers [41], [42]. SVM-based rule extraction has also been explored extensively due to the high performance of SVMs [43].

1.3 Incorporating Fuzziness in Neural Network Rule Extraction

As an illustration of the characteristics of layered fuzzy neural networks for inferencing and rule generation, the models by Hayashi [17], [18] and Hudson et al. [19] are described first. A *distributed single-layer perceptron-based* model trained with the *pocket algorithm* was used by Hayashi [17], [18] for diagnosing *hepatobiliary* disorders. All contradictory training data were excluded, as these cannot be handled by the model. The input layer consists of fuzzy and crisp cell groups while the output is modeled only by fuzzy cell groups. The crisp cell groups are represented by *m* cells taking on two values, {(+1, +1, ,…, +1), (-1, -1,…, -1)}. Fuzzy cell groups, however, use binary *m*-dimensional vectors, each taking values of {+1, -1}. Linguistic relative importance terms such as *very important* and *moderately important* are allowed in each proposition. Linguistic truth values such as *completely true*, *true*, *possibly true*, *unknown*, *possibly false*, *false*, and *completely false* are also assigned by the domain experts, depending on the output values. By using different linguistic truth values, a pattern belonging to more than one class can be modeled. Extraction of fuzzy IF-THEN production rules is possible by using a top-down traversal involving analysis of the node activation, bias, and the associated link weights.

1.4 Theoretical Foundation of Neural Network Rule Extraction

A fuzzy system adaptively infers and modifies its fuzzy association from representative numerical samples. Neural networks, in contrast, can *blindly* generate and refine fuzzy rules from training data. Fuzzy sets are considered to be advantageous in the logical field and in easily handling higher order processing.

Higher flexibility is a characteristic feature of neural networks produced by learning and, hence, NNs are better suited for data-driven processing [20].

In 1993, Buckley, Hayashi and Czogala [21] mathematically proved the equivalence of neural nets and fuzzy expert systems. In other words, they proved that we can describe the contents of trained neural networks by a set of linguistic IF-THEN rules. Moreover, this paper firmly established the theoretical foundation of neural network rule extraction.

Hayashi and Buckley [22] proved that 1) any rule-based fuzzy system may be approximated by a neural net, and 2) any neural net (e.g., feedforward net, multilayered net) may be approximated by a rule-based fuzzy system. This kind of equivalence between the fuzzy-rule-based system and neural networks was also studied [21],[22],[23],[24].

1.5 Computational Complexity of Neural Network Rule Extraction

A salient theoretical discovery in this area is that, in many cases, the computational complexity of extracting rules from trained neural networks and the complexity of extracting rules directly from data are both NP-hard [25].

Bologna [32] claimed that the difficulty of extracting rules is related to the dimensionality of the input samples. More precisely, the dimensionality is related to *n* binary valued input neurons. We could find up to 2^n rules. Generally, with large-dimensional problems, several rules may be missed. In such a situation, the degree of matching between the rules and the neutral network classification, also denoted as *fidelity*, is less than 100%.

It is also worth mentioning that Roy [26] astutely disclosed the conflict between rule extraction and traditional connectionism. In detail, the idea of rule extraction from a neural network involves certain procedures, specifically the reading of parameters from a network. Such reading is not allowed by the traditional connectionist framework on which these neural networks are based. Roy [26] indicated that such a conflict could be resolved by introducing a control-theoretic paradigm, which was supported by new evidence from neuroscience about the role of neuromodulators and neurotransmitters in the brain.

1.6 Neuro-fuzzy Hybridization

Neuro-fuzzy hybridization [51] is done broadly in two ways: a neural network equipped with the capability of handling fuzzy information [termed *fuzzy-neural network* (FNN)], and a fuzzy system augmented by neural networks to enhance some of the neural network characteristics such as flexibility, speed, and adaptability [termed *neural-fuzzy system* (NFS)].

Other kinds of categorizations for neuro-fuzzy models have been reported in related literature [28]. Buckley and Hayashi [28] classified fuzzified neural networks possessing the following: 1) real number inputs, fuzzy outputs, and fuzzy weights; 2) fuzzy inputs, fuzzy outputs, and real number weights; 3) fuzzy inputs, fuzzy outputs, and fuzzy weights. Hayashi et al. [29] fuzzified the delta rule for the *multilayer perceptron* (MLP) by using fuzzy numbers at the input, output, and weight levels.

But, the algorithm with the stopping rule has problems. Ishibuchi et al. [30] incorporated triangular and trapezoidal fuzzy number weights, and thereby increased the complexity of the algorithm. Some of these problems were overcome by Feuring et al. in [31].

1.7 Rule Extraction from Neural Network Ensemble

In the beginning of the 1990s, Hansen and Salamon [38] showed that the generalization ability of learning systems based on artificial neural networks can be significantly improved through ensembles of artificial neural networks, i.e., training multiple artificial neural networks and combining their predictions via voting. Since combining works remarkably well, it became a very popular topic in both neural network and machine learning communities [35].

In general, a neural network ensemble is constructed in two steps: training a number of component neural networks, and then combining the component predictions [36].

The rationale for considering a combination of methods is similar to that of ensemble NNs [5]. However, ensembling is more robust and mitigates the effect of one method that gives bad results and ruins the performance [52].

Although many authors have generated comprehensible models from individual networks, much less work has been done in the explanation of neural network ensembles [32].

Bologna proposed the *Interpretable Multi-Layer Perceptron* (IMLP) and the *Discretized IMLP* (DIMLP) models with generated rules from neural network ensembles [33, 34]. The DIMLP is a special neural network model for which symbolic rules are generated to explain the knowledge embedded within the connections and the activation neurons. Bologna described how to translate symbolic rules into the DIMLP and how to extract rules from one or several combined neural networks.

Rules are generated from a DIMLP network by the induction of a special decision tree and taking into account virtual hyperplane frontiers.

In [32], Bologna's rule extraction was compared to other rule extraction techniques applied to neural networks. For seven out of eight classification problems, the accuracy of his results were equal to or better than those given by other techniques.

The scale of the computational complexity of his rule extraction algorithm in polynomial time is related to the dimensionality of the problem, the number of examples, and the size of the network. Continuous valued attributes do not need to be transformed to binary attributes, as is done in many rule extraction techniques. That is a clear advantage with respect to decompositional rule extraction algorithms with high exponential computational complexity (for comparison, see Section 2.2 of [32]). In practice, the execution time of learning and rule extraction is very reasonable. For more mathematical details on the computational complexity, refer to Section 3.3 of [32] and Section 6.4 of [34].

With Rule Extraction From Neural network Ensemble (REFNE) proposed by Zhou et al. [35], attributes are discretized during rule extraction, whereas Bologna's rule extraction algorithm performs the discretization during learning through the use of staircase activation functions. Furthermore, rules generated by REFNE are limited to three antecedents, whereas DIMLP does not impose any constraints. Another important difference is that we extract unordered rules from DIMLP ensembles, whereas ordered rules are generated by REFNE. Bologna's rule extraction algorithm has no parameters; hence, it could be easier for a non-specialist in rule extraction to use the DIMLP ensemble rather than those rule extraction techniques that require several parameters be set [32].

The REFNE approach proposed by Zhou et al. is designed to extract symbolic rules from trained neural network ensembles that perform classification tasks. REFNE utilizes trained ensembles to generate a number of instances and then extracts rules from those instances. REFNE can gracefully break the ties made by individual neural networks in prediction [35].

In recent years, many approaches for rule extraction from trained neural networks have been developed. A review of these approaches can be generally categorized into two classes: function-analysis-based and architecture-based approaches. The function-analysis-based approaches extract rules by regarding the trained networks as entities that can be easily modified for extracting rules from trained ensembles instead of the networks as entities. REFNE is derived from a function-analysis-based approach, called STARE [46], for extracting rules from trained neural networks [35].

The architecture-analysis-based approaches extract rules by disassembling the architectures of trained neural networks. Such architectures are those that are relatively difficult to modify for extracting rules from trained ensembles. The reason is that even if rules could be extracted from each individual network, it is still difficult to unite them into a consistent rule set that explains the capability of the ensemble because simply gathering them together can only result in a messy hodgepodge of rules [35].

However, no matter where the approaches for rule extraction from network ensembles are derived from, they must pay attention to some specific characteristics of ensembles, e.g., the ambiguity in a prediction caused by ties made by individual neural networks [35].

Zhou et al. [35] presented a pedagogical algorithm for extracting prepositional rules from a complicated neural network system. With different configurations, the algorithm can extract rules with high fidelity but moderate accuracy, or high accuracy but moderate fidelity. A particularly interesting issue that has not been addressed appears in [35]: which configuration should we prefer? This question places us in an uncomfortable situation: to sacrifice the fidelity, or to sacrifice the accuracy. In fact, pursuing high fidelity and high accuracy may not be possible in certain situations, although this has not been recognized before [27].

Zhou et al. [36] analyzed the relationship between the ensemble and its component neural networks from the context of both regression and classification. Their work revealed that it may be better to ensemble *many* instead of *all* the available neural networks at hand. This result is interesting because most approaches ensemble *all* the available neural networks for prediction. Then, to show the feasibility of their theory, an ensemble approach named Genetic Algorithm based Selective ENsemble (GASEN) was presented. A large empirical study showed that GASEN is superior to bagging [1] and boosting [2] in both regression and classification because it utilizes far fewer component neural networks but achieves stronger generalization ability [36].

In 2012, Hara and Hayashi proposed two ensemble neural network rule extraction algorithms. The former is for two-class classification [13]. The latter is for multiple-class classification [37]. Both of these algorithms use standard MLPs and the Re-RX algorithm proposed by Setiono [9]. Their recognition accuracy is very high.

For reference, Adeodato et al. [39] showed that the ensemble of MLPs produces better results than does the single MLP solution. For this purpose, the performance of the ensemble was compared to the average of the performances of each single MLP.

1.8 Neural Network Ensembles in This Paper

In ensemble neural networks, bagging [1], boosting [2], AdaBoosting [47], averaging [48], and other techniques have been suggested as ways to split a data set and perform multifaceted analyses. Each of these studies reported classification accuracy exceeding that of individual neural networks [3], [4], [10].

Because neural network ensembles are multiple individual neural networks, they present their own problems: their complexity is greater, rule extraction is more difficult, and they use more computing resources than are necessary. In the studied neural network ensembles to date, research on methods such as weighted voting [11] and averaging [48] for integrating the output has been conducted. Algorithms that use bagging or boosting in the C4.5 algorithm have been presented, but these do not directly address the problems, because splitting the data set into parts and applying the C4.5 algorithm to them generates rules that determine the class, and so the total output is classes applied broadly based on the rules. This in turn means that the rules are numerous and redundant. Consequently, we believe that methods such as bagging or boosting cannot be assumed to extract rules from an ensemble neural network.

Nevertheless, neural networks are known to be an effective method for real-world classification problems involving nonlinear data. Contrary to the standard explanation that neural networks operate as “black boxes,” many studies have been conducted on methods for rule extraction [49]. These studies can be seen as an outgrowth of the extraordinary advances that have been made in information technology (IT) and the ability of IT to easily handle massive volumes of data. Extracting rules from neural networks is not simply a matter of breaking open the black box. From the perspective of data mining, it increases the opportunities to use neural network technology as data mining technology.

In our proposed algorithm, we selected part of the learning data set , LD, and extracted a primary rule set from the learning data subset, LD’. Next, we classified our learning data subset into those instances that satisfied the primary rules and those that did not. Using the non-satisfying instances, we used the Re-RX [9] algorithm to extract a secondary rule set. The primary and secondary rule sets can be regarded as the ensemble neural network’s complete rules, which have sufficiently high recognition capabilities.

We conducted experiments using the CARD data set, the German Credit data set, the Thyroid data set, and the Page block data set, which can be obtained from the UCI repository [12]. In this paper, we provide a discussion based on the results of experiments and a conclusion that states the three open questions.

2 Purpose of This Paper

This paper has three major objectives.

The first objective is to increase recognition rates. Neural networks show high recognition accuracy when used as a data mining technique, but they do not show sufficiently high recognition accuracy when using a partial data set. In most cases, multiclass problems occur where poor recognition rates are seen. In this paper, we demonstrate extremely high recognition rates, even with multiclass problems. It is understood that one cause of low recognition rates in existing methods is overfitting. In our research, we used a randomly extracted learning data subset that was arbitrarily taken from the source data set. Using a partial learning data set is effective in reducing the number of factors in the source data set that lead to overfitting.

The second objective is to extract rules from an ensemble neural network. Ensemble neural networks are extremely effective in the field of data mining due to their strong recognition capabilities, but in fields where high reliability is demanded, such as medicine and finance, the recognition capability of ensemble neural networks alone is not enough. The solution is to extract rules that express what the ensemble neural network recognizes with a high degree of comprehensibility. If the recognition results can be expressed as rules, then the technique can even be used in fields that demand a high level of reliability. This would expand the range of applications for ensemble neural networks in data mining. In our research, we used the E-Re-RX algorithm [13], [37] that we are proposing for rule extraction. This algorithm is effective for rule extraction when the data set includes both discrete and continuous attributes, and the extracted rules show a high degree of comprehensibility.

The third objective is to minimize the use of computer resources. Ensemble neural networks have typically achieved good recognition accuracy by using a large number of neural networks, but using a large number of neural networks creates the problem of creating unused neural networks. In our research, we achieved high recognition accuracy by using an ensemble neural network consisting of the smallest number of neural networks possible, which is two.

3 Structure of the E-Re-RX Algorithm

3.1 Origin of Neural Network Ensemble

A neural network ensemble is a learning paradigm, in which a collection of a finite number of neural networks is trained for the same task. The neural network ensemble originates from Hansen and Salamon's work [38], which showed that the generalization ability of a neural network system can be significantly improved by ensembling a number of neural networks, i.e., by training many neural networks and then combining their predictions.

3.2 Re-RX Algorithm

The Re-RX algorithm [9] is designed to generate classification rules from data sets that have both discrete and continuous attributes. The algorithm is recursive in nature and generates hierarchical rules. The rule conditions for discrete attributes are disjointed from those for continuous attributes. The continuous attributes only appear in the conditions of the rules lowest in the hierarchy.

The outline of the algorithm is as follows.

```
Algorithm Re-RX(S, D, C)
Input: A set of data samples S having discrete attributes D and continuous attributes C.
Output: A set of classification rules.

1. Train and prune a neural network using the data set S and all its D and C attributes.
2. Let D' and C' be the sets of discrete and continuous attributes, respectively, still
   present in the network, and let S' be the set of data samples correctly classified by
   the pruned network.
3. If D' has associated continuous attributes C', generate a hyperplane to split the
   samples in S' according to the values of the continuous attributes C', then stop.
   Otherwise, by using only the discrete attributes D', generate the set of
   classification rules R for data set S'.
   For each rule Ri generated:
   If support(Ri)>$\delta_1$ and error(Ri)>$\delta_2$, then
   – Let Si be the set of data samples that satisfy the condition of rule Ri and let Di
     be the set of discrete attributes that do not appear in the rule condition of Ri.
   – If D' has associated continuous attributes C', generate a hyperplane to split the
     samples in Si according to the values of the continuous attributes Ci, then stop.
     Otherwise, call Re-RX (Si, Di, Ci).
```

In the above, we define the support of (Ri) to be the proportion of samples covered by rule Ri and the error of (Ri) to be the proportion of samples it incorrectly classifies.

3.3 Ensemble-Re-RX (E-Re-RX) Algorithm [13][37]

In the proposed E-Re-RX algorithm, we first produce the learning data set LD', which is necessary for training the first neural network. LD' is the set of instances extracted at random in an arbitrary proportion from the full learning data set LD. LD' is input into a neural network having one node in its hidden layer. The neural network is trained and pruned [7], and the rules are extracted.

In this paper, we restrict ourselves to back-propagation neural networks with one hidden layer because such networks have been shown to possess a universal approximation property [6], [40]. An effective neural network pruning algorithm is a crucial component of any neural network rule extraction algorithm. By removing the inputs not needed for solving the problem, the extracted rule set becomes more concise. In addition, the pruned network also filters noise that might be present in the data. Such noise could be data samples that are outliers or incorrectly labeled. From these extracted rules, we re-extract rules in accordance with the Re-RX algorithm, and take the final rule set as the primary rules R.

We divide LD into data sets that do and do not conform to these primary rules. The set of instances that do not conform are taken as LDf, which is input into the second neural network. The second neural network is similarly trained on LDf and pruned, rules are extracted, and rules are then re-extracted in accordance with the Re-RX algorithm. These extracted rules are the secondary rules Rf.

Integrated rules are obtained from extracted rules R and Rf. In the rule integration, we focus on the primary rule and the secondary rule for attributes and values that are the same. If the attributes and values for the primary rule and the secondary rule match exactly, and the class labels are also the same, the secondary rule is integrated into the primary rule. For example, assume the following rule is obtained as a primary rule.

R: If D42=0, then predict Class 1.

The following rule is obtained as a secondary rule.

Rf: If D42=0, then predict Class 1.

In this case, all of the attributes, values, and class labels that emerge in the rules match. Therefore, these rules are integrated. Another case may appear where one rule is expressed as either a primary or secondary rule, whereas another primary or secondary rule may have some matching attributes and values. In this case, whichever rule can be encompassed by the other is integrated into the other, regardless of class labels. For example, the following is obtained as a primary rule.

R: If D42=1 and D38=0 and D43=0 and D27=0 and D24=0 and D45=0 and D2=0 and D21=1, then predict Class 1.

The following is obtained as a secondary rule.

Rf: If D24=0 and D2=0 and D45=0 and D21=1, then predict Class 2.

In this case, all of the attributes and values of the secondary rule are encompassed within the primary rule. Because the secondary rule can be encompassed by the primary rule, even though the class labels differ, the secondary rule is integrated into the primary rule. The presence of attributes in the primary rule that are absent in the secondary rule are considered to result in more accurate class identification.

With the neural network ensemble, it is possible to determine the final output by integrating the outputs of the multiple neural networks. With the E-Re-RX algorithm, it is possible to determine the overall final output by integrating the rules. This rule integration enables the reduction of the number of neural networks and irrelevant rules. The essentials of the E-Re-RX algorithm are outlined as follows.

Input: Training sets LD’ and LDf

Output: Integrated rule set obtained by integration of the primary and secondary rule sets.

1. Extract at random an arbitrary proportion of instances from the learning set LD and designate this randomly extracted learning set LD’.
2. Perform training and pruning of LD’ with the first neural network.
3. Apply the Re-RX algorithm [9] to the output of Step 2 to obtain the primary rules.
4. Based on these primary rules, generate the set LDf consisting of instances that do not satisfy these rules.
5. Train and prune LDf with the second neural network.
6. Apply the Re-RX algorithm [9] to the output of Step 5 to output the secondary rules.
7. Integrate the primary and secondary rules.

The composition of the two standard MLPs is shown schematically in Fig. 1.

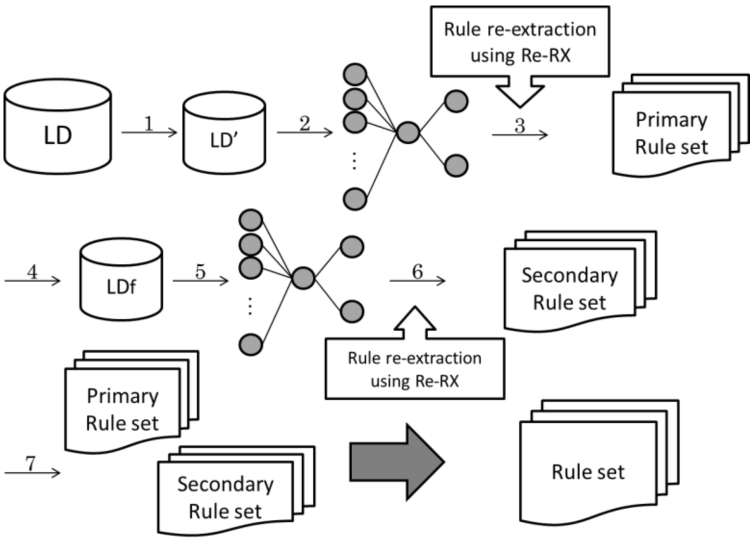

Fig. 1. Basic process for extraction of primary and secondary rules of high accuracy from two standard MLPs

3.4 Integration of Rules

Extracting rules from multiple standard MLPs is assumed to increase the number of rules, and some of these extracted rules may be redundant or irrelevant as classification rules.

The accuracy of the rules is maintained and their number is reduced by integrating the primary and secondary rules in accordance with the attributes.

In this paper, the rules extracted from the Re-RX algorithm use the decision tree formed by the J4.8 algorithm. By using the J4.8 algorithm for multiple generated rules and particularly for primary and secondary rules, the multiple attributes of the same type and value are integrated into rules having more attribute types; that is, the primary and the secondary rules are integrated into finer rules. In judgments made with a decision tree, rules having a larger number of attribute types are considered to be more accurately classified. In this paper, when rules are integrated, a reduced rule number is achieved by integration into finer rules.

That said, in some instances a primary rule will contradict a secondary rule. Here, the instance that generates the secondary rule is judged to differ depending on the hyperplane and the associated rule generated during primary rule training. Making this distinction adequately explains cases in which contradictory rules are extracted. In contradictory rules, the attributes and values are exactly the same but the class labels differ, or the class labels and attributes that appear in the rules are the same but the attribute values differ. In these cases, the rules that appear in the secondary rules are integrated into the primary rules. The decision is made based on the number of samples in the running data set. More specifically, the running data set LD' has a greater number of samples than the running data set used to generate the secondary rules for the LDf. Thus, the primary rules encompass more samples. This means that when comparing primary rules and secondary rules on the basis of reliability, the primary rules can be regarded as having greater reliability. This is why the secondary rules are integrated into the primary rules.

4 Example

To illustrate the use of our algorithm, we first apply it to a credit approval data set used in a recent benchmark study. The data set is publicly available as the CARD1 data set. The data set contains a total of 690 samples: 518 training samples and 172 test samples. Altogether, the samples contain 51 attributes, of which 6 are continuous and the rest are binary. Because no detailed explanation is available on what each of the attributes represents, continuous input attributes 4, 6, 41, 44, 49, and 51 are simply labeled C4, C6, C41, C44, C49, and C51, respectively. The remaining binary-valued attributes are labeled D1, D2, D3, D5, D7, …, D40, D42, D43, D45, D46, D47, D48, and D50. Therefore, the number of input units is 51, and since the samples are in two groups, the number of output units is 2. We extracted 50% (259) of the instances in the CARD1 data set at random, and trained and pruned the first neural network by using the extracted set designated as LD'. From the resulting network, we extracted rules using the decision tree obtained with J4.8 and performed rule re-extraction in accordance with the Re-RX algorithm and the re-extraction threshold $\delta 1 = \delta 2 = 0.05$. The resulting primary rules are expressed as follows.

R1: If D42=0, then predict Class 1.
R2: If D42=1 and D7=0 and D19=0 and D43=0, then predict Class 1.
R3: If D42=1 and D7=0 and D19=0 and D43=1, then predict Class 2.
R4: If D42=1 and D7=0 and D19=1, then predict Class 1.
R5: If D42=1 and D7=1, then predict Class 2.

Based on these primary rules, we produced the data set for 70 instances, LDf, which was then used to train and prune the second neural network. We next performed rule re-extraction by using the same procedure applied for the first neural network. The resulting secondary rules are expressed as follows.

Rf1: If D42=0, then predict Class 1.
Rf2: If D42=1 and D13=0, then predict Class 1.
Rf3: If D42=1 and D13=1, then predict Class 2.

Comparison of the above primary and secondary rules immediately shows that R1 and Rf1 represent the same rule. Rf1 was therefore integrated into R1, and the following final rule set was the result.

R1: If D42=0, then predict Class 1.
R2: If D42=1 and D7=0 and D19=0 and D43=0, then predict Class 1.
R3: If D42=1 and D7=0 and D19=0 and D43=1, then predict Class 2.
R4: If D42=1 and D7=0 and D19=1, then predict Class 1.
R5: If D42=1 and D7=1, then predict Class 2.
Rf2: If D42=1 and D13=0, then predict Class 1.
Rf3: If D42=1 and D13=1, then predict Class 2.

The correct answer ratios obtained with the learning data set and the test data set using these rules were 96.14% and 94.19%, respectively. The value of the test data set was 5% higher than the test data set value of 89.53% reported by rule extraction with the well-known C4.5 variant as the Re-RX algorithm [9]. Moreover, the number of rules was 14 with Re-RX, but only 7 with E-Re-RX because no re-extraction was performed.

5 Results

Using the same methods, we performed all experiments on the CARD3, German Credit, Thyroid, and Pageblock data. All of this data is publicly available from the UCI repository [12].

The German Credit data set includes 56 discrete attributes and 7 continuous attributes. It has 2 classes indicating "good customer" and "bad customer." The Thyroid data set includes 15 discrete attributes and 3 continuous attributes. It has 3 classes indicating "normal," "hyper," and "hypo." The Pageblock data set includes 6 discrete attributes and 4 continuous ones. It has 5 classes indicating "text," "horizontal line," "graphic," "vertical line," and "picture." In each of these data sets, as in the CARD1 data, the attributes on each line are labeled with a D for a discrete attribute or a C for a continuous attribute.

Our experimental results are shown in Table 1, which shows the recognition rates for the data sets, and Table 2, which shows the number of rules. Our E-Re-RX results are presented with reference to Hara and Hayashi [13], [37]. Our Re-RX results are presented with reference to Setiono et al. [9]. Our results for MNNEX, PITS, Neural Network Bagging (shown as "NNBagging" in Table 1), and Neural Network AdaBoosting (shown as "NN AdaBoosting" in Table 1) are presented with reference to Akhand et al. [10].

We conducted each of the E-Re-RX experiments by setting $\delta_1 = \delta_2 = 0.05$, except for the German Credit experiments, where we set $\delta_1 = \delta_2 = 0.09$.

6 Discussion

In these experiments, we found the recognition accuracy offered by the Re-RX algorithm [9] to be more than sufficient for Card1 and Card3. We believe this can be explained by the fact that we partitioned the learning data set and worked with only part of it. In short, by working with a partial data set, we reduced the number of instances that lead to overfitting, while keeping a number just high enough for the primary rule set. Next, using only the data that did not satisfy the primary rule set, we performed another round of rule extraction, which was able to extract rules that could not have been extracted from the learning data set and produced a high recognition rate. A comparison of the number of rules showed that using an ensemble neural network sometimes increased the number of rules. However, by integrating rules, we were able to eliminate redundant rules, which seemed to hold the level of redundancy to a minimum.

Table 1. Comparison of recognition accuracy levels achieved by various methods with each data set

	E-Re-RX	Re-RX	MNNEC	PITS	NN Bagging	NN AdaBoosting
Card1	94.14%	89.53%	-	-	-	-
Card2	79.65%	86.38%	-	-	-	-
Card3(0.5)	94.77%	88.95%	-	-	-	-
Card3(0.7)	95.93%					
German Credit	82.20%	80.48%	76.48%	77.52%	75.60%	75.12%
Thyroid	99.64%	-	-	94.20%	94.11%	96.72%
Page Block	95.25%	-	-	-	92.58%	96.30%

Table 2. Number of rules resulting from the E-Re-RX experiment compared to that resulting from the Re-RX experiment

	Card1	Card2	Card3 (0.5)	Card3 (0.7)	German Credit	Thyroid	Page Block
E-Re-RX	7	6	14	17	17	6	9
Re-RX	13	7	7		41	—	—

Next, for the German Credit data, we were able to obtain the highest level of recognition accuracy with E-Re-RX. In particular, we exceeded the Re-RX algorithm by 2%, although it has a high recognition accuracy, and reduced the number of rules by a significant 40%. Here again, we attribute the difference to working with a partial learning data set. Using fewer instances, as in *LD*', made it possible to classify those instances with fewer rules. Likewise, including a fewer instances in *LDf* resulted in fewer extracted rules. Accordingly, it can be seen that even after rule integration, our method resulted in fewer rules extracted than did the Re-RX algorithm.

For the Thyroid data set, we achieved results better than the existing methods: 5% better than Neural Network Bagging, and 3% better than Neural Network Adaboosting. We attribute this to the fact that we extracted Class 2 and Class 3 rules in the primary rule set, and Class 1 and Class 3 rules in the secondary rule set. The Thyroid data set consists almost entirely of instances that are Class 3, so in an ordinary fitting, a bias will exist toward fitting to Class 3, and so it is assumed to sometimes be impossible to fit to Class 1 and Class 2 correctly. Akhand et al. [10] reported a maximum value of 94.81%, but found that in the absence of a bias toward Class 3, changes in accuracy improvements were dictated by the extent to which Class 1 and Class 2 were fitted accurately. In our research, we were able to extract Class 1 and Class 3 rules from the partial data, and the instances that did not satisfy these rules were Class 2 in many cases. We were able to fit to Class 2 efficiently in this way and to extract rules with high recognition accuracy, even in a multiclass problem. As a result, the accuracy was much higher than that of previous methods.

Finally, for the Pageblock data set, our results were worse than those by Neural Network Adaboosting, but 3% better than those by Neural Network Bagging. Here, for reasons that are the same as in the Thyroid data set, dealing with a partial data set allowed us to extract Class 1 and Class 2 rules in the primary rule set for instances that did not satisfy the primary rules of Class 3, Class 4, or Class 5. In the secondary rule set, we were able to extract Class 4 and Class 5 rules, which can be seen to have high recognition accuracy. However, we were unable to extract Class 3 rules in this experiment. This can be attributed to the fact that Class 3 instances made up no more than 0.5% of the total data set. In other words, because Class 3 exerted little influence on the weight correction, it was not possible to classify by Class 3. It is thought this can be improved by preparing a data set consisting of instances that do not satisfy the primary or secondary rule sets and performing fitting and rule extraction to extract rules related to Class 3. This suggests that an effective approach to multiclass problems would be an iterative method in which instances that do not satisfy the rules are gathered into a new data set, on which fitting and rule extraction are performed iteratively until rules that identify all of the classes have been extracted. This implies a more intensive use of computing resources, but because it avoids the problem of setting up an unknown number of neural networks, we still consider it to be an effective method.

7 Conclusion

We first surveyed the various theoretical and historical backgrounds for neural data analysis to investigate the approaches for ensemble neural network rule extraction. Next, we set out to address three problems in ensemble neural networks: to increase recognition rates, to extract rules from ensemble neural networks, and to minimize the use of computing resources. We proposed a minimal ensemble neural network consisting of two standard MLPs, which enabled high recognition accuracy and the extraction of comprehensible rules. Furthermore, this enabled rule extraction that resulted in fewer rules than those in previously proposed methods. The results make it possible for the output from an ensemble neural network to be in the form of rules, thus breaking open the "black box" of ensemble neural networks. Ensemble neural networks promise a new approach to data mining, and we are confident that our results will help make data mining more useful and increase the opportunities to use data mining with high recognition accuracy.

Finally, as future work, we provide the following three open questions on the E-Re-RX algorithm:

1) The first is “Can the proposed E-Re-RX algorithm be extended to an ensemble neural network consisting of three or more MLPs and extract comprehensible rules?”
2) The second is “Can the proposed E-Re-RX algorithm find various optimal parameter values so that we can get comprehensible rules with higher recognition accuracy?”
3) The third is “Can the proposed E-Re-RX algorithm be extended to the use of neural network structures other than a standard MLP?”

References

1. Breiman, L.: Bagging predictors. Mach. Learn. 24, 123–140 (1996)
2. Freund, Y., Schapire, R.: Experiments with a new boosting algorithm. In: Proc. of the Thirteenth International Conference on Machine Learning, Bari, Italy, pp. 148–156 (1996)

3. Zhang, G.P.: Neural networks for classification: A Survey. IEEE Trans. Systems, Man and Cybernetics–Part C: Applications and Reviews 30(4), 451–462 (2000)
4. Rokach, L.: Ensemble-based classifiers. Artificial Intelligence Review 33, 1–39 (2010)
5. Yao, X., Islam, M.: Evolving artificial neural network ensembles. IEEE Computational Intelligence Magazine 3(1), 31–42 (2008)
6. Cybenko, G.: Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals, and Systems 2, 303–314 (1989)
7. Setiono, R.: A penalty-function approach for pruning feedforward neural networks. Neural Comp. 9(1), 185–204 (1997)
8. Bishop, C.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
9. Setiono, R., Baesens, B., Mues, C.: Recursive neural network rule extraction for data with mixed attributes. IEEE Trans. Neural Netw. 19, 299–307 (2008)
10. Akhand, M.A.H., Murase, K.: Neural Network Ensembles. Lambert Academic Publishing (LAP) (2010)
11. Alpaydin, E.: Multiple Neural Networks and Weighted Voting. IEEE Trans. on Pattern Recognition 2, 29–32 (1992)
12. University of California, Irvine Machine Learning Repository, `http://archive.ics.uci.edu/ml/`
13. Hara, A., Hayashi, Y.: Ensemble neural network rule extraction using Re-RX algorithm. In: Proc. of WCCI (IJCNN) 2012, Brisbane, Australia, June 10-15, pp. 604–609 (2012)
14. Mitra, S., Hayashi, Y.: Neuro-Fuzzy Rule Generation: Survey in Soft Computing Framework. IEEE Trans. Neural Netw. 11(3), 748–768 (2000)
15. Gallant, S.I.: Connectionist expert systems. Commun. ACM 31, 152–169 (1988)
16. Saito, K., Nakano, R.: Medical diagnosis expert systems based on PDP model. In: Proc. IEEE Int. Conf. Neural Netw, San Diego, CA, pp. I.255–I.262 (1988)
17. Hayashi, Y.: Neural expert system using teaching fuzzy input and its application to medical diagnosis. Inform. Sci.: Applicat. 1, 47–58 (1994)
18. Hayashi, Y.: A neural expert system with automated extraction of fuzzy if-then rules and its application to medical diagnosis. In: Lippmann, R.P., Moody, J.E., Touretzky, D.S. (eds.) Advances in Neural Information Processing Systems, pp. 578–584. Morgan Kaufmann, Los Altos (1991)
19. Hudson, D.L., Cohen, M.E., Anderson, M.F.: Use of neural network techniques in a medical expert system. Int. J. Intell. Syst. 6, 213–223 (1991)
20. Takagi, H.: Fusion technology of fuzzy theory and neural network—Survey and future directions. In: Proc. Int. Conf. Fuzzy Logic and Neural Networks, Iizuka, Japan, pp. 13–26 (1990)
21. Buckley, J.J., Hayashi, Y., Czogala, E.: On the equivalence of neural nets and fuzzy expert systems. Fuzzy Sets Syst. 53(2), 129–134 (1993)
22. Hayashi, Y., Buckley, J.J.: Approximation between fuzzy expert systems and neural networks. Int. J. Approx. Res. 10, 63–73 (1994)
23. Buckley, J.J., Hayashi, Y.: Numerical relationship between neural networks, continuous function and fuzzy systems. Fuzzy Sets Syst. 60(1), 1–8 (1993)
24. Buckley, J.J., Hayashi, Y.: Hybrid neural nets can be fuzzy controllers and fuzzy expert systems. Fuzzy Sets and Syst. 60, 135–142 (1993)
25. Golea, M.: On the complexity of rule extraction from neural networks and network querying. In: Proc. the AIBS 1996 Workshop on the Rule Extraction from Trained Neural Networks, Brighton, UK, pp. 51–59 (1996)

26. Roy, A.: On connectionism, rule extraction, and brain-like learning. IEEE Trans. Fuzzy Systems 8(2), 222–227 (2000)
27. Zhou, Z.-H.: Rule Extraction: Using Neural Networks or for Neural Networks? J. Comput. Sci. & Technol. 19(2), 249–253 (2004)
28. Buckley, J.J., Hayashi, Y.: Fuzzy neural networks: A survey. Fuzzy Sets Syst. 66, 1–13 (1994)
29. Hayashi, Y., Buckley, J.J., Czogala, E.: Fuzzy neural network with fuzzy signals and weights. Int. J. Intell. Syst. 8(4), 527–573 (1993)
30. Ishibuchi, H., Kwon, K., Tanaka, H.: A learning algorithm of fuzzy neural networks with triangular fuzzy weights. Fuzzy Sets Syst. 71, 277–293 (1995)
31. Feuring, T., Buckley, J.J., Hayashi, Y.: A gradient descent learning algorithm for fuzzy neural networks. In: Proc. IEEE Int. Conf. Fuzzy Syst. FUZZ-IEEE 1998, Anchorage, AK, pp. 1136–1141 (1998)
32. Bologna, G.: Is it worth generating rules from neural network ensembles? J. of Applied Logic 2, 325–348 (2004)
33. Bologna, G.: A study on rule extraction from several combined neural networks. Int. J. Neural Syst. 11(3), 247–255 (2001)
34. Bologna, G.: A model for single and multiple knowledge based networks. Artificial Intelligence in Medicine 28, 141–163 (2003)
35. Zhou, Z.-H.: Extracting symbolic rules from trained neural network ensembles. AI Communications 16, 3–15 (2003)
36. Zhou, Z.-H.: Ensemble neural networks: Many could be better than all. Artificial Intelligence 137, 239–263 (2002)
37. Hara, A., Hayashi, Y.: A new neural data analysis approach using ensemble neural network rule extraction. In: Villa, A.E.P., Duch, W., Érdi, P., Masulli, F., Palm, G. (eds.) ICANN 2012, Part I. LNCS, vol. 7552, pp. 515–522. Springer, Heidelberg (2012)
38. Hansen, L.K., Salamon, P.: Neural network ensembles. IEEE Trans. Pattern Analysis and Machine Intelligence 12, 993–1001 (1990)
39. Adeodato, P.J.L., et al.: MLP ensembles improve long term prediction accuracy over single networks. Int. J. Forecasting 27, 661–671 (2011)
40. Hornik, K., et al.: Multilayer feedforward networks are universal approximators. Neural Netw. 2(5), 359–366 (1989)
41. Chorowski, J., Zurada, J.M.: Extracting Rules from Neural Networks as Decision Diagrams. IEEE Trans. Neural Netw. 22(12), 2435–2446 (2011)
42. Augasta, M.G., Kathirvalavakumar, T.: Reverse engineering the neural networks for rule extraction in classification problems. Neural Processing Letters 35, 131–150 (2012)
43. Barakat, N., Bradley, A.P.: Rule extraction from support vector machines: A review. Neurocomputing 74, 178–190 (2010)
44. Liu, S., et al.: Combined rule extraction and feature elimination in supervised classification. IEEE Trans. Nanobioscience 11(3), 228–236 (2012)
45. Zhou, Z.H.: Medical diagnosis with C4.5 rule preceded by artificial neural network ensemble. IEEE Trans. Information Technology in Biomedicine 7(1), 37–42 (2003)
46. Zhou, Z.H., et al.: A statistics based approach for extracting priority rules from trained neural networks. In: Proc. IEEE-INNS-ENNS Int. Conf. Neural Netw., Como, Italy, vol. 3, pp. 401–406 (2000)
47. We, X., et al.: Top 10 algorithms in data mining. Knowledge and Information Systems 14, 1–17 (2008)

48. Robert, B., et al.: Boosting of the margin: A new explanation for the effectiveness of voting methods. The Annals of Statistics 26(5), 1651–1686 (1998)
49. Duch, W., Setiono, R., Zurada, J.M.: Computational intelligence methods for rule-based data understanding. Proceedings of the IEEE 92(5), 771–805 (2004)
50. Tickle, A.B., et al.: The truth will come to light: Directions and challenges in extracting the knowledge embedded within trained artificial neural networks. IEEE Trans. Neural Netw. 9, 1057–1068 (1998)
51. Lin, C.T., Lee, C.S.G.: Neural fuzzy systems—A neuro–fuzzy synergism to intelligent systems. Prentice-Hall, Englewood Cliff (1996)
52. Khosravi, A., et al.: Comprehensive review of neural network-based prediction intervals and new advances. IEEE Trans. Neural Netw. 22(9), 1341–1356 (2011)