Network Based Rule Representation for Knowledge Discovery and Predictive Modelling

Han Liu, Alexander Gegov and Mihaela Cocea
School of Computing
University of Portsmouth
Portsmouth, United Kingdom
Han.Liu@port.ac.uk, Alexander.Gegov@port.ac.uk and Mihaela.Cocea@port.ac.uk

***Abstract*—Due to the vast and rapid increase in data, data mining has been an increasingly important tool for the purpose of knowledge discovery to prevent the presence of rich data but poor knowledge. In this context, machine learning can be seen as a powerful approach to achieve intelligent data mining. In practice, machine learning is also an intelligent approach for predictive modelling. A special type of machine learning methods, which are known as rule based methods such as decision trees, can be used to build a rule based system as a special type of expert systems for both knowledge discovery and predictive modelling. A rule based system may be represented through different structures. The techniques for representing rules are known as rule representation, which is significant for knowledge discovery in relation to the interpretability of the model, as well as for predictive modelling with regard to efficiency in predicting unseen instances. This paper justifies the significance of rule representation. Some networked topologies for rule representation are introduced against existing techniques. The network topologies are validated using complexity analysis in order to show their advantages comparing with the existing techniques in terms of model interpretability and computational efficiency.**

Keywords—Data Mining; Machine Learning; Knowledge Discovery; Predictive Modelling; Knowledge Representation; If-Then Rules; Rule Based Classification

I. INTRODUCTION

The daily increase in the size of data has motivated knowledge discovery in databases [1]. This is in order to prevent the presence of rich data but poor knowledge [2], which means that there would potentially be a large amount of knowledge that can be extracted from data. Data mining is commonly seen as an important tool for knowledge discovery [3]. Data mining can be done by subject experts through manual analysis of data or by machines through empirical analysis of data. Due to the presence of big data, it is necessary to employ more intelligent methods to achieve intelligent data mining. In this context, machine learning can be seen as a powerful approach that could serve for such data mining tasks. On the other hand, machine learning is also an intelligent approach for predictive modelling in a black box manner while knowledge discovery follows a white box approach, i.e. predictive modelling emphasizes on the mapping from inputs to outputs without interpreting the reasons whereas knowledge discovery needs to interpret the reasons for the mapping. The rest of this section focuses on the background on data mining and machine learning as well as applications of rule based systems for knowledge discovery and predictive modelling.

Machine learning is a branch of artificial intelligence and involves two stages: training and testing [4]. Training aims to learn something from known properties by using learning algorithms and testing aims to make predictions on unknown properties by using the knowledge learned in the training stage. From this point of view, training and testing are also known as learning and prediction respectively. In practice, a machine learning task aims to build a model, which is further used to make predictions, through the use of learning algorithms. Therefore, this task is usually referred to as predictive modelling. Machine learning could be divided into two types: supervised learning and unsupervised learning, in accordance with the form of learning. Supervised learning means learning with a teacher because all instances from a training set are labelled. The aim of this type of learning is to build a model by learning from labelled data and then to make predictions on other unlabeled instances with regard to the value of a predicted attribute. The predicted value of an attribute could be either discrete or continuous. Therefore, supervised learning could be involved in both classification and regression tasks for categorical prediction and numerical prediction respectively. In contrast, unsupervised learning means learning without a teacher. This is because all instances from a training set are unlabeled. The aim of this type of learning is to find previously unknown patterns from data sets. It includes association, which aims to identify correlations between attributes, and clustering, which aims to group objects based on similarity measures.

On the other hand, as mentioned earlier in this section, machine learning algorithms are popularly used in data mining tasks to discover some previously unknown pattern. Therefore, this task is usually referred to as knowledge discovery. From this point of view, data mining tasks also involve classification, regression, association and clustering. Both classification and regression can be used to reflect the correlation between multiple independent variables and a single dependent variable. The difference between classification and regression is that the former typically reflects the correlation in qualitative aspects whereas the latter reflects it in quantitative aspects. Association is used to reflect the correlation between multiple independent

variables and multiple dependent variables in both qualitative and quantitative aspects. Clustering can be used to reflect patterns in relation to grouping of objects.

One practical application of machine learning is the construction of expert systems. Rule based systems are a special type of expert system, which typically consists of a set of if-then rules referred to as a rule set. Rule based systems could be constructed by adopting rule based methods that belong to a special type of machine learning methods and can serve for classification, regression and association. A unified framework for construction of rule based classification systems has been recently developed in [5]. In this framework, rule representation is justified as a significant impact factor for the efficiency of rule based systems in predicting unseen instances. In addition, rule representation is also important for knowledge extraction due to the interpretability of a particular representation. In other words, poor representation would usually make a rule set become cumbersome and less readable.

The rest of this paper is organized as follows. Section II outlines the significance of rule representation in both data mining and machine learning tasks. Section III describes the existing techniques of rule representation from conceptual point of view and compares them in terms of their complexity. Section IV introduces some network topologies used as network based rule representation techniques. The techniques are validated through theoretical analysis in terms of computational complexity and structure complexity in comparison with existing techniques. Section V summaries the completed work and highlights the contributions to research and development in data mining, machine learning and expert systems. Further directions of this research area are also suggested at the end of this paper.

II. Significance of Rule Representation

As mentioned in Section I, rule representation is a significant impact factor that may affect both computational efficiency in the testing stage for machine learning tasks and knowledge interpretability for data mining tasks. This section focuses on justifying why rule representation is significant for knowledge discovery and predictive modelling.

For the purpose of knowledge discovery, it is important that the knowledge is highly interpretable for people, which means the knowledge needs to be represented in such a way that makes it easier to read and understand. In the context of rule based systems, knowledge is actually represented in the form of if-then rules. Higgins justified in [6] why a rule based knowledge representation is necessary with the following arguments:

- A network was conceived of in [7], which needs a number of nodes exponential in the number of attributes in order to restore the information on conditional probabilities of any combination of inputs. It is argued in [6] that the network restores a large amount of information that is mostly less valuable.
- Another type of networks known as Bayesian Networks introduced in [8] needs a number of nodes same as the number of attributes. However, the network only restores the information on joint probabilities based on the assumption that each of the input attributes is totally independent of the others. Therefore, it is argued in [6] that this network is unlikely to predict more complex relationships between attributes due to lack of information on correlational probabilities between attributes.
- There are some other methods that fill the gaps in Bayesian Networks by deciding to only choose some higher-order conjunctive probabilities such as the first neural networks [9] and another method based on correlation/dependency measure [10]. However, it is argued in [6] that these methods still need to be based on the assumption that all attributes are independent of each other.

On the basis of above arguments, Higgins motivated the use of rule based knowledge representation and mentioned the advantage that rules used to interpret relationships between attributes can provide explanations with regard to a decision of an expert system [6].

However, like data structures [11], rules can also be represented in different structures which may provide different level of readability and interpretability. In this context, rules need to be represented in a way that makes it easier for people to read and understand the knowledge interpreted as rules. Therefore, the form of rule representation is significant in data mining tasks for knowledge discovery.

For the purpose of predictive modelling, rule representation is also significant as mentioned in Section I. This is because rules represented in different structures would usually lead to different levels of computational efficiency in the testing stage for machine learning tasks. In software engineering, different data structures usually lead to different levels of computational efficiency in some operations relating to data management such as insertion, update, deletion and search. As mentioned in [5, 12], the main objective in the prediction stage is to find the first firing rule by searching through a rule set. In this context, it indicates that predicting on unseen instances by a rule set is a search problem. As mentioned above, different data structures may provide different levels of search efficiency. For example, a collection of items stored in a linear list can only be searched linearly if these items are not given indexes. However, if the same collection of items is stored in a tree, then it is achievable to have a divide and conquer search. The former way of search would be in linear time whereas the latter way in logarithmic time. In this sense, efficiency in search of firing rules would also be affected by the structure of the rule set. It is also defined in [13] that one of the biases for rule based methods is 'search bias', which refers to the strategy used for the hypothesis search. In general, what is expected is to make it unnecessary to examine the whole rule set, but as few rule terms as possible. More detailed justifications about this are given in Section III and IV.

On the basis of above descriptions, rule representation is considered highly significant in both data mining and machine learning tasks, which means that a rule set is expected to have a high level of interpretability for knowledge discovery as well as to demonstrate a low level of computational complexity for predictive modelling.

III. Existing Rule Rrepersentations

As mentioned in Section II, rule representation is significant for both knowledge discovery and predictive modelling. This section describes two existing techniques of rule representation, namely decision trees and linear lists, and compares them with respect to their computational complexity and interpretability.

A. Decision Trees

Decision Tree is an automatic representation for classification rules that are generated by a 'divide and conquer' approach [14]. This indicates that if a rule based method that follows the above-named approach is adopted to generate rules, then the rules are automatically represented in a tree structure. However, decision tree representation is criticized by Cendrowska and identified as a major cause of overfitting in [15] due to the replicated sub-tree problem as illustrated in Fig.1.

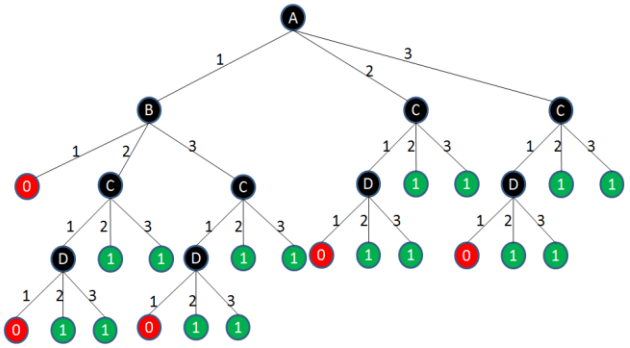

Fig.1. Cendrowska's replicated subtree example

It can be seen from the Fig.1 that the four sub-trees which have node C as their roots are identical. Cendrowska justified in [15] that those rules which have no common attributes would not be able to fit in a tree structure and that replicated sub-tree problem would arise if such rules are forced to fit in a tree structure. It is also argued in [15, 16] that it is required to examine the entire tree in order to extract rules about a single classification in the worst case. This drawback on representation makes it difficult to manipulate for expert systems and thus seriously lowers the computational efficiency in predicting unseen instances. For the purpose of predictive modelling, as mentioned in Section II, computational efficiency in the testing stage is significant especially when the expert systems to be constructed are time critical [17].

On the other hand, decision trees are often quite complex and difficult to understand [13]. Even if decision trees are simplified by using pruning algorithms, it is still difficult to avoid that the decision trees become too cumbersome, complex and inscrutable to provide insight into a domain for knowledge usage [13, 14]. This undoubtedly lowers the interpretability of decision trees and is thus a serious drawback for the purpose of knowledge discovery.

All of the limitations mentioned above motivate the direct use of 'if-then' rules represented by a linear list structure. Mode details about linear lists are introduced in the following subsection.

B. Linear Lists

As mentioned in Subsection A of this section, decision tree representation has serious limitations for knowledge discovery and predictive modelling and thus the direct use of if-then rules is motivated. In comparison to decision trees, linear lists do not need to constrain that all rules must have common attributes and thus reduces the presence of redundant terms in a rule set. However, as if-then rules are represented in a linear list structure, predicting unseen instances in this representation is demonstrated in linear search with the time complexity of O(n) while the total number of rule terms is used as the input size (n). This is because linear list representation follows a linear search by going through the whole rule set rule by rule in an outer loop; and by going through term by term for each rule in an inner loop. The process of linear search can be illustrated by using the example rule set below:

Rule 1: if x_1=0 and x_2=0 then y=0;
Rule 2: if x_1=0 and x_2=1 then y=0;
Rule 3: if x_1=1 and x_2=0 then y=0;
Rule 4: if x_1=1 and x_2=1 then y=1;

On the basis of above rule set, if an instance with two inputs (x_1=1 and x_2=1), then it needs to first go through Rule 1 checking the values of x_1 and x_2 and then move onto Rule 2 taking the same check again until Rule 4 is checked and found firing.

The above description implies that it may have to go through the whole rule set to find the first rule firing in the worst case. This would lead to huge computational costs when the representation is used to represent a rule set generated by learning from large training data. As mentioned in Section II, a rule representation technique is expected to identify the firing rules without the need to go through the whole rule set. Therefore, for the purpose of predictive modelling, linear lists still cannot fulfil the goal with regard to efficient search of firing rules. In this sense, it is necessary to develop another technique of rule representation which demonstrates a level of computational efficiency higher than linear time.

In addition, when a training set is large, there would be a large number of complex rules generated. In this case, the set of rules represented in a linear list structure would become very cumbersome and difficult to interpret for knowledge usage. In other words, a large number of complex rules represented in a linear list is quite like a large number of long paragraphs in an article, which would be very difficult for people to read and understand. Instead, people prefer to look at diagrams to gain information. In this sense, graphical representation of rules would be expected to improve the interpretability of knowledge discovered from data.

C. Discussion

On the basis of the above description about limitations of tree and list representations in terms of computational efficiency, the development of a new representation of classification rules is needed, which should have a level of efficiency higher than linear time in time complexity. This new representation is described in Section IV.

On the other hand, in addition to time complexity limitation, the two existing representations have the limitation of poor interpretability, especially when large data sets are involved. Higgins has developed a representation called rule based network in [6], which can improve the interpretability of knowledge representation in the context of probabilistic logic. Section IV introduces more details about this as well as the generalization of rule based network representation.

IV. Network Based Rule Representation

Section III identified the limitations of decision tree and linear list representations and outlined the need to develop new techniques for rule representation. This is in order to achieve a more efficient search of firing rules than linear search as well as to deliver a more interpretable representation of knowledge than decision trees and linear lists do. In addition, predictions can be made based on different logics such as deterministic, probabilistic and fuzzy logic. Therefore, this section introduces the three types of logic and a special type of rule based network representation developed by Higgins [6]. This section also introduces other modified versions of the network based rule representation, which includes a unified network topology for generalized representation in order to fulfil the topology being based on all of the three logics mentioned above.

A. *Logic*

Ross stated in [18] that logic is a small part of human capability for reasoning, which is used to assist people in making decisions or judgments. As mentioned in [19], in the context of Boolean logic, each variable is only assigned a binary truth value: 0 (false) or 1 (true). It indicates that reasoning and judgment are made under certainty resulting in deterministic outcome. From this point of view, this type of logic is also referred to as deterministic logic. However, in reality, people usually can only make decisions, judgment and reasoning under uncertainty. Therefore, the other two types of logic, namely probabilistic logic and fuzzy logic, are used more widely, both of which can be seen as an extension of deterministic logic. The main difference is that the truth value is not binary but continuous between 0 and 1. The truth value implies a probability of truth between true and false in probabilistic logic and a degree of that in fuzzy logic.

Deterministic logic deals with any events under certainty. For example, a crisp set has all its elements fully belong to it, i.e. each element has a full membership to the set.

Probabilistic logic deals with any events under probabilistic uncertainty. For the same example about sets, an element may be randomly allocated to one of five sets with normal distribution of probability. Once the element has been allocated to a particular set, then it has a full membership to the set.

Fuzzy logic deals with any events under non-probabilistic uncertainty. In the context of set theory, each set is referred to as a fuzzy set. This is because each element may not have a full membership to the set, i.e. the element belongs to the fuzzy set to an extent.

In the context of rule based systems, a deterministic rule based system would have each rule either fire or not. If it fires, the consequence would be deterministic. A probabilistic rule based system would have a firing probability for each rule. The consequence would be probabilistic depending on posterior probability of it given specific antecedents. A fuzzy rule based system would have a firing strength for each rule. The consequence would be weighted depending on the fuzzy truth value of the most likely outcome. In addition, fuzzy rule based systems deal with continuous attributes by mapping the values to a number of linguistic terms according to the fuzzy membership functions defined.

B. *Attribute-Value Oriented Rule Based Network*

As mentioned in Section II, both decision tree and linear list representations have their own limitations. A networked representation of classification rules is developed, which is called rule based networks and provides a higher level of computational efficiency than tree and list representations for the same rule set in the prediction stage.

The rule based network representation is illustrated in Fig.2, which is based on the relationship between attribute values and class labels and thus referred to as attribute-value oriented rule based network.

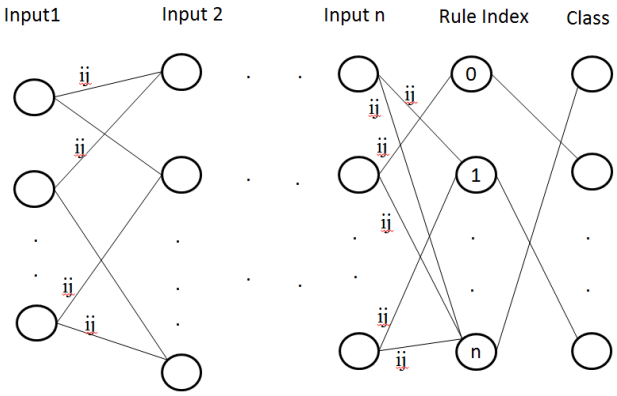

Fig.2. Rule Based Network v1

In general, this is an *n+2* layer network for classification problems. The first n layers represent n input attributes. In each of the layers, each node represents a value of the corresponding attribute. Therefore, the number of nodes in each of the n layers is dependent on the number of values for the corresponding attribute. The last second layer represents the rule index which is equal to the order of this rule minus one. For example, if the rule index is 0, it indicates that this is the first rule in the rule set. The number of nodes in this layer is dependent on the number of rules. The last layer in the network represents the class output and the number of nodes is dependent on the number of classes. There are also connections between different layers, which are to be explained further using specific examples. However, in general, the connections could be between two layers which are not adjacent to each other. For example, the nodes in the first layer could have connections with other nodes in the third layer. This is very like a travel route which includes a number of cities. In this context, each city is like a rule term and each route is like a rule. It is possible that there are cities which are not adjacent to each other but included in the same travel route. In addition, any two nodes

may have more than one connection. This is because the same part of conjunction of rule terms may be in two or more rules as illustrated by the rules below:

If a=0 and b=0 and c=0 then class=0;
If a=0 and b=0 then class=1;

In the context of travel route as mentioned above, this is like that there could be common cities included in different routes. In other words, there could be more than one path to reach between two cities.

The rule based network representation that is illustrated in Fig.2 would demonstrate a divide and conquer search for firing rules. It could be justified by the example rule set used in Subsection B of Section 3 for analysis of time complexity for linear lists. The rule set represented by the network representation is illustrated in Fig.3 and the values of the two inputs (x_1 and x_2) are both 1. Therefore, the two input layers (x_1 and x_2) both have the nodes labelled 1 and become green as illustrated in Fig.3.

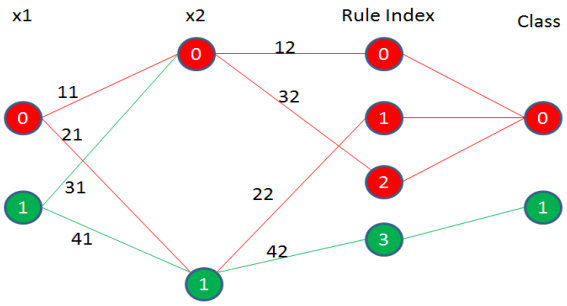

Fig.3. Rule Based Network example (v1) [5]

For Rule Based Networks, the prediction on an unseen instance is undertaken by going through rule terms in divide and conquer search (i.e. only going through those terms that fire). The total number of terms is used as the input size of data (n), which is the same as in linear list representation and thus the efficiency is O(log (n)). As it can be seen from Fig.3, it only takes three steps (going through connections '31', '41' and '42') to find the first rule firing (the rule index is 3). This is because the input value of x_1 is 1 and thus the connections '11' and '21' can be skipped. In the second layer, only connection '42' is checked since the input value of x_2 is 1 and thus 'the connections '12' and'32' can be skipped. In addition, the connection '22' is skipped as well because the connection '21' is already discarded and thus it is not worth to go through the connection '22' any more. The above descriptions indicate that finding the rules that fire does not involve examining the whole network, which makes the efficiency of the rule based network higher than that of linear lists, the latter of which is O (n) as mentioned in Section III and Table I. In practice, the advantage of rule based networks would significantly speed up the process of prediction when the corresponding rule set is generated by learning from large training data.

As mentioned in Section III, rule representation is also significant in fulfilling the requirement of interpretable knowledge representation. In this context, the network representation illustrated in Fig.2 would also demonstrate a good interpretability with respect to correlation between inputs and outputs. A correlation can be interpreted by connections between nodes in different layers. For example, it can be seen from Fig.3 that node 1 in the first layer (x_1) is firstly connected to node 1 in the second layer (x_2) and finally to node 1 in the last layer (Class) via node 3 in the third layer (Rule Index). These connections make up a path that interprets a correlation between the two input attributes (x_1 and x_2) and the class attribute. In comparison with linear list, the network representation only needs each attribute and its corresponding values to appear once as a layer and a node in the layer respectively. This also reduces the structure complexity and thus improves the interpretability of knowledge.

TABLE.I. Comparison in efficiency

Rule Based Network	Linear List	Decision Tree
O(log(n)), which indicates it is not required to examine a whole network.	O(n), which indicates it is required to examine a whole list in the worst case.	O(log(n)), which indicates it is not required to examine a whole tree but the value of n is likely to be higher than that in the other two representations due to the presence of redundant terms.

NB: n is the total number of rule terms in a rule set.

C. *Attribute Oriented Rule Based Network*

As mentioned in Section III, Higgins has developed a representation called rule based network as illustrated in Fig.4, which is based on the relationship between input attributes and class labels. It is thus referred to as attribute oriented rule based network and is fundamentally different from the one illustrated in Fig.2.

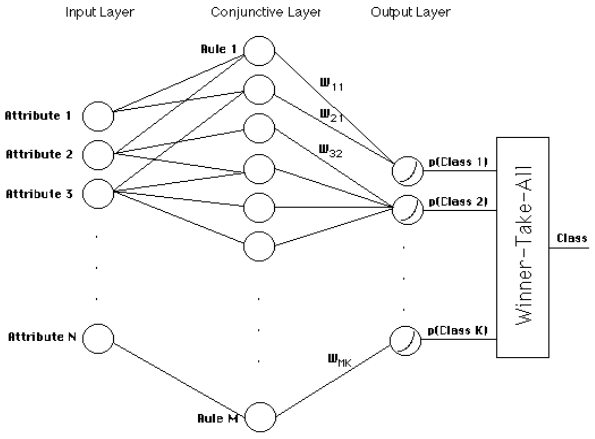

Fig.4. Higgins's non-deterministic rule based network for classification [6]

In this network, as explained in [6], each node in the input layer represents an input attribute. Each node in the middle layer represents a rule. The connections between the nodes in the input layer and the nodes in the conjunctive layer indicate which rules relate to which attributes. In the output layer, each node represents a class label. The connections between the nodes in the conjunctive layer and the nodes in the output layer reflect the mapping relationships between rule antecedents and classifications (consequents). Each of the connections is also weighted as denoted by w_{mk}, where m is the index of the rule and k is the index of the class. The weight reflects the

confidence of the rule for predicting the class given the antecedent of the rule. In this way, each class is assigned a weight, which is derived from the confidences of the rules having the class as consequents. The final classification is predicted by weighted majority voting, which is known as 'Winner-Take-All strategy' as illustrated in Fig.4 [6].

The network topology illustrated in Fig.4 could be seen as a special type of rule based network representation based on the relationship between input attributes and class labels. This is because of the possibility that there are two or more rules that fire with different classifications as rule consequences. This issue needs to be resolved by conflict resolution strategies as introduced in [20]. Higgins's network topology actually takes into account this conflict and deals with it by the 'Winner-Take-All strategy' [6]. Therefore, the network topology could be seen as a type of non-deterministic rule based network with certain inputs but uncertain outputs. However, the conflict mentioned above would never arise with the rule sets that are generated by adopting the divide and conquer approach. In this context, if the rule generation is based on deterministic logic, both inputs and outputs would be deterministic. As it is, the networked topology is modified to become a deterministic rule based network that is illustrated by Fig.5.

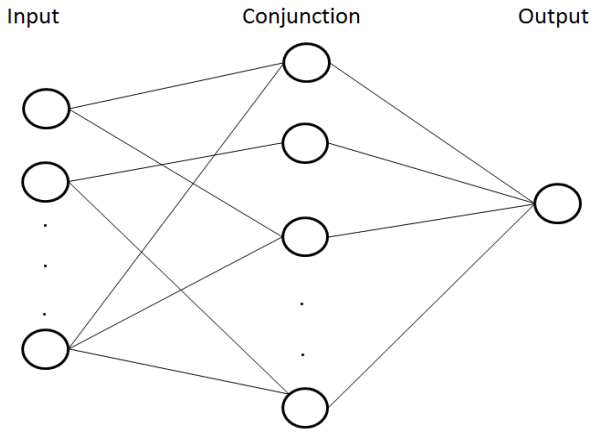

Fig.5. Deterministic Rule Based Network v2

In general, this is a three layer network. In the first layer, each node represents an input attribute and this layer is referred to as input layer. The number of nodes in this layer is dependent on the number of attributes in a data set. In the middle layer, each node represents a rule to make the conjunction among inputs and provide outputs for the node in the last layer and thus the middle layer is referred to as conjunction layer. The number of nodes in this layer is dependent on the number of rules generated. The only node in the last layer represents the class output and thus this layer is referred to as output layer. In addition, the nodes in input layer usually have connections to other nodes in the conjunction layer. Each of the connections represents a condition judgment which is explained further using specific examples. However, a node in the input layer may not necessarily have connections to other nodes in the conjunction layer. This is due to a special case that an attribute may be totally irrelevant to making a classification. In other words, this attribute is not involved in any rules in the form of rule terms. From this point of view, this version of rule based network representation can help identify the relevance of attributes for feature selection tasks, which is listed in Table II and discussed further in this section.

TABLE.II. Comparison in interpretability

Criteria	RBN	LL	DT
correlation between attributes and classes	explicit	implicit	poor
relationship between attributes and rules	explicit	implicit	implicit
ranking of attributes	explicit	poor	poor
ranking of rules	explicit	explicit	poor
attribute relevance	explicit	poor	poor
overall	high	medium	low

NB: RBN= Rule Based Network, LL= Linear List and DT= Decision Tree

On the other hand, this type of networked representation is based on the relationship between attributes and class labels as mentioned earlier in this subsection. Therefore, this representation can be used to reflect correlations between input attributes and class labels, i.e. it enables the identification of the input attributes that have the highest influence on determining each of the class labels.

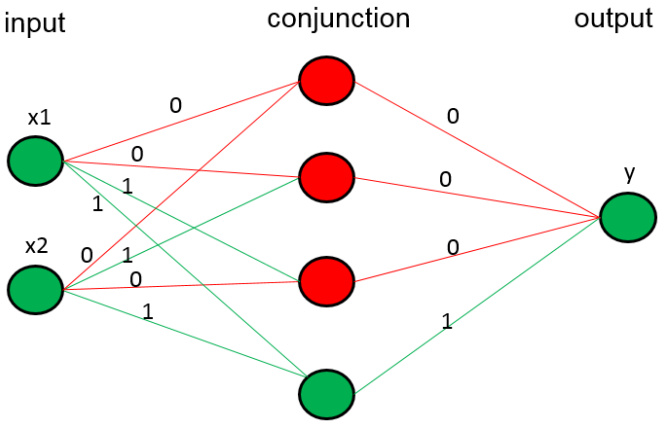

Fig.6. Deterministic Rule Based Network example (v2)

The example rule set that is used in Section III and represented by this network topology is illustrated in Fig.6. In this diagram, both input values are supposed to be 1 (shown as green) and each node in input layer represents an input attribute; each node in the middle layer represents a rule and the layer is referred to as conjunction layer due to the fact that each rule actually reflects the mapping between inputs and outputs and that the output values strongly depends on the conjunction of input values; finally, the node in the output layer represents the class attribute. On the other hand, each of the connections between input layer and conjunction layer represents a condition judgment. If the condition is met, then the connection is colored by green. Otherwise, it is colored by red. In addition, each of the connections between the conjunction layer and the output layer represents an output value from the corresponding rule. In other words, if all of the conditions in a rule are met, then the corresponding node in the conjunction layer becomes green. Otherwise, the corresponding node becomes red. The former case would result in that a node

representing a rule becomes green and that the output value from the rule is assigned to the class attribute in output layer. In the meantime, the connection between the node representing the rule and another node representing the class attribute becomes green, which means the class attribute would be assigned the output value from the rule. In contrast, the latter case would result in that the node in conjunction layer becomes red and that the output value from the corresponding rule cannot be assigned to the class attribute.

As illustrated in Table II, this type of networked rule representation also shows the relationship between attributes and rules explicitly as shown connections between nodes in input layer and nodes in conjunction layer. In addition, the networked representation also introduces a ranking for both input attributes and rules based on their importance. The importance of an input attribute is measured by the weighted average of ranks for those rules that relate to the input attribute. For example, an attribute A relates to two rules namely rule 1 and rule 2. If the ranks for rule 1 and rule 2 are 4 and 8 respectively, then the average of ranks would be 6= ((4+8)/2). In real applications, this characteristic about ranking of attributes may significantly contribute to both knowledge discovery and feature selection with respect to feature importance. Besides, the strength of the representations also lies in the strong interpretability on mapping relationship between inputs and outputs, which is significantly useful for knowledge discovery. On the basis of the above descriptions, the rule based network illustrated in Fig.5 is thus a practically significant technique in data mining tasks.

D. Generalized Rule Based Network

As mentioned in Subsection B of this section, a rule set may have some or all rules non-deterministic in terms of relationships between rule antecedents and consequents due to the presence of uncertainty in datasets. In this context, the rule set would be used to predict classes based on probabilistic or fuzzy logic. Therefore, a unified topology for rule based networks, which could fulfil being based on different logics such as deterministic, probabilistic and fuzzy logic, is developed and illustrated in Fig.7.

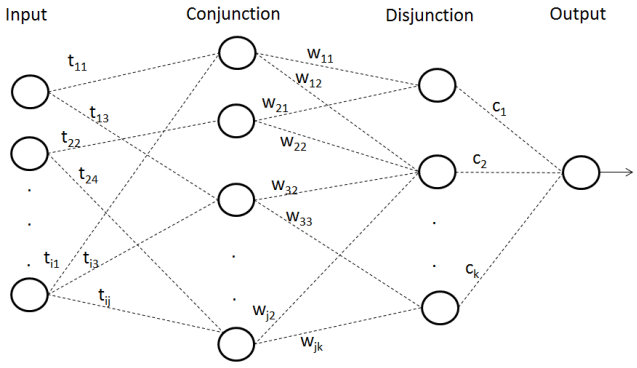

Fig.7. Unified Rule Based Network

In this network topology, the modifications are made to the one illustrated in Fig.5 by adding a new layer called disjunction and assigning a weight to each of the connections between nodes. The disjunction layer is similar to the output layer in Higgins's network topology illustrated in Fig.4. In this layer, each node represents a class label and the number of nodes is dependent on the number of classes. However, the final prediction is not necessarily made by choosing the most common class which has the highest posteriori probability in total. In contrast to Fig.4 and Fig.5, the topology also allows representing inconsistent rules, which means that the same rule antecedent could be mapped to different classes (consequents). For example, the first node in the conjunction layer is mapped to both the first and the second node in the disjunction layer as illustrated in Fig.7. With regard to the weights assigned to the connections between nodes, they would represent the truth values if the computation is based on deterministic or fuzzy logic. The truth value would be crisp (0 or 1) for deterministic logic whereas it would be continuous (between 0 and 1) for fuzzy logic. If the computation is based on probabilistic logic, the weights would represent the probabilities of the corresponding cases.

In the context of deterministic logic, each of the connections between the nodes in the input layer and the nodes in the conjunction layer would be labelled 1 for its weight, i.e. t_{ij}= 1 where i is the index of the attribute and j is the index of the rule, if the corresponding condition as part of the rule antecedent is met. A rule would have its antecedent satisfied if and only if all of the conditions are met. In this case, the rule is firing to indicate its consequent (as the class predicted) which is represented by a node in the disjunction layer. If the rule is consistent, the corresponding node should have a single connection to another node in the disjunction layer. The connection would be labelled 1 as its weight denoted by w_{jk}, where k is the index of the class. In this case, if there is only one rule firing or more rules firing without conflict of classification, then the output would be deterministic. This is because there is only one node in the disjunction layer providing a weight greater than or equal to 1 for its connection to the node in the output layer. For all other nodes, the weight provided for the corresponding connection would be equal to 0.

However, as mentioned earlier, a rule may be inconsistent, which means that the same rule antecedent may be mapped to different classes as its consequent. In this case, the corresponding node would have multiple connections to different nodes in the disjunction layer. For each of the connections, the weight would be equal to a value between 0 and 1. Nevertheless, the sum of weights for the connections would be equal to 1. With regard to each of classes, it may be mapped from different rule antecedents. Therefore, each class would have a summative weight denoted by c_k, which is equal to the sum of the weights for the rule antecedents mapped to the class. Finally, the node in the output layer makes the weighted majority voting for the final prediction.

In the context of probabilistic logic, the t_{ij} would be equal to a value between 0 and 1 as a conditional probability. Similar to deterministic logic, a rule is firing if and only if all of the conditions are met. However, the rule antecedent would be assigned a firing probability computed in the corresponding node in the conjunction layer. The firing probability is simply equal to the product of the conditional probabilities for the rule terms (if corresponding attributes are independent) and also to the posterior probability of the rule consequent given the rule antecedent. If the rule is inconsistent, the sum of posterior

probabilities for the possible classes (w_{jk}) would also be equal to the firing probability above. This is because the rule consequent is the disjunction of the output terms, each of which has a different class as the output value. In the disjunction layer, each class is assigned a weight which is equal to the sum of its posterior probabilities given different rule antecedents. The final prediction is made by weighted majority voting in same way as based on deterministic logic.

In the context of fuzzy logic, in contrast to probabilistic logic, in the conjunction layer, the t_{ij} would be equal to a value between 0 and 1 as a fuzzy truth value for each corresponding condition. Similar to the other two types of logic, a rule is firing if and only if all of the conditions are met. However, the rule antecedent would be assigned a firing strength computed in the corresponding node in the conjunction layer. The firing strength is simply computed by choosing the minimum among the fuzzy truth values of the conditions (that are assumed independent). The fuzzy truth value for the rule consequent is equal to the firing strength. If the rule is inconsistent, the fuzzy truth value (w_{jk}) for having each possible class as the consequent would be derived by getting the minimum between the firing strength and the original fuzzy truth value assigned to this class for this rule. In the disjunction layer, the weight for each class is computed by getting the maximum among the fuzzy truth values (w_{jk}) of the rules having the class as the consequents. The final prediction is made by weighted majority voting in the same way as the above two types of logic.

Overall, the unified rule based network representation does not only show which input attributes are most significant for each class label in terms of determining the class label of a test instance, but also measure the corresponding degree of likelihood. It is important especially for fuzzy rule based systems required to assign a weight to each of the connections between nodes. This is because each of the connections is only involved in one rule in this representation. In contrast, decision tree representation may have the same connection shared by different rules with the need that different weights are assigned to the same connection for different rules, which results in confusions. In addition, if a linear list representation has each single rule term assigned a weight, it is likely to make the rules less readable. All above demonstrates a significant strength of using the unified network topology for knowledge discovery in real applications due to the presence of uncertainty.

V. Conclusion

This paper introduces newly developed techniques of rule representation which are network based. The variants of the network representation contribute to improvement on computational efficiency for predictive modelling (as illustrated in Table I) as well as model interpretability for knowledge discovery (as illustrated in Table II) in comparison with decision tree and linear list representations. In addition, this paper also introduces a generalized network topology for rule based systems based on any types of logic, as well as a specialized topology for deterministic rule based systems. However, fuzzy logic is popularly used in practice for constructing rule based systems. Therefore, some fuzzy logic based techniques, such as Type 1 or 2, will be investigated for potentially solving the issues relating to interpretability. On the other hand, the network topology illustrated in Fig.7 applies to any types of computational network such as a neural network, which has perceptron layers instead of conjunction and disjunction layers and each node represent a perceptron. The network topology can also represent a digital circuit, which has a number of computational layers and each node represent a logic gate such as AND, OR and NOT. Therefore, the network topology provides a general framework in computational intelligence and philosophical perspectives in complex systems and networks.

References

[1] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, "From data mining to knowledge discovery in databases," in AI Magazine, American Association for Artificial Intelligence, 1996, pp.37-54.

[2] F. Stahl and I. Jordanov. "An overview of use of neural networks for data mining tasks," WIREs: Data Mining and Knowledge Discovery, Wiley, 3 (2), pp. 193-208, 2012.

[3] P. N. Tan, M. Steinbach and V. Kumar, Introduction to Data Mining, New Jersey: Pearson Education, Inc, 2006.

[4] T. Mitchell, Machine Learning, McGraw Hill. pp.2, 1997.

[5] H. Liu, A. Gegov and F. Stahl, "Unified framework for construction of rule based classification systems," In: W. Pedrycz and S.M. Chen (eds) Information Granularity, Big Data and Computational Intelligence, Studies in Big Data 8, Springer, pp.209-230, 2015.

[6] C. M. Higgins, "Classification and approximation with rule based networks." PhD thesis. California Institute of Technology Pasadena, California, America, 1993.

[7] A. M. Uttley, "The design of conditional probability computers," Information and control, 2:1-24, 1959.

[8] I. Kononenko, "Bayesain neual networks," Biological Cybernetics, 61: 361-370, 1989.

[9] F. Rosenblatt, Principles of Neurodynamics: Perceptons and the Theory of Brain Mechnishms, Spartan Books, Washington, DC, 1962.

[10] O. Ekeberg and A. Lansner, "Automatic generation of internal representations in a probabilistic artifical neural network," Proceedings of the First European Conference on Neural Networks, June 6-9 1988.

[11] A. V. Aho, J. E. Hopcraft and J. D. Ullman, Data Structures and Algorithms. Retrieved at http://www.ourstillwaters.org/stillwaters/csteaching/DataStructuresAndAlgorithms/toc.htm. 2001.

[12] H. Liu, A. Gegov and F. Stahl, "Categorization and construction of rule based systems," In: 15th International Conference on Engineering Applications of Neural Networks, 5 - 7 September 2014, Sofia, Bulgaria, Springer, pp.183-194.

[13] J. Furnkranz, "Separate-and-Conquer rule learning," Artificial Intelligence Review 13: 3-54, Kluwer Academic Publishers, 1999.

[14] J. R. Quinlan, "C4.5: programs for machine learning," Morgan Kaufman, 1993.

[15] J. Cendrowska, "PRISM: an algorithm for inducing modular rules," International Journal of Man-Machine Studies 27 (1987) 349–370.

[16] X. Deng, "A covering-based algorithm for classification: PRISM." CS831: Knowledge Discover in Databases, 2012.

[17] A. Gegov, Complexity Management in Fuzzy Systems. Berlin: Springer, 2007.

[18] T. J. Ross, Fuzzy Logic with Engineering Applications (2nd Edition). John Wiley & Sons Ltd, West Sussex, 2004.

[19] S. G. Simpson, Mathematical Logic. Lecture Notes for Introductory Courses in Mathematical Logic. The Pennsylvania State University, University Park, State College, 2013.

[20] A. Holland, "Lecture 2: Rules based systems," Lecture Notes in Intelligent Systems, University Colleage Cork, 2010.