International Journal of Smart Sensor and Adhoc Network

Volume 1 | Issue 2 Article 20

October 2011

Efficient Rule Set Generation using Rough Set Theory for Classification of High Dimensional Data

Prasanta Gogoi
Department of Computer Science and Engineering, Tezpur University, Napaam, Tezpur, India 784028, prasant@tezu.ernet.in

Ranjan Das
Department of Computer Science and Engineering, Tezpur University, Napaam, Tezpur, India 784028, ranjan@tezu.ernet.in

B Borah
Department of Computer Science and Engineering, Tezpur University, Napaam, Tezpur, India 784028, bgb@tezu.ernet.in

D K. Bhattacharyya
Department of Computer Science and Engineering, Tezpur University, Napaam, Tezpur, India 784028, dkb@tezu.ernet.in

Recommended Citation
Gogoi, Prasanta; Das, Ranjan; Borah, B; and Bhattacharyya, D K. (2011) "Efficient Rule Set Generation using Rough Set Theory for Classification of High Dimensional Data," *International Journal of Smart Sensor and Adhoc Network*: Vol. 1 : Iss. 2 , Article 20.
DOI: 10.47893/IJSSAN.2011.1036
Available at: https://www.interscience.in/ijssan/vol1/iss2/20

Efficient Rule Set Generation using Rough Set Theory for Classification of High Dimensional Data

Prasanta Gogoi, Ranjan Das, B Borah & D K Bhattacharyya

Department of Computer Science and Engineering, Tezpur University, Napaam, Tezpur, India 784028
E-mail : {prasant, ranjan, bgb, dkb }@tezu.ernet.in

Abstract - In this paper, a rough set theory (RST) based approach is proposed to mine concise rules from inconsistent data. The approach deals with inconsistent data. At first, it computes the lower and upper approximation for each concept, then adopts a learning from an algorithm to build concise classification rules for each concept satisfying the given classification accuracy. Lower and upper approximation estimation is designed for the implementation, which substantially reduce the computational complexity of the algorithm. UCI ML Repository datasets are used to test and validate the proposed approach. We have also used our approach on network intrusion dataset captured using our local network from network flow. The results show that our approach produces effective and minimal rules and provide satisfactory accuracy over several real life datasets.

Keywords - *Rough Set; Inconsistency; Minimal; Redundant; Intrusion Data; PSC.*

I. INTRODUCTION

The rules are the prescribed standards on the basis of which decisions are made for specific purpose. The rule is a statement that establishes a principle or standard, and serves as a norm for guiding or mandating action or conduct. The rule can be a conditional statement that tells the system how to react to a particular situation. In data mining, the rule generation was first introduced by Agrawal et. al. in 1993 in terms of Market-basket analysis [1] as association rules. In association rules, the rule generation is based on the concept of frequent pattern mining for the discovery of interesting associations and correlations among itemset. Afterwards, methods were developed for classification rule mining [2]. The rule-based methods can be found in different applications of decision making and prediction like in the domain of medical research [4], in the areas of economics and finance [5] and in network security [6].

The cost of developing and maintaining rule sets is an important issue for the rule based systems. Based on the literature survey, it has been observed that three types of rule generation techniques are commonly found, viz., frequent association rule mining [7], rare association rule mining [8], and multi-objective rule mining [9].need to create these components, incorporating the applicable criteria that follow.

We observe that the rules generated by the above three approaches often are incapable of

- handling inconsistency in the database
- generating minimal rule set
- generating non-redundant rule set

In different real life or synthetic dataset, inconsistency is a common problem. Inconsistency is caused by the existence of indiscernbility relation in decision table. A data set is represented as a table, where each row represents an object or record. Every column represents an attribute that can be measured for each object. This table is a decision table. Attributes are of two categories: condition and decision. The indiscernbility relation occurs in a decision table if in objects of equivalent condition attributes, decision attributes are different. Consider a decision table with objects p_1, p_2, p_3 in *Table I.* Condition attributes are *A* and *C*, and decision attribute is *D*. The attributes of objects p_1 and p_3 are equivalent whereas their decision attributes are different. Here, objects p_1 and p_3 are indiscernible and the decision table has inconsistency. None of the previously mentioned techniques can provide any means to generate classification rules in these situations. In view of the above mentioned limitations, rough set theory (RST) is introduced for classification rule generation on inconsistent dataset. RST was first introduced by Pawlak [11] in the year 1982. RST is especially well suited to deal with inconsistencies [10]. One of the major advantages of RST is that it does not require any additional information on the data such as probability distribution or grade membership and it is capable of handling inconsistency.

TABLE I : INCONSISTENT DATASET

Objects	**Condition Attributes**		**Decision Attributes**
	A	***C***	***D***
P_1	*low*	*high*	*yes*
P_2	*low*	*low*	*no*
P_3	*low*	*high*	*no*

Followings are our contributions in this paper

- A method is proposed to find indiscernibility relation in data set to find inconsistency
- Determination of lower and upper approximation for inconsistent data.
- Minimized and non-redundant rule generation by using lower and upper approximation for classification.

The remainder of this paper is organized as follows. In the next section, we present related work on rule generation. In Section 3, we have given the proposed method of rule generation. Experimental results are presented in Section 4. In Section 5, we outline the conclusion and future work.

II. RELATED WORKS

A. Preliminaries of Rough Set

Rough set was proposed to classify imprecise and incomplete information. There have been contributions on applying rough sets theory (RST) to rule discovery. In [19], RST was used on clusters to determine rules for association explanations. Adetunmbi [20] used rough sets to data that contain the minimal subset of attributes associated with a class label for classification. RST can help to determine whether there is redundant information in the data to gather the essential data needed for applications. The RST based rule generation approach can be able to generate minimal and non-redundant rule set in inconsistent data.

B. Rough Set

RST is an approach to vagueness. It is an extension of the classical set theory, for use when representing vagueness i.e. imprecision. Rough set is expressed by a boundary region of a set [11]. The basic concept of the RST is the notion of approximation space, which is an ordered pair *I = (U, R)*, where

- *I*: information system
- *U*: nonempty set of objects, called universe
- *R:* equivalence relation on *U*, called indiscernibility relation. If $x, y \in U$ and *xRy* then *x* and *y* are indistinguishable in *I.*

Each equivalence class induced by *R*, is called an elementary set in *A* and represented as *U/R*. A definable set in *I* is any finite union of elementary sets in *I*. For $x \in U$, let $[X]_R$ denote the equivalence class of *R*, containing *x*. For each $X \subseteq U$, *X* is characterized in *I* by a pair of sets- its lower and upper approximation in *I*, defined respectively as:

$$\underline{R}X = \{x \in U \mid [X]_R \subseteq X\}$$
$$\overline{R}X = \{x \in U \mid [X]_R \cap X \neq \varnothing\} \quad (1)$$

A rough set in *I* is all subsets of *U* having the same lower and upper approximations. Reduct and core are two related concepts in RST.

<u>*Reduct*</u>: A reduct [11] is a set of attributes that preserves partition. It means that a reduct is the minimal subset of attributes that enables the same classification of elements of the universe as the whole set of attributes.

In order to express the idea of reduct, let $B \subseteq A$ and $a \in B$ in an information system *I = (U, A)* where U is the universe of objects, A is set of attributes, and *R(B)* is a binary relation.

- *a* is dispensable in *B* if *R(B) = R(B –{a})*; otherwise *a* is indispensable in *B*
- Set *B* is independent if all its attributes are indispensable.
- $B' \subseteq B$ is a reduct of *B* if B' is independent and $R(B') = R(B)$

The attributes other than the reduct are redundant attributes. The removal of redundant attributes cannot deteriorate the classification. Usually, there are several reducts in a dataset.

Core : The core [11] is the set of all indispensable attributes, i.e., it is the intersection of all reducts.

Let *Red(B)* is the set of all reducts of *B* in an information system *I = (U, A)* where $B \subseteq A$ then the core of *B* is defined as

$$core(B) = \bigcap \mathrm{Re}\,d(B) \quad (2)$$

The core is included in every reduct, i.e., each element of the core belongs to some reduct. Thus, the core is the most important subset of attributes, for none of its elements can be removed without affecting the classification.

C. HCRI Algorithm

It is a heuristic algorithm for mining concise rules from inconsistent data (HCRI [3]). This method is based on the variable precision rough set model. It deals with inconsistent data to mine concise rules. It first computes the reduct for each concept, then computes the reduct for each object. It adopts a heuristic method to build concise classification rules for each concept. To compute the equivalence classes, it uses two hash functions, which substantially reduce the complexity to $O(n)$, $n = |U|$. The hash functions compute the cardinality of lower approximation. The input to the method is a set of inconsistent objects U and the output is a set of concise rules satisfying a given classification accuracy.

D. LEM2 Algorithm

LEM2 [21] (Learning by Example Module, Version - 2) is a machine learning algorithm based on rough set theory. The usual task of LEM2 algorithm is to learn a discriminate rule set, i.e., to learn the smallest set of minimal rules, describing the concept. This algorithm can generate both certain and possible rules from a decision table with attributes being numerical as well categorical. LEM2 needs discretization for numerical attributes.

For inconsistent data, LEM2 induces two sets of rules: certain rule set and possible rule set. The first set is computed from lower approximations of concepts, the second one from upper approximations. It is assumed that the rule set will be used automatically by a classification component. Nevertheless, induced rules are available and comprehensible by the user. Thus, it is possible to use rules manually, like in other systems.

The LEM2 algorithm is a single local covering approach. It yields single minimal discriminate description, which means, learning the smallest set of minimal rules for every concept. The local coverings are constructed from minimal complex. A formal definition of minimal complex and local covering is reported next from [11].

Definition: Minimal complex and Local covering Let B be a nonempty lower or upper approximation of a concept represented by a decision-value pair *(d,w)*. The set T is a minimal complex of B if and only if B depends on T and no proper subset T' of T exists such that B depends on T'. Let $\Im$ be a collection of non-empty set of attribute-value pairs for equivalence class *[T]* of T. Then $\Im$ is the local covering of B iff

- each member T of $\Im$ is a minimal complex of B
- $\bigcup_{T\in\Im}[T] = B$ and
- $\Im$ is minimal i.e., $\Im$ has the smallest possible number of members.

LEM2 algorithm is found suitable in rule generation for inconsistent data. In the next section, we propose a LEM2 based technique for rule generation.

III. PROPOSED WORK

We have proposed an effective rule generation technique using RST based on LEM2 algorithm. The proposed method can be found to be significant especially for those datasets having inconsistencies. Our method starts with the inconsistency checking for each concept in the dataset. If it finds the inconsistency, then computes the upper approximation and the lower approximation. To compute the inconsistency and to find the upper and lower approximations it introduces the following method to support the LEM2 based rule generation technique.

A. Evaluation of Upper & Lower Approximation

For computation of the upper and lower approximation for each concept of dataset, it executes the steps given below.

Algorithm: *Compute CLU*

1. Identify the set of concepts Γ .
2. Take an arbitrary object c from a concept $C_i \in \Gamma$ and make a comparison of each attribute-value pair with all the objects c' of another concept $C'_j \in \Gamma$. If attribute-value pairs of object c and object c' are matching, then inconsistency occurs with respect to the concepts C_i and C'_j .
    a. Search any other inconsistent pair of objects of the concepts (C_i , C'_j).
    b. Generate U_{approx} by taking union of the set of objects $\{C_i\}$ of concept $C_i \in \Gamma$ with the inconsistent pairs of objects of concept (C_i , C'_j) i.e., $U_{approx} = \{C_i\} \bigcup \{C'_j\}$.
    c. Generate L_{approx} by subtracting C'_j from C_i i.e., $L_{approx} = \{C_i\} - \{C'_j\}$.
3. Otherwise objects are consistent.

B. *Proposed Method*

The proposed rule generation approach is based on LEM2 algorithm. LEM2 is a single local covering approach and it yields single minimal discriminate description. In LEM2, the user may or may not consider any attribute priority. However, in contrary to LEM2, the proposed rule generation approach considers

- the attribute priority, and
- extracts the output of the method to *Compute CLU*

Let β is upper or lower approximation of a concept or a concept itself and B is a members of β. $\Im$ is a single local covering for the set β, i.e., it yields the smallest set of minimum rules for the entire set β. G is a temporary storage of B. T is a set of attribute-value pairs. t is a member of T, i.e. $t \in T$. *[t]* is a equivalence class of t, i.e., the set of all objects which have the attribute-value pair t. *T(G)* is a set of attribute-value pairs which are present in objects of G, i.e., $T(G) := \{t \mid [t] \cap G \neq Null\}$. *[T –{t}]* is a set of objects which have the attribute-value pairs other than t. S is a member of $\Im$ other than T, i.e., $S \in \Im - \{T\}$.

<u>*Procedure*</u>

```
input : a set $\beta$
output: a single, local covering $\Im$ of set $\beta$;
begin {Procedure}
  while($\beta \neq$ Null )
  begin
    for each concept,
      if found inconsistency then
          $L_{approx}$ and $U_{approx}$ will be the member of $\beta$
      else
          the concept will be the member of $\beta$.
    for each $B \in \beta$ do
    begin
      $G := B;$
      $\Im := Null;$
      while ($G \neq Null$)
      begin
     $T := Null;$
        $T(G) := \{t \mid [t] \cap G \neq Null\};$
        while ($T = Null$) or ($[T] \not\subseteq B$)
        begin
          Select an attribute-value pair $t \in T(G)$ with the
          highest attribute priority, if a tie occurs, select a
          $t \in T(G)$ such that $|t \cap G|$ is maximum; if
          another tie occurs, select a $t \in T(G)$ with the
          smallest cardinality of $[t]$; if further tie occurs,
          select the first pair;
           $T := T \cup \{t\};$
           $G := [t] \cap G;$
              $T(G) := \{t \mid [t] \cap G \neq Null\};$
              $T(G) := T(G) - T;$
           end{while}
             for each $t$ in $T$ do
             if $[T - \{t\}] \subseteq B$ then $T := T - \{t\};$
                    $\Im := \Im \cup \{T\};$
                    $G := B - \bigcup_{T \in \Im} [T];$
          end{while}
        end{while}
        for $T$ in $\Im$ do
        if $\bigcup_{S \in \Im - \{T\}} [S] = B$ then $\Im := \Im - \{T\};$
     end{Procedure}
```

C. *Complexity Analysis*

Let n be the total number of objects in our sample dataset. Now, in order to verify the inconsistency, we have to compare each individual object with every other objects present in our dataset. So, the complexity of the computation of upper and lower approximation is $O(n^2)$. LEM2 has the complexity of $O(nm)$ where n is the number of objects and m is the number of attributes. The complexity of our proposed algorithm is $O(n^2) + O(nm)$.

Our algorithm expects that the sample dataset to have inconsistency. The inconsistencies may arise in only some of the concepts (not all). So, our algorithm initially compute the U_{approx} and L_{approx} for those concepts only. The concepts which do not have inconsistency will fed to the program without finding upper and lower approximation. The upper and lower approximation will be computed before execution of the main procedure.

IV. EXPERIMENTAL RESULTS

All the necessary experiments were carried out in a workstation having the configuration of Intel core 2 Quad @2.4GHz, 2 GB RAM, 160GB HDD. The procedures are executed in Linux environment with *C* compiler.

The accuracy of each experiment was measured based on percentage of successful classification *(PSC)* [20] on the evaluated dataset, where

$$PSC = \frac{\text{No. of Correctly Classified Instances}}{\text{No. of Instances in Dataset}} \times 100 \qquad (3)$$

A. Results on UCI Dataset

The proposed method was tested on several real life datasets from UCI Machine Learning Repository dataset [22] and also the one given in [10]. We implemented our proposed method using *C*. The results of the experiments are reported in *Table V*. It can be observed from the table that it performs consistently well for categorical dataset. Since, the method has been specially designed for handling inconsistency in the dataset, it expects the occurrence of at least some inconsistencies in the dataset. Another important advantage of the method is its input order independency. As can be observed from *Table V* that for the UCI Machine Learning Repository datasets like mushroom, glass identification, breast cancer etc, the algorithm has been able to generate rules which classify with more than *90%* accuracy. The example of the generated rules for different UCI dataset are reported in *Table II.* An interesting observation is that, number of rule generation is not dependent on the number of instances in the dataset. For example, as can be seen from the *Table V* that Mushroom [22] dataset has the maximum number of instances, i.e., 8124, however, the number of rule generated (as can be seen from *Table V*) for this dataset is not maximum. But with the increase in the dimensionality, the number of rules generated also increases as given in *Table V*, as it is evident for the case of Soyabean-Small [22] dataset as shown in *Table V*.

TABLE II : SAMPLE RULES FOR UCI DATASET

Data set	Rules
Demo [10]	*(Hemoglobin, fair) ^ (Temperature, low) → (Comfort, low)*
	(Blood_ Pressure, high) → (Comfort, very low)
Breast cancer [22]	*(Fractal Dimension, 2) ^ (Concavity, 2) → (Diagonis, 2)*
	(Fractal_ Dimensio, 2) ^ (Summary, 8) → (Diagonis, 4)
Mushroom [22]	*(stalk_ color_ above_ rin, c) ^ (stalk_ root, b) → (habitat, m)*
	(geil_color, f) ^ (ring_ type, g) → (habitat, g)
Glass [22]	*(Aluminium, 1.32) → (type_ of_ glass, building_ windows_ float_ processed)*
	(Potassium, 0.00) ^ (Iron, 0.00) → (type_ of_ glass, tableware)

B. Result on Real Life Network Intrusion Dataset

The proposed method was also evaluated using our own dataset that include various type of features extracted based on net-flow data captured using our local network. Using some of the existing attack tools, we generated a group of attacks against a local network server and collected the produced traffic as known attack traffic. The existing attacks are generated using tools found in [23].

A flow is a unidirectional series of IP (internet protocol) packets passing through an observation point in the network during a certain time interval. All packets belonging to a particular flow have a set of common properties. *NetFlow* is a network protocol based collection of summarized IP traffic information. We used open source collecting tool *nfdump* for receiving the exported flow records from network devices. For gathering the normal traffic, we recorded samples of the usual traffic of the network within 4 weeks period. Thus, we did collection of 1,48,712 net-flow records of 16 attack types and normal records. The extracted net-flow level features are reported in *Table VI.*

The results of the net-flow intrusion dataset is given in *Table III*. The detection performance of the method over net-flow intrusion dataset is well. The *PSC* of net-flow intrusion dataset, in case of normal class, is found as *99.94%* whereas for all-attacks class, it is *96.21%.* Examples of the generated rules for net-flow intrusion dataset are given in *Table IV*.

TABLE III : RESULTS OVER NET-FLOW INTRUSION DATASET

Attacks	**Generation Tool [23]**	**Sizes**	**Detection Sizes**	**Accuracy (PSC)**
bonk	*targa2.c*	*13000*	*12562*	
jolt	*targa2.c*	*1394*	*1374*	
nestea	*targa2.c*	*92*	*92*	
newtear	*targa2.c*	*137*	*136*	
syndrop	*targa2.c*	*66*	*65*	
teardrop	*targa2.c*	*130*	*130*	
winnuke	*targa2.c*	*12000*	*11559*	
1234	*targa2.c*	*30000*	*28929*	
oshare	*targa2:c*	*12000*	*11672*	
saihyousen	*targa2.c*	*252*	*251*	
smurf	*smurf4.c*	*30*	*30*	
fraggle	*Fraggle.c*	*12000*	*10784*	
syn	*Nmap*	*8000*	*7598*	
xmas	*Nmap*	*13000*	*12942*	
window	*targa2.c*	*14000*	*13565*	
land	*targa2.c*	*2*	*2*	
All attacks		*116096*	*111691*	*96.21%*
normal		*32616*	*32598*	*99.94%*

TABLE IV : SAMPLE RULES FOR NET-FLOW INTRUSION DATASET

Sl. No	**Rules**
1	*(protocol, ICMP) ^ (source port, 0) → (class; 1234)*
2	*(protocol, UDP) ^ (RST, 0) → (class, bonk)*
3	*(sourc_ IP, 172.16.15.12) ^ (protocol, UDP) → (class, fraggle)*
4	*(count_ serv_ dest, 1385) → (class, jolt)*
5	*(protocol, UDP) ^ (URG, 1) → (class, nestea)*

V. CONCLUSION AND FUTURE WORKS

This work proposes a classification rule generation method based on LEM2 algorithm. The proposed method is typically employable in those datasets which have inconsistencies. The method has been found to exhibit satisfactory performance whenever the dataset contains inconsistencies at least for some concepts. We have tested our rule generation method on several, real life UCI machine learning repository datasets for the classification and the results have been found satisfactory. The experimental results discussed in the earlier section demonstrate the effectiveness of the proposed method.

The method covers only the local covering option. For every concept, it generates a minimum, non-redundant set of classification rules. However, the method is silent to address the generation of minimum, non-redundant classification rule set collectively over the whole dataset, that is global covering. There are scopes to consider the global covering option as well. It might have yield better results if we go for a mixed approach that local as well as global coverings. We are working towards LEM2 algorithm based minimal rule generation for other network intrusion datasets.

REFERENCES

[1] Srikant, R., Vu, Q., and Agrawal, R. (1997) Mining association rules with item constraints. Proc. of the 3rd International Conference on Knowledge Discovery and Data Mining, California USA, August, pp. 67-73. AAAI Press.

[2] Han, J. and Kamber, M. (2001) Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, 500 Sansome Street, Suite 400, San Francisco, CA 94111.

[3] Sai, Y., Nie, P., Xu, R., and Huang, J. (2006) A rough set approach to mining concise rules from inconsistent data. Proc of IEEE GrC 2006, Atlanta USA, May 10-12, pp. 333-336. IEEE.

[4] J.W., J. P. and J.W., R. B. (2002) Rule generation and model selection used for medical diagnosis. Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology - Challenges for future intelligent systems in biomedicine, 12(1), 69-78.

[5] Grosan, C. and Abraham, A. (2006) Stock market modeling using genetic programming ensembles. Genetic Systems Programming: Theory and Experiences, 13, 133-148.

[6] Vollmer, T., Foss, J. A., and Manic, M. (2011) Autonomous rule creation for intrusion detection. Proc. of SCCI 2011, Paris, France, Apr.11-15, 2011, pp. 1-8. IEEE.

[7] Agrawal, R., Imielinski, T., and Swami, A. (1993) Mining association rules between sets of items in large databases. Proc of 1993 ACM SIGMOD, Washington, DC, USA, May 25-28, pp. 207-216. ACM, New York, NY, USA.

[8] Kiran, R. U. and Reddy, P. K. (2010) Mining rare association rules in the datasets with widely varying items' frequencies. Lecture Notes in Computer Science, 5981/2010, 49-62.

[9] Ghosh, A. and Nath, B. T. (2004) Multi-objective rule mining using genetic algorithms. Information Sciences: an International Journal, 163, 123-133.

[10] Slowinski, R. (1992) In Intellegent Decision Support: Handbook of Applications and Advances of the Rough Set Theory. Kluwer Academic Publishers Norwell, MA, USA.

[11] Pawlak, Z., Grazymala-Busse, J. W., Slowinski, R., and Ziarko, W. (1995) Rough sets. Communications of the ACM, 38, 88-95.

[12] Fernandez, M. C., Menasalvas, E., scar Marban, Pena, J. M., and Millan, S. (2001) Minimal decision rules based on the apriori algorithm. International Journal of Applied Mathematics & Computer Science, 11, 691-704.

[13] Grzymala-Busse, J. W. (1997) A new version of the rule induction system lers. Fundamenta Informaticae, 31, 27-39.

[14] Shichao, Z. and Xindong, W. (2011) Fundamentals of association rules in data mining and knowledge discovery. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 1, 97-116.

[15] Agrawal, R. and Srikant, R. (1994) Fast algorithms for mining association rules in large databases. In Bocca, J. B., Jarke, M., and Zaniolo, C. (eds.), Proc of VLDB'94, Santiago de Chile, Chile, September 12-15, pp. 487-499. Morgan Kaufmann.

[16] Lin, D. and Kedem, Z. (1998) Pincer-search: A new algorithm for discovering the maximum frequent set. Proc of EDBT '98, Valencia, Spain, March 23-27, pp. 105-119. Springer-Verlag London, UK.

[17] Han, J., Pei, J., and Yin, Y. (2000) Mining frequent patterns without candidate generation. Proc of ACM SIGMOD '00, NY, USA, May 14-19, pp. 1-12. ACM New York.

[18] Qodmanan, H. R., Nasiri, M., and Minaei-Bidgoli, B. (2011) Multi objective association rule mining with genetic algorithm without specifying minimum support and minimum con_dence. Expert Systems with Applications, 38, 288-298.

[19] Li, J. and Cercone, N. (2005) A rough set based model to rank the importance of association rules. Lecture Notes in Computer Science, 3642/2005, 109-118.

[20] Adetunmbi, A. O., Falaki, S. O., Adewale, O. S., and Alese, B. K. (2008) Network intrusion detection based on rough set and k-nearest neighbour. International Journal of Computing and ICT Research, 2, 60-66.

[21] Grzymala-Busse, J. W. (1988) Knowledge acquisition under uncertainty - a rough set approach. Journal of Intelligent & Robotic Systems, 1, 3-16.

[22] Blake, C. L. and Merz, C. J. (2001). UCI Machine Learning Repository. Irvine, CA: University of California, Department of Information and Computer Science, http://www.ics.uci. edu / ~ mlearn/MLRepository.html.

[23] (2003). Attacks tools and information. http://packetstormsecurity.nl/index.html.

[24] Szathmary, L., Valtchev, P., and Napoli, A. (2010) Generating rare association rules using the minimal rare itemsets family. International Journal of Software Informatics, 4, 219-238.

.

TABLE V : EXPERIMENTAL RESULTS

Data set	**Data Types**	**Instance Sizes**	**Attribute Sizes**	**Training set Sizes**	**Test set Sizes**	**Rules Generated Sizes**	**Accuracy (PSC)**
Demo [10]	Categorical	9	6	9	9	7	100%
Breast cancer [22]	Categorical	286	10	500	149	17	97.32%
Congressional Voting [22]	Categorical	435	17	300	135	14	98.52%
Mushroom [22]	Categorical	8124	23	450	150	18	92.67%

Data set	**Data Types**	**Instance Sizes**	**Attribute Sizes**	**Training set Sizes**	**Test set Sizes**	**Rules Generated Sizes**	**Accuracy (PSC)**
Glass [22]	Numeric	214	10	144	70	13	85.71%
Iris [22]	Numeric	150	4	150	150	18	93.5%
New-Thyroid [22]	Numeric	215	6	215	215	49	86.5%
Ballon [22]	Categorical	20	5	20	20	7	98%
Hayes-roth [22]	Categorical	160	6	160	160	50	95.5%
Soyabean-Small [22]	Categorical	. 307	.35	.625	.625	.80	.91.7%
Balance-scale [22]	Categorical	625	.5	.625	.625	.50	.94.8%

TABLE VI : FEATURES OF OUR NET-FLOW INTRUSION DATASET

Basic features		**Basic features**	
Features	***Description***	***Features***	***Description***
Duration	Length of the flow (in seconds)	Source byte	Number of data bytes transfer from source IP to destination IP
Protocol type	Type of protocols e.g. TCP, UDP, ICMP	Land	Same source IP/source port are equal to Destination IP/Destination port
Src IP	Source node IP address	**Time-window features**	
Dest IP	Destination IP address	count-dest	Number of flows to unique destination IP addresses inside the network in the last *T* seconds from the same source
Src port	Source port	count-src	Number of flows from unique source IP addresses inside the network in the last *T* seconds to the same destination
Dest port	Destination port	count-serv- src	Number of flows from the source IP to the same destination port in the last *T* seconds.
ToS	Type of service	count-serv-dest	Number of flows to the destination IP using same source port in the last *T* seconds.
URG	Urgent flag of TCP header	**Connection based features**	
ACK	Acknowledgement flag	count-dest-conn.	Number of flows to unique destination IP addresses inside the network in the last *N* flows from the same source
PSH	Push flag	count-src-conn	Number of flows from unique source IP addresses inside the network in the last *N* flows to the same destination
RST	Reset flag	count-serv-src-conn	Number of flows from the source IP to the same destination port in the last *N* flows.
SYN	SYN flag	count-serv-dest-conn	Number of flows to the destination IP using same source port in the last *N* flows.
FIN	FIN flag		

◈◈◈

.