Evaluation of Classification Techniques Using MapReduce for Meta-Path Classification in Heterogeneous Information Networks

[1] Mrs. V. Vetriselvi, [2] V. Prahatha

[1]Associate Professor, [2]Research Scholar

[1,2]Department of Computer Science

[1,2]Shrimati Indira Gandhi College, Tiruchirappalli, Tamil Nadu, India-620002

Abstract: Classification of the nodes along with the interconnected semantic edges in a Heterogeneous Information Network (HIN) has a lot of significance in identifying the class labels which involves the application of knowledge and dissemination of knowledge from one node to the other. In this research work, PathSim similarity measure - Host based Intrusion Detection system with Proposed PFSM and Proposed IRGM gives minimum number of rules for detecting the unknown attacks in the network has applied for finding classification techniques along with the use of the well-known MapReduce paradigm to classify the meta-paths in a Heterogeneous Information Network. Applying MapReduce simplified the classification approach which deals with huge data present in the Heterogeneous Information Networks. Experiments were carried out on DARPA dataset, and the results are accurate and successful.

I. Introduction

Collective classification methods [6] aim at exploiting the label autocorrelation among a group of inter-connected instances and predict their class labels collectively, instead of independently. The dependencies among the related instances should be considered explicitly during classification process. Most approaches in collective classification focus on exploiting the dependencies among interconnected objects in homogeneous networks. However, many real-world applications are facing large scale heterogeneous information networks [5] with multiple types of objects inter-connected through multiple types links. These networks are multimode and multi-relational networks, which involves large amount of information. For example, consider a bibliographic network with five types of nodes (papers, author, affiliations, conference and proceedings) and five types of links.

Multi-Mode and Multi-Relational Data: One fundamental problem in classifying heterogeneous information networks is that the network structure involves multiple types of nodes and multiple types of links. For example, in Figure 1, one paper node can be linked directly with different types of objects, such as authors, conference proceedings and other papers, through different types of links, such as citation, authored By, etc. Trivial application of conventional methods by ignoring the link types and node types cannot fully exploit the structural information within a heterogeneous information network.

Heterogeneous Dependencies: Another problem is that objects in heterogeneous information networks can be linked indirectly through different types of relational paths. Each types of relational path corresponds to different types of indirect relationships between objects. For example, paper nodes can be linked with each other indirectly through multiple indirect relationships, such as, 1) the “paperauthor-paper” relation indicates relationships of two papers sharing same authors; 2) the “paper-author-institute-authorpaper” relation denotes relationship between papers that are published from the same institute. Heterogenous information networks can encode various complex relationships among different objects. Thus, ignoring or treating all relations equally will loss information dependence information in a heterogeneous information network [17][18][19][20][21].

II. LITERATURE REVIEW

Li, Ji-chao, et al. "**A link prediction method for heterogeneous networks based on BP neural network**." *Physica A: Statistical Mechanics and its Applications* 495 (2018): 1-17.

In this paper, we put forward a novel integrated framework, called MPBP (Meta-Path feature-based BP neural network model), to predict multiple types of links for heterogeneous networks. More specifically, the concept of meta-path is introduced, followed by the extraction of meta-path features for heterogeneous networks. Next, based on the extracted meta-path features, a supervised link prediction model is built with a three-layer BP neural network. Then, the solution algorithm of the proposed link prediction model is put forward to obtain predicted results by iteratively training the network. Last, numerical experiments on the dataset of real examples of a gene-disease network and a combat network are conducted to verify the eectiveness and feasibility of the proposed MPBP. It shows that the MPBP with very good performance is superior to the baseline methods.

Pio, Gianvito, et al. "**Multi-type clustering and classification from heterogeneous networks**." *Information Sciences* 425 (2018): 107-126

in this paper we propose the algorithm HENPC, which is able to work on heterogeneous networks with an arbitrary structure. In particular, it extracts possibly overlapping and hierarchically-organized heterogeneous clusters and exploits them for predictive purposes. The different levels of the hierarchy which are discovered in the clustering step give us the opportunity to choose either more globally-based or more locally-based predictions, as well as to take into account autocorrelation phenomena at different levels of granularity. Experiments on real data show that HENPC is able to significantly outperform competitor approaches, both in terms of clustering quality and in terms of classification accuracy.

Liang, Wenxin, et al. "**Supervised ranking framework for relationship prediction in heterogeneous information networks**." *Applied Intelligence* 48.5 (2018): 1111-1127.

In recent years, relationship prediction in heterogeneous information networks (HINs) has become an active topic. The most essential part of this task is how to effectively represent and utilize the important three kinds of information hidden in connections of the network, namely local structure information (Local-info), global structure information (Global-info) and attribute information (Attrinfo). Although all the information indicates different features of the network and influence relationship creation in a complementary way, existing approaches utilize them separately or in a partially combined way. In this article, a novel framework named Supervised Ranking framework (S-Rank) is proposed to tackle this issue. To avoid the class imbalance problem, in S-Rank framework we treat the relationship prediction problem as a ranking task and divide it into three phases. Firstly, a Supervised PageRank strategy (SPR) is proposed to rank the candidate nodes according to Global-info and Attr-info. Secondly, a Meta Path-based Ranking method (MPR) utilizing Local-info is proposed to rank the candidate nodes based on their meta path-based features. Finally, the two ranking scores are linearly integrated into the final ranking result which combines all the Attr-info, Global-info and Local-info together. Experiments on DBLP data demonstrate that the proposed S-Rank framework can effectively take advantage of all the three kinds of information for relationship prediction over HINs and outperforms other well-known baseline approaches.

Ahmad, Awais, et al. "**Toward modeling and optimization of features selection in Big Data based social Internet of Things**." *Future Generation Computer Systems* 82 (2018): 715-726.

this paper presents a system architecture that selects features by using Artificial Bee Colony (ABC). Moreover, a Kalman filter is used in Hadoop ecosystem that is used for removal of noise. Furthermore, traditional MapReduce with ABC is used that enhance the processing efficiency. Moreover, a complete four-tier architecture is also proposed that efficiently aggregate the data, eliminate unnecessary data, and analyze the data by the proposed Hadoop-based ABC algorithm. To check the efficiency of the proposed algorithms exploited in the proposed system architecture, we have implemented our proposed system using Hadoop and MapReduce with the ABC algorithm. ABC algorithm is used to select features, whereas, MapReduce is supported by a parallel algorithm that efficiently processes a huge volume of data sets. The system is implemented using MapReduce tool at the top of the Hadoop parallel nodes with near real-time.

Zhang, Daokun, et al. "**MetaGraph2Vec: Complex Semantic Path Augmented Heterogeneous Network Embedding**." *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, Cham, 2018.

Network embedding in heterogeneous information networks (HINs) is a challenging task, due to complications of different node types and rich relationships between nodes. As a result, conventional network embedding techniques cannot work on such HINs. Recently, metapathbased approaches have been proposed to characterize relationships in HINs, but they are ineffective in capturing rich contexts and semantics between nodes for embedding learning, mainly because (1) metapath is a rather strict single path node-node relationship descriptor, which is unable to accommodate variance in relationships, and (2) only a small portion of paths can match the metapath, resulting in sparse context information for embedding learning. In this paper, we advocate a new metagraph concept to capture richer structural contexts and semantics between distant nodes. A metagraph contains multiple paths between nodes, each describing one type of relationships, so the augmentation of multiple metapaths provides an effective way to capture rich contexts and semantic relations between nodes. This greatly boosts the ability of metapath-based embedding techniques in handling very sparse HINs. We propose a new embedding learning algorithm, namely MetaGraph2Vec, which uses metagraph to guide the generation of random walks and to learn latent embedding of multi-typed HIN nodes. Experimental results show that MetaGraph2Vec is able to outperform the state-of-the-art baselines in various heterogeneous network mining tasks such as node classification, node clustering, and similarity search.

III. FEATURE SELECTION TECHNIQUES

The majority of real-world classification problems require supervised learning where the underlying class probabilities and class-conditional probabilities are unknown, and each instance is associated with a class label. In real-world situations, relevant features are often unknown a priori. Therefore, many candidate features are introduced to better represent the domain. Unfortunately, many of these are either partially or completely irrelevant/redundant to the target concept. A relevant feature is neither irrelevant nor redundant to the target concept; an irrelevant feature does not affect the target concept in any way, and a redundant feature does not add anything new to the target concept. In many applications, the size of a dataset is so large that learning might not work as well before removing these unwanted features. Reducing the number of irrelevant/redundant features drastically reduces the running time of a learning algorithm and yields a more general concept. This helps in getting a better insight into the underlying concept of a real-world classification problem. Feature selection methods try to pick a subset of features that are relevant to the target concept [14][15][16].

Feature selection is defined by many authors by looking at it from various angles. But as expected, many of those are similar in intuition and/or content. The following lists those that are conceptually different and cover a range of definitions.

- Idealized: find the minimally sized feature subset that is necessary and sufficient to the target concept.
- Classical: select a subset of M features from a set of N features, M<N, such that the value of a criterion function is optimized over all subsets of size M.
- Improving Prediction accuracy: the aim of feature selection is to choose a subset of features for improving prediction accuracy or decreasing the size of the structure without significantly decreasing prediction accuracy of the classifier built using only the selected features.
- Approximating original class distribution: the goal of feature selection is to select a small subset such that the resulting class distribution, given only the values for the selected features, is as close as possible to the original class distribution given all feature values.

Notice that the third definition emphasizes the prediction accuracy of a classifier, built using only the selected features, whereas the last definition emphasizes the class distribution given the training dataset. These two are quite different conceptually. Hence, our definition considers both factors. Feature selection attempts to select the minimally sized subset of features according to the following criteria. The criteria can be:

1. the classification accuracy does not significantly decrease; and
2. the resulting class distribution, given only the values for the selected features, is as close as possible to the original class distribution, given all features.

Ideally, feature selection methods search through the subsets of features, and try to find the best one among the competing 2N candidate subsets according to some evaluation function. However this procedure is exhaustive as it tries to find only the best one. It may be too costly and practically prohibitive, even for a medium-sized feature set size (N). Other methods based on heuristic or random search methods attempt to reduce computational complexity by compromising performance. These methods need a stopping criterion to prevent an exhaustive search of subsets. In our opinion, there are four basic steps in a typical feature selection method.

- a generation procedure to generate the next candidate subset;
- an evaluation function to evaluate the subset under examination;
- a stopping criterion to decide when to stop; and
- a validation procedure to check whether the subset is valid.

The generation procedure is a search procedure [46,26]. Basically, it generates subsets of features for evaluation. The generation procedure can start: (i) with no features, (ii) with all features, or (iii) with a random subset of features. In the first two cases, features are iteratively added or removed, whereas in the last case, features are either iteratively added or removed or produced randomly thereafter [26]. An evaluation function measures the goodness of a subset produced by some generation procedure, and this value is compared with the previous best. If it is found to be better, then it replaces the previous best subset. Without a suitable stopping criterion the feature selection process may run exhaustively or forever through the space of subsets. Generation procedures and evaluation functions can influence the choice for a stopping criterion. Stopping criteria based on a generation procedure include: (i) whether a predefined number of features are selected, and (ii) whether a predefined number of iterations reached. Stopping criteria based on an evaluation function can be: (i) whether addition (or deletion) of any feature does not produce a better subset; and (ii) whether an optimal subset according to some evaluation function is obtained. The loop continues until some stopping criterion is satisfied. The feature selection process halts by outputting a selected subset of features to a validation procedure. There are many variations to this three-step feature selection process, which are discussed in Section 3. The validation procedure is not a part of the feature selection process itself, but a feature selection method (in practice) must be validated. It tries to test the validity of the selected subset by carrying out different tests, and comparing the results with previously established results, or with the results of competing feature selection methods using artificial datasets, real-world datasets, or both.

IV. CLASSIFICATION TECHNIQUES

The Classification is the task of generalizing known structure to apply to new data. For example, an email program might attempt to classify an email as legitimate or spam. Common algorithms include

- Decision Tree,
- K-Nearest Neighbor,
- Support Vector Machines,
- Naive Bayesian Classification,
- Neural Networks

V. PROPOSED FRAMEWORK FOR HETEROGENOUS INFORMATION SYSTEM

Phase 1: Pre-Processing: In this phase, Standardization is exploited to competent the classification exactness in the sorting stage [9].

Phase 2: Feature Selection: In this phase, a filter feature selection method and Classifier are mongrelized to acquire the durable significant features with determined classification accuracy and least error rates. The algorithm suggested in this phase is created as the Precocious feature selection method. This algorithm picks the features established on the exactness by the classification technique ANN [9].

Phase 3: Classification Stage: In this phase, Artificial Neural Network has employed to categorize the information into three classes Secure, Known attacks and Unknown attacks [9].

Phase 4: Rule Generation Method: In this phase, proposed Intellectual rule generation which is the combination of ARM and MapReduce has used to generate the attack rule structure for unknown attacks in the networks.

Current association rule algorithms need to be redesigned to handle huge data problems to perform an efficient evaluation of the objectives and keep the quality of the rules obtained simultaneously. To accomplish that, an intellectual rule generation method has proposed that follows a MapReduce design to discover Association Rules from different proportions of the data [10]. The challenge is how to discover quality association rules that represent the complete dataset through subset of rules obtained in different splits of the dataset. To fulfil this goal, the association rule algorithm is only executed in a subset (Map), thus, the fitness function evaluates only a subset of instances. The evolutionary process for each Map ends when a number of evaluations is reached (N_{eval}) and a set of ARs is returned (RuleSet) [16].

Once all Maps are processed by the rule mining algorithm, the RuleSet from each Map is collected. Then, the global quality of the ARs of each RuleSet is evaluated using the entire dataset. To accomplish that, antecedent support, consequent support and rule support are partially calculated for each Map, and then they are aggregated to obtain the global evaluation that allow us to calculate the quality measures of the complete dataset. After that, all the ARs of each RuleSet are used to update an external set of rules henceforth named GlobalRulePool, that store the nondominated solutions found for the entire dataset considering the quality measures previously calculated. These quality measures will be the objective functions used by the sequential association rule algorithm. Subsequently, the redundant ARs of the GlobalRulePool are removed. The following stages are incorporated into the proposed Intellectual Rule Generation method to obtain the unknown attack rules structure as well as the quality rules.

Stage 1: The first stage, that follows a MapReduce design, is devoted to run an algorithm to obtain a set of ARs (RuleSet) for each subset in which the input dataset is divided. The step by step procedure the retrieving all the instances of the Map Split and runs the association rule algorithm in the instances of Map partition.

```
Input: data that represents the dataset, each row is an Array of
values.
Output: The rules discovered by the AR algorithm
Step 1: RuleSet ←
Step 2: map partitions instances ∈ data
Step 3: {instances will contain all instances in each partition}
Step 4: associationRules ← algorithm.run(instances)
Step 5: end map
```

Stage 2: The second Stage, that also follows a MapReduce scheme, performs the support computation of the ARs obtained in the previous phase considering all the instances of the dataset.

- This phase aims at evaluating the quality of the rules over the entire dataset because it is necessary in the next phase. Note each RuleSet of the first phase has been only evaluated
- using the instances of the subset in which they have been trained. Therefore, in the second phase is necessary to calculate the quality of the rules over the entire dataset with the aim at selecting the best rules that present the best performance in the original dataset with all the instances and not only in the subset trained.

The following explains the step by step procedure that computes the support of each association rule:

Mapper Stage

```
Input: RuleSet a set of rules discovered by the association rule algorithm
Output: (key, value) pair, where key indicates ruleId of each QAR of RuleSet evaluated
in the instance corresponding to the offset key and value contains the support of the
antecedent (supA), support of the consequent (supC ) and support of the rule (supR) of
each rule in such instance.
Step 1: supports ←
Step 2: map partitions instances ∈ data
Step 3: {instances will contain all instances in each partition}
Step 4: for each Rule ∈ RuleSet do
Step 5: {ruleId, {supA, supC, supR}} ← computeSupports(instances,Rule)
Step 6: end for
Step7: end map
```

Reducer Stage: It computing the support of the model:

```
Input: supports a (key, value) pair, where key is the id of a rule and value is the support
of the
antecedent (supA), support of the consequent (supC ) and support of the rule (supR) of
such rule for each instance.
Output: (key, value) pair, where key is the id of each rule of RuleSet and value contains
the global support of the antecedent, consequent and the rule considering all the
instances.
Step 1: Rules ← supports.reduce{(a, b) =>
Step 2: a.supA + = b.supA
Step 3: a.supC += b.supC
Step 4: a.supR += b.supR
Step 5:}
```

Stage 3: The third stage sequentially updates a global rule set denominated as GlobalRuleP ool that stores the non-dominated solutions found in the entire dataset. To accomplish that, the same measures used by the sequential algorithm to evaluate the quality of the ARs are calculated by the support obtained in the second phase.

Input: (key, value) pair, where key is the id of each rule of RuleSet and value contains the global
support of the antecedent, consequent and the rule considering all the instances.
Output: GlobalRulePool the external set of rules that contains the non-dominated solutions found for the entire dataset.

```
Step 1: for RuleRS ∈ RuleSet do
Step 2:   if RuleRS satisfies Minimum Thresholds then
Step 3:     for RuleGP ∈ GlobalRulePool do
Step 4:       if ruleRS dominates ruleGB then
Step 5:         remove ruleGB from GlobalRulePool
Step 6:         add ruleRS to GlobalRulePool
Step 7:       end if
Step 8:       if ruleRS and ruleGB are non-dominated solutions then
Step 9:         add ruleRS to GlobalRulePool
Step 10:      end if
Step 11:    end for
Step 12:  end if
Step 13: end for
Step 14: GlobalRulePool ← removeRedundant(GlobalRulePool)
```

Stage 4: The fourth stage builds a new rule set of a GlobalRulePool by following a MapReduce scheme.
Input: data that represents the dataset, each row is an Array of values.
Output: newData a new dataset with all uncovered instances.

```
Step 1: newData ← data.filter{instance =>
Step 2:   keep ← TRUE
Step 3:   for Rule ∈ GlobalRulePool do
Step 4:     if instance is satisfied by Rule then
Step 5:       keep ← FALSE
Step 6:     end if
Step 7:   end for
Step 8:   keep
Step 9: }
```

Phase 5: Pattern Matching Method: Brute-Force method has used in this framework for matching the unknown attacks patterns in the host-based IDS.

The brute force algorithm [11] consists in checking, at all positions in the input between positions 0 and n − m (left to right), whether an occurrence of the pattern starts there or not. After each attempt, the algorithm shifts the pattern by exactly one position to the right. This algorithm is considered naive based on the fact that it does no pre-computation on the keyword. However, that is advantageous if memory space is a concern, since it keeps only a small constant (not a function of n or m) amount of information about its current position.

Input:

```
    x ← array of m bytes representing the keyword
    m ← integer representing the keyword length
    y ← array of n bytes representing the text input
    n ← integer representing the text length
Step 1: For every possible character in y, for j = 0 → n − m do
Step 2:   $i \leftarrow 0$
Step 3:   while $i < m$ and $x[i] = y[i + j]$ do
Step 4:     $i \leftarrow i + 1, i = count\ of\ matching\ characters\ at\ and\ after\ y[j]$
Step 5:   end while
Step 6:   $if\ i\ \geq m\ then$
Step 7:     output j
Step 8:   end if
Step 9: end for
Step 10: end procedure
```

FLOW DIAGRAM OF RESEARCH WORK

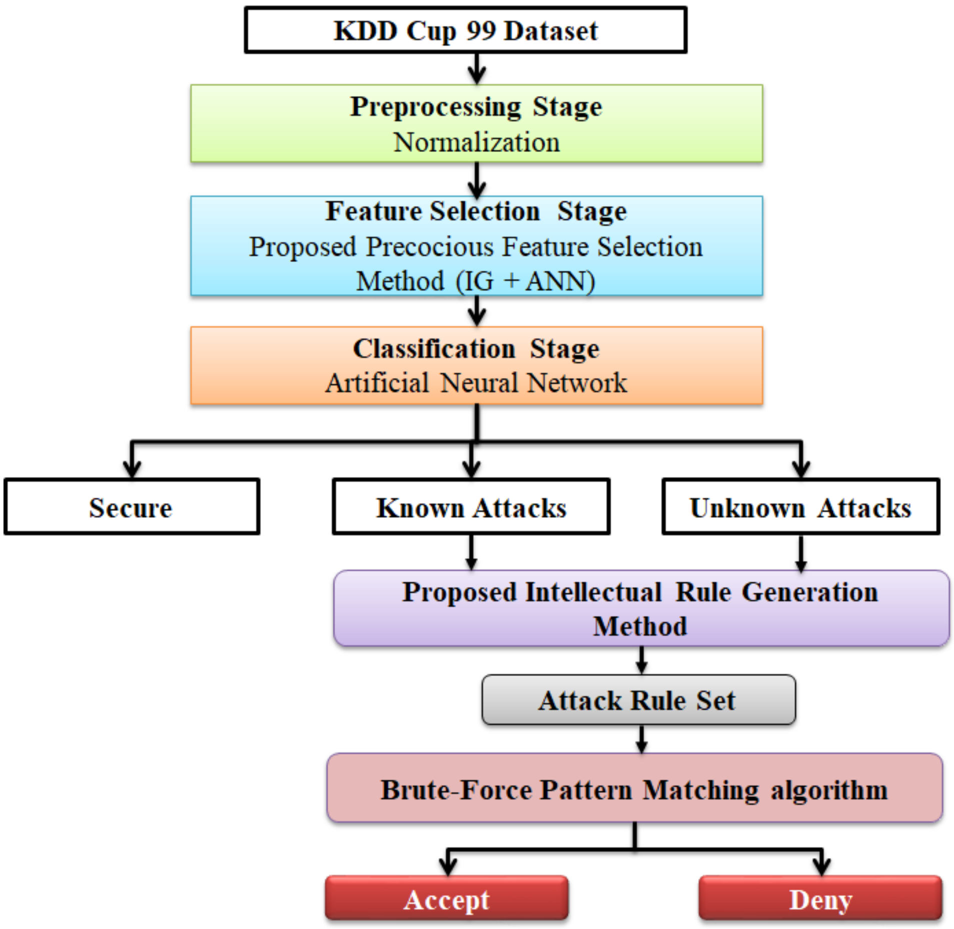
KDD Cup 99 Dataset
Preprocessing Stage
Normalization
Feature Selection Stage
Proposed Precocious Feature Selection Method (IG + ANN)
Classification Stage
Artificial Neural Network
Secure
Known Attacks
Unknown Attacks
Proposed Intellectual Rule Generation Method
Attack Rule Set
Brute-Force Pattern Matching algorithm
Accept
Deny

Figure 1: Architecture Diagram for handling Heterogeneous information

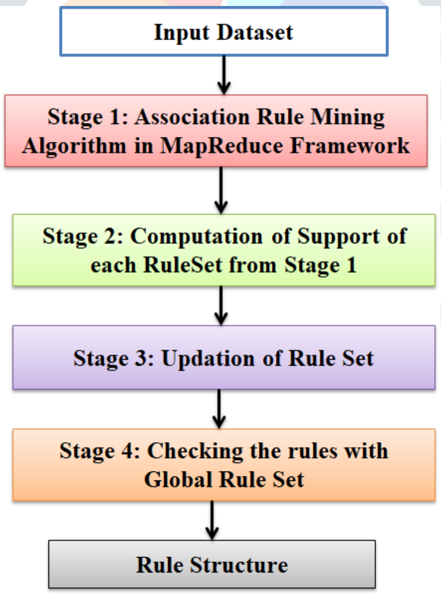
Input Dataset
Stage 1: Association Rule Mining Algorithm in MapReduce Framework
Stage 2: Computation of Support of each RuleSet from Stage 1
Stage 3: Updation of Rule Set
Stage 4: Checking the rules with Global Rule Set
Rule Structure

Figure 2: Proposed Intellectual Rule Generation Method

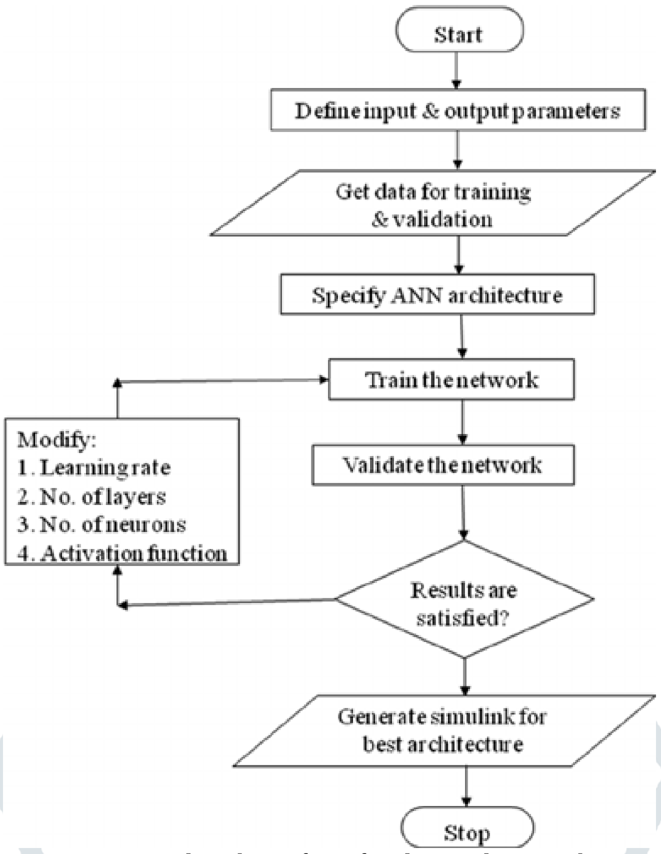

Figure 3: Flowchart of Artificial Neural Network

VI. IMPLEMENTATION

DATA COLLECTION

This is the data set used for The Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with KDD CUP-99. The Fifth International Conference on Knowledge Discovery and Data Mining. The competition task was to build a network intrusion detector, a predictive model capable of distinguishing between ``bad" connections, called intrusions or attacks, and ``good" normal connections. This database contains a standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment.

RESULT OBTAINED BY PROPOSED FEATURE SELECTION METHOD

Following table 8.1 provides the outcome attained by the pro-posed Precocious Feature Selection method and current filter-based feature selection techniques like Gain Ratio, Symmetrical Uncertainty, and Information Gain. From table 2, Gain Ratio filters 32 features, Symmetrical Uncertainty filters 31 features, Information Gain screens only 27 features, and the proposed PFSM gives only 23 features.

Table 1: Number of Features obtained by existing feature selection methods and proposed Feature Selection methods

Gain Ratio	**Symmetrical Uncertainty**	**Information Gain**	**Proposed Precocious Feature Selection Method**
is_guest_32	src_bytes	Src_bytes	dst_host_srv_count
logged_in	dst_bytes	dst_host_same_srv_rate	dst_host_same_srv_rate
Dst_bytes	dst_host_srv_count	Dst_bytes	Src_bytes
dst_host_srv_serror_rate	dst_host_same_srv_rate	dst_host_rerror_rate	dst_host_rerror_rate
Src_bytes	dst_host_rerror_rate	Service	dst_host_serror_rate
dst_host_serror_rate	Service	dst_host_diff_srv_rate	Dst_bytes
dst_host_same_srv_rate	dst_host_srv_rerror_rate	dst_host_srv_rerror_rate	dst_host_srv_rerror_rate
dst_host_srv_count	dst_host_serror_rate	srv_Count	dst_host_srv_serror_rate
dst_host_srv_rerror_rate	logged_in	count	dst_host_count
dst_host_rerror_rate	dst_host_srv_serror_rate	dst_host_serror_rate	srv_Count
Service	dst_host_diff_srv_rate	dst_host_srv_serror_rate	dst_host_diff_srv_rate
dst_host_count	srv_Count	dst_host_same_src_port_rate	count
hot	dst_host_count	logged_in	Service
dst_host_diff_srv_rate	dst_host_same_src_port_ rate	dst_host_count	dst_host_same_src_port_rate
srv_rerror_rate	count	rerror_rate	srv_rerror_rate
dst_host_same_src_port_r ate	flag	srv_rerror_rate	serror_rate
Flag	srv_rerror_rate	same_srv_rate	srv_serror_rate
srv_count	rerror_rate	diff_srv_rate	dst_host_srv_diff_host_rate

rerror_count	is_guest_32	dst_host_srv_diff_host_rate	rerror_rate
dst_host_srv_diff_host_ra te	dst_host_srv_diff_host_r ate	serror_rate	same_srv_rate
srv_serror_rate	same_srv_rate	srv_serror_rate	diff_srv_rate
count	diff_srv_rate	is_guest_32	srv_diff_host_rate
serror_rate	hot	hot	Protocol_type
diff_srv_rate	srv_serror_rate	Protocol_type	
Protocol_type	hot	srv_diff_host_rate	
same_srv_rate	srv_serror_rate	num_compromised	
srv_diff_host_rate	serror_rate	num_failed_32s	
num_compromised	Protocol_type		
num_failed_32s	srv_diff_host_rate		
Wrong_fragment	num_compromised		
land	num_failed_32s		
urgent			

RESULT OBTAINED BY PROPOSED INTELLECTUAL RULE GENERATION METHOD (IRGM) FOR UNKNOWN ATTACK RULE GENERATION PHASE

Number of Rules Generated

Following table 8.2 depicts the number of rules generated by the original dataset, Gain ratio, Symmetrical Uncertainty, Information Gain and Proposed Precocious Feature Selection method (PFSM) processed dataset. It is clear that the proposed PFSM gives a smaller number of rules when it is compared with the other existing feature selection methods.

Table 2: Number of Rules obtained by Original dataset, Gain Ratio, Symmetrical Uncertainty, Information Gain and Proposed PFSM processed dataset using Proposed Intellectual Rule Generation Method

Support and Confidence Value	Number of Rules Obtained by Proposed IRGM				
	Original Dataset	Gain Ratio	Symmetrical Uncertainty	Information Gain	Proposed PFSM
0.1 and 0.1	1125	1052	994	965	740
0.2 and 0.2	1045	987	901	863	621
0.3 and 0.3	989	896	878	796	532
0.4 and 0.4	921	736	821	654	458
0.5 and 0.5	878	691	703	636	403
0.6 and 0.6	811	584	632	598	382
0.7 and 0.7	765	496	524	545	276
0.8 and 0.8	610	345	405	492	214
0.9 and 0.9	523	296	327	397	158
1.0 and 1.0	398	274	298	304	64

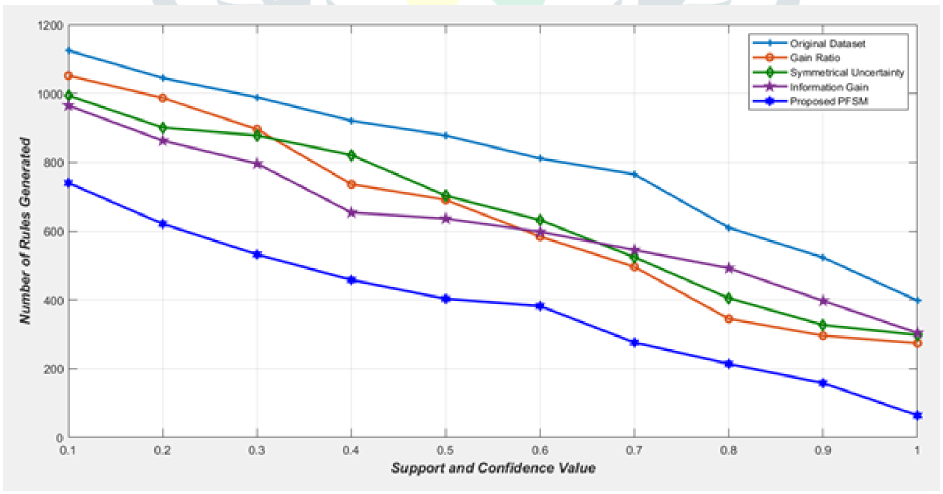

Figure 4: Graphical Representation of the Number of Rules generated by Original Dataset, GR, SU, IG and Proposed PFSM using Proposed IRGM

Memory Consumption for Rules Generated

Memory consumption refers to the amount of memory taken for generating the association rule, which is measured in terms of Mega Bytes (MB) [14].

$$M\ (Memory\ Consumption) = n\ \times M(n)$$

where M is the memory consumption for association rule generation, n is the number of frequent generated rules, and $M(n)$ is the memory required for generation of rules. When the memory consumption for association rule generation is low, a method is said to be more efficient. Table 4 depicts the memory consumption (in Mega Bytes) for generated rules by using ARM. From the table 8.3, it is clear that the proposed PFSM method consumes less memory when compared with other feature selection methods.

Table 3: Memory Consumption (in Mega Bytes) for generated rules by using Proposed IRGM for original dataset, Gain Ratio, Symmetrical Uncertainty, Information Gain and Proposed PFSM processed dataset

Support and Confidence Value	**Memory Consumption in Mega Bytes by Proposed IRGM**				
	Original Dataset	**Gain Ratio**	**Symmetrical Uncertainty**	**Information Gain**	**Proposed PFSM**
0.1 and 0.1	9.8	7.2	6.8	7.1	4.5
0.2 and 0.2	11.2	8.5	9.2	9.5	6.1
0.3 and 0.3	20.3	9.0	10.6	11.5	7.9
0.4 and 0.4	28.3	12.3	10.9	13.5	8.1
0.5 and 0.5	29.8	18.9	13.6	17.8	9.6
0.6 and 0.6	35.6	21.3	19.6	18.8	10.7
0.7 and 0.7	40.8	28.4	21.2	25.4	11.3
0.8 and 0.8	42.6	31.4	28.4	30.7	12.2
0.9 and 0.9	49.2	39.8	30.7	35.6	13.7
1.0 and 1.0	52.3	41.5	35.5	36.2	14.9

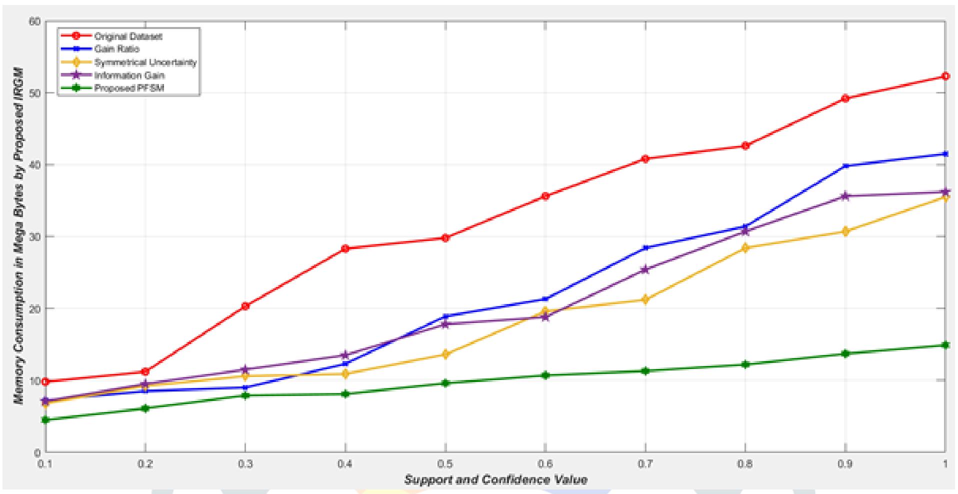

Figure 5: Graphical Representation of the Memory Consumption in MegaBytes for Original dataset, GR, SU, IG and proposed PFSM processed dataset using Proposed IRGM

Computational time taken for generating rules

The computational time for frequent item sets generation measures the amount of time taken for generating the frequent item sets with respect to given support and confidence values. It is measured in terms of milliseconds (ms) and mathematically formulated as follows [14],

$$RT = n \times T(n)$$

where *RT* is the running time, *n* represents the number of frequent item sets generated, and *T(n)* represented time taken for frequent item set generations. When the running time for frequent item set generation is low, the method is said to be more efficient. Table 8.4 presents the total running time (in seconds) for the generated rules by using ARM.

Table 4: Computation time taken for generating rules by using Proposed IRGM for Original dataset, Gain Ratio, Symmetrical Uncertainty, Information Gain and Proposed PFSM processed datasets.

Support and Confidence Value	**Total Computational time (in milliseconds) by Proposed IRGM**				
	Original Dataset	**Gain Ratio**	**Symmetrical Uncertainty**	**Information Gain**	**Proposed PFSM**
0.1 and 0.1	112	98	97	96	65
0.2 and 0.2	105	91	84	90	62
0.3 and 0.3	98	85	79	81	56
0.4 and 0.4	91	79	71	75	45
0.5 and 0.5	85	65	64	63	40
0.6 and 0.6	78	54	51	53	35
0.7 and 0.7	65	51	48	45	28
0.8 and 0.8	59	49	42	42	25
0.9 and 0.9	51	48	41	34	24
1.0 and 1.0	45	41	38	31	22

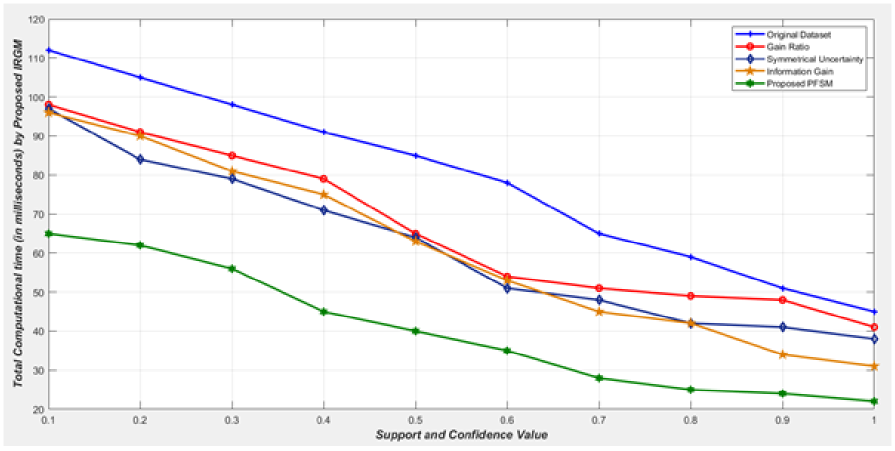

Figure 6: Graphical Representation of the Total Computation time in Milliseconds for Original dataset, GR, SU, IG and proposed PFSM processed dataset using Proposed IRGM

RESULT OBTAINED BY PATTERN MATCHING ALGORITHM

In this section, the experimental result that we have obtained by using various single keyword pattern matching algorithm in the proposed framework. The various pattern matching algorithms like Karp-Rabin Algorithm, Knuth-Morris-Pratt Algorithm, Boyer-Moore Algorithm are compared with Brute-Force algorithm in the proposed framework. The performance analysis of the algorithms is analyzed by using running time of the algorithm with input size of the pattern.

Table 5: Single Keyword Pattern Matching algorithm running time in seconds

Input Size	Single Keyword Pattern Matching Algorithms Running Time in Seconds			
	Karp-Rabin	Knuth-Morris-Pratt	Boyer-Moore	Brute-Force
1000	16	15	17	14
2000	48	46	45	44
3000	76	74	80	67
4000	104	103	102	90
5000	128	129	130	109
6000	145	144	146	118
7000	161	162	164	125
8000	178	187	179	140
9000	198	189	199	156

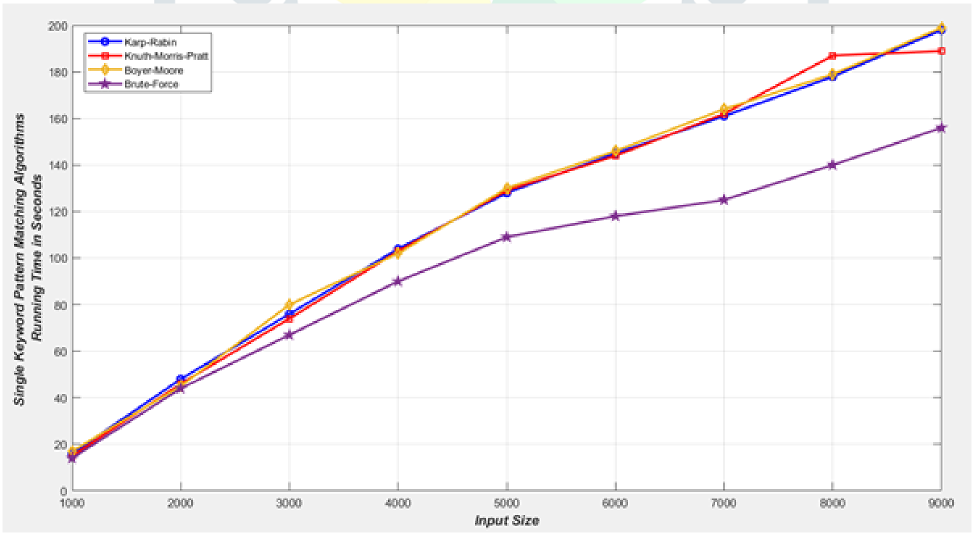

Figure 7: Graphical Representation of the Single Keyword Pattern Matching Algorithms Running Time in Seconds for proposed PFSM

VII. CONCLUSION

From the result obtained, it is clear that the proposed framework for the Host based Intrusion Detection system with Proposed PFSM and Proposed IRGM gives minimum number of rules for detecting the unknown attacks in the network. The total running time of the frameworks depends the rule structure and pattern matching in the flows. When the proposed framework with proposed IRG method has utilized for unknown rule generation and single keyword pattern matching Brute-Force method has used to reduce the total running time of the framework. The proposed framework consumes less memory and less execution time when it is used with proposed PFSM and proposed IRG method. The future direction of this study will include testing newer algorithms with other feature selection techniques. We will experiment on cluster techniques as well as ensemble algorithms.

REFERENCES

[1] Shengyi Pan, Thomas Morris and Uttam Adhikari, "Developing a hybrid intrusion detection system using data mining for power sys-tems," IEEE Transactions on Smart Grid, Vol.6, No.6, pp.3104-3113, 2015.
[2] Syed Ali Raza Shah and Biju Issac, "Performance comparison of intrusion detection systems and application of machine learning to Snort system," Future Generation Computer Systems, Vol. 80, pp.157-170, 2018.
[3] M. Elbasiony, Reda, et al, "A hybrid network intrusion detection framework based on random forests and weighted k-means," Ain Shams Engineering Journal, Vol. 4, No. 4, pp.753-762, 2013.
[4] Iftikhar Ahmad, et al., "Enhancing SVM performance in intrusion detection using optimal feature subset selection based on genetic principal components," Neural computing and applications, Vol.24, No.7-8, pp.1671-1682, 2014.
[5] Aburomman, Abdulla Amin and Mamun Bin Ibne Reaz, "A novel SVM-kNN-PSO ensemble method for intrusion detection system," AppliedSoftComputing, Vol. 38, pp.360-372, 2016.
[6] Thaseen, Ikram Sumaiya and Cherukuri Aswani Kumar, "Intrusion detection model using fusion of chi-square feature selection and multi class SVM," Journal of King Saud University-Computer and Information Sciences, Vol. 29, No.4, pp.462-472, 2017.
[7] Li, Longjie, et al., "Towards Effective Network Intrusion Detection: A Hybrid Model Integrating Gini Index and GBDT with PSO," Journal of Sensors, 2018.
[8] Jadhav, Swati, Hongmei He, and Karl Jenkins. "Information gain directed genetic algorithm wrapper feature selection for credit rat-ing." Applied Soft Computing 69 (2018): 541-553.
[9] Venkataraman, Sivakumar, and Rajalakshmi Selvaraj. "Optimal and Novel Hybrid Feature Selection Framework for Effective Data Classification." Advances in Systems, Control and Automation. Springer, Singapore, 2018. 499-514.
[10] Ramírez-Gallego, Sergio, et al. "An information theory-based fea-ture selection framework for big data under apache spark." IEEE Transactions on Systems, Man, and Cybernetics: Systems 48.9 (2018): 1441-1453.
[11] MR Gauthama Raman, et al., "A hypergraph and arithmetic resi-due-based probabilistic neural network for classification in intru-sion detection systems," NeuralNetworks, Vol. 92, pp.89-97, 2017.
[12] Abien Fred M. Agarap, "A Neural Network Architecture Combining Gated Recurrent Unit (GRU) and Support Vector Machine (SVM) for Intrusion Detection in Network Traffic Data," Proceedings of the 2018 10th International Conference on Machine Learning and Computing, ACM, 2018.
[13] Dataset Source: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
[14] Poornappriya, T. S., and M. Durairaj. "High relevancy low redundancy vague set based feature selection method for telecom dataset." *Journal of Intelligent & Fuzzy Systems*, Preprint: 1-18.
[15] M. Durairaj, T S Poornappriya, "Choosing a spectacular Feature Selection technique for telecommunication industry using fuzzy TOPSIS MCDM.", *International Journal of Engineering & Technology*, 7 (4) (2018) 5856-5861.
[16] M. Durairaj, T. S. Poornappriya, "Importance of MapReduce for Big Data Applications: A Survey", *Asian Journal of Computer Science and Technology*, Vol.7 No.1, 2018, pp. 112-118.
[17] M. Lalli, V.Palanisamy,(2016), "Filtering Framework for Intrusion Detection Rule Schema in Mobile Ad Hoc Networks", International Journal of Control Theory and Applications –(IJCTA),9(27), pp. 195-201, ISSN: 0974-5572
[18] M. Lalli, V.Palanisamy,(2017), "Detection of Intruding Nodes in Manet Using Hybrid Feature Selection and Classification Techniques", Kasmera Journal, ISSN: 0075-5222, 45(1) (SCIE)(Impact Factor:0.071).
[19] M. Lalli, V.Palanisamy, (Sep 2014), "A Novel Intrusion Detection Model for Mobile Adhoc Networks using CP-KNN", International Journal of Computer Networks & Communications- (IJCNC), Vol.6, No.5, ISSN:0974-9322.
[20] M. Lalli, "Statistical Analysis on the KDD CUP Dataset for Detecting Intruding Nodes in MANET", *Journal of Applied Science and Computations*, Volume VI, Issue VI, JUNE/2019, 1795-1813.
[21] M. Lalli, "Intrusion Detection Rule Structure Generation Method for Mobile Ad Hoc Network", *Journal of Emerging Technologies and Innovative Research*, June 2019, Volume 6, Issue 6, 835-843.