Clustered Ensemble Feature Selection with M-GRU Classification for Efficient Intrusion Detection System of Industrial Systems

M Karthigha (karthighamohan@gmail.com)
Sri Ramakrishna Engineering College
Latha L
Kumaraguru College of Technology

Research Article

Keywords: industrial control system, intrusion detection, ensemble, gated recurrent unit

Posted Date: April 22nd, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1571372/v1

Clustered Ensemble Feature Selection with M-GRU Classification for Efficient Intrusion Detection System of Industrial Systems

Karthigha M and Latha L

Department of Computer Science Engineering
[1] Sri Ramakrishna Engineering College, Coimbatore - 641 022, India
[2]Kumaraguru College of Technology, Coimbatore campus-641 049, India
[1]karthighamohan@gmail.com , [2]latha.l@kct.ac.in

Abstract: In today's scenario, infrastructures, such as the power grid and nuclear power plant, the industrial control system (ICS) is crucial. Yet, there is growing worry that ICS systems are susceptible to threats or attacks, and that even minor changes or manipulation could cause major damage to industrial operations**.** In this paper, an efficient intrusion detection system with clustered ensemble feature selection and Multi-Level Modified Gated Recurrent Unit (M-GRU) classification model is proposed. Clustered ensemble feature selection approach is to find the best feature subset. The features are ranked based on scores from base algorithms and aggregated using aggregation algorithms. Diverse base algorithms are elected using clustering technique. The features designated are fed into a multi class classification algorithm Multi-Level Modified Gated Recurrent Unit (M-GRU). NSL-KDD training and testing dataset is used in this work for feature selection and classification. The results infer that the proposed feature selection method with classification model improves accuracy and lowers false alarm rate compared to other models.

Keyword: industrial control system, intrusion detection, ensemble, gated recurrent unit.

1. Introduction

Industrial Control System (ICS) is used for industrial process control. Real time data acquisition, monitoring and automation or partial automation of industrial processes in major fields like electric utilities, chemical plants, manufacturing, refineries, pipelines, transportation are governed by ICS [1]. ICS encompasses several types of control system which includes supervisory control and data acquisition system (SCADA), process control system(PCS), distributed control system (DCS), safety instrument system [2].

In the past ICS are isolated from other networks. But in recent times as these systems are extensively used in many fields, it is integrated profoundly with internet and other networks [3].Because of connecting ICS components to the internet, vulnerabilities of ICS to cyber-attacks are heavily increasing.

Active research has been prompted to prevent cyber-attacks in ICS. Intrusion detection can be instigated to curb cyber-attacks. But implementing Intrusion detection system in ICS is intricate as ICS are geographically distributed, in need of secure communication channel and real time requirements [4]. More intelligent intrusion detection systems have to be developed and deployed in real industrial control system environment which collects enormous data from diverse areas. The data may possibly contain inappropriate and redundant features which causes increased processing time, overfitting and low detection rate with high false alarm rate [5]. Feature selection helps to ascertain significant features and these selected features prominently contribute in the classification process of IDS to improve accuracy of IDS, increases training and testing time and reduce overfitting. Moreover, it reduces the complexity of the model to interpret results in easier way. Furthermore, feature selection moderates processing cost, reduces storage space, and escalate the understanding of the test data. [6]

Feature selection methods are essentially categorized into three methods: filter approach, wrapper approach, and embedded approach. [7].Filter approach selects the features based on statistical measures. This can be classified into individual feature estimation and subset feature estimation. Heuristic or met heuristic approaches help to estimate the rank of the features in individual feature evaluation. In subset feature estimation subset are drawn from a certain strategy [8]. ANOVA, Chi-square and correlation coefficient can be used to select subset of features. Wrapper approach uses all possible combinations of features against a machine algorithm. Unlike filter approach, wrapper approach makes use of the classifier performance to estimate the optimal features which consequently results in improved prediction accuracy. The commonly used wrapper approaches are Forward Selection, Backward Elimination, and Recursive Feature Elimination. [9]. Embedded approach utilizes both filter and wrapper approach. These techniques exert the algorithms which has its own innate Feature selection method. Select k-best, LASSO and ridge regression can be employs as they have intrinsic score functions and penalty functions. [10]

Ensemble feature selection integrates the output of individual feature selectors so that it affords an improved estimation to the optimal selection of features. The user doesn't have to linger on a certain feature selector. This type of feature selection methods is gaining attention because of its enhanced performance and robust results.

*Corresponding author. Tel.: +91-6380427344.
E-mail address: karthighamohan@gmail.com

Different levels of variations can be done in ensemble feature selection [11],

- Threshold methods
- Aggregation methods
- Learning method
- Feature subsets
- Dataset subsets

Ensemble feature selection is categorized into homogeneous and heterogeneous feature selection. In homogenous feature selection, subsamples of the dataset are created. The feature selector termed as base selector applied is same for all the training subsets. An aggregation method is employed to integrate the results of various subsets. In heterogeneous feature selection, different base selectors are applied for training dataset after that an aggregation method is utilized to integrate the results from various feature selectors. Consider a dataset D = (x_k,...,x_n), xk = (x_{k1} ,..., x_{mk}) with n rows and m attributes. Different heterogeneous algorithms (A_1,..., An) is applied to D resulting on p feature subsets (FS_1,..., FSp) each comprising selected features FSp = (fp,1,..., fp,s).

The feature list obtained from feature selection is trained with a classification model. Gated recurrent unit (GRUs) is an enhanced version of recurrent neural networks that can be used as classification model. The dataset is validated with this model to perform attack classification

1.1 Our Contribution

The contributions in this paper are summarized as follows

- An extensive scope of clustered ensemble feature selection of intrusion detection technique for ICS is developed for randomly shuffled dataset.
- GMM based Clustering is employed to select the different base selectors thereby escalating the diversity among them.
- Developed a multi-level modified GRU(M-GRU) for multi class classification with NSL-KDD dataset
- Investigated the effect of M-GRU model on different feature subsets based on different thresholds of diverse aggregation methods that obtained from ensemble feature selection method.
- Compared the results of M-GRU with for various thresholds onto other algorithms with different threshold.

2 Literature Review

This review elucidates individual feature selection techniques, ensemble feature selection techniques and various classification models. Abdullah [12] proposed an feature selection method based on information gain for NSL-KDD dataset. The dataset is split into different subsets. Each subset is ranked with information gain and a threshold is applied to select the number of feature. Finally, classification is done after aggregating the individual results. Cross validated random forest classifier gives the highest accuracy of 99.9%

Khammassi et al [13] proposed Genetic algorithm-Logistic regression (GA-LR) wrapper based approach for feature selection. C4.5, RF, and NBTree classification models are used against the feature subset obtained from GA-LR wrapper for KDD99, UNSW-NB15 datasets. The results infer that the proposed algorithm provides 0.105% False Alarm Rate with 18 features for KDD dataset and 6.39% False Alarm Rate with 18 features for UNSW-NB15 dataset.

Malik[14] proposed Particle Swarm Optimization-Random Forest (PSO-RF) based technique for dimensionality reduction followed by classification for developing an intrusion detection system. Benchmark dataset KDD99Cup dataset is used to detect different kind of attacks in the network. The intrusion detection rate is 96.78% and false positive rate is 0.1546% by using PSO-RF technique and is contemplated to be better than other techniques.

Pham et al [15] proposed an ensemble model in which two approaches are used for feature selection. Naive bayes classifier and Gain ratio are the two feature selection techniques employed to tree classifiers against NSL-KDD dataset. The experimental results show that the bagging approach with J48 as the base classifier with 35 features contributes 84.25% accuracy and 2.79% false alarm rate.

Shukla et al [16] proposed ahomogenous filter based ensemble feature selection approach wherein mRMR, JMI, and CMIM feature selection methods are made use of against KDD Cup 99 and the NSL-KDD dataset. These methods provide individual rank from which the output is combined to form final feature

output by amalgamation process. The result reveals that proposed method attained an accuracy of 98.95% and 98.12% in the in KDD Cup 99 and NSL-KDD data set respectively.

He et al [17] proposed ensemble feature selection to improve accuracy of classification. Mean decrease impurity, stability selection, chi-square test, random forest classifier, and recursive feature elimination feature selection techniques are employed against unsw_nb15, kddcup99, and cidds-001 datasets. Support vector machine, KNN and multi-layer perception, and decision tree are exploited to show the improvement of accuracy by utilizing ensemble feature selection. The result indicated that cidds-001 dataset gives accuracy 99.40% for the proposed approach.

Krishnaveni et al [18] proposed ensemble feature selection and ensemble classifiers for network intrusion detection. Filter features selection approaches Information Gain, Gain-ratio, Chi Squared, Symmetric Uncertainty and Relief are utilized against real time honeypot dataset, NSL KDD dataset and Kyoto. These individual results are aggregated using combination rule. SVM, decision tree, logistic regression, naïve bayes classifiers are used for evaluating the feature selection method. This method shows viable results in case of Kyoto dataset taking 7 features with 99.89 accuracy.

Salo et al [19] proposed an Information gain and Principal component analysis based dimensionality reduction technique which uses an ensemble classification based on Support Vector Machine, Instance based learning method and multi perceptron. The datasets used are ISCX 2012, NSL-KDD, and Kyoto 2006+ for dimensionality reduction. The accuracy of 99.01% ,98.24%,98.95% detection rate of 99.1%, 98.2% and 99.8% and false alarm rate of 0.01%,0.017%, 0.021%gained for proposed algorithm for SCX 2012,NSL-KDD and Kyoto 2006 + datasets.

Ling et al [20] proposed effective IDS using a bidirectional simple recurrent unit that helps to remove the vanishing gradient problem that helps to upgrade the effectiveness of training. Two datasets are used from Mississippi university for simulation. The proposed method has a accuracy of 92.94% which is better than other algorithms naïve bayes, SVM, CNN, LSTM, GRU, REP Tree for both datasets. It is found that the model training time has also been reduced for the proposed algorithm when compared to other algorithms.

Ge et al [21] proposed an effective IDS to enhance the security in IoT environment. Multi class classification model based on feed forward network with high dimensional categorical feature encoding is proposed. BoT-IoT dataset is used in this work with 73,360,900 records which is higher than the other datasets.

In this paper ensemble feature selection with modified Gated recurrent classifier is implemented to build an intrusion detection system for industrial control network.

3 Dataset Description

NSL KDD is a benchmarking dataset utilized in our proposed system. NSL KDD dataset is the successor of KDD Cup dataset. This dataset comprises of internet traffic captured by an intrusion detection system. This has 43 features out of which 41 features refers to internet traffic and remaining two features includes label and score[22].

The training dataset has 125973 records and testing dataset has 22544 records. The dataset consists of four types of attack classes [23].

- Denial of Service (DoS) - The attacker targets a victim and shutdown the server. It depletes the target's resources by flooding abnormal amount of traffic into target's network.
- Probe- The attacker attempts to observe the network and steal information about the network.
- User to Root (U2R) – In this attack in which a user with normal user account attempts to escalate the privilege as a root user to gain access to the system or network.
- Remote to Local (R2L) - This attack attempts to get local access to the remote machine and find their way into the network.

The features and their categories are listed below

- Intrinsic features: 1–9
- Content features: 10–22
- Time-based features: 23–31
- Host-based features: 32–41

4 Proposed System for Experimental analysis

To improve the accuracy of intrusion detection system and reduce the false alarm rate we propose a proficient M-GRU based IDS with ensemble feature selection method. Fig. 1 exhibits the overall IDS framework of the proposed clustered feature selection with M-GRU based IDS and it comprises of the following phases:

1. Data Preprocessing-In this initial step the raw data of NSL KDD dataset is preprocessed to apply further feature selection method. For instance, the categorical features are converted into numerical category and the label field is converted into to numerical values.
2. Random Shuffling- Dataset values within each feature are exchanged/ shuffled at random.
3. Feature Selection-Clustered Ensemble Feature selection model helps to reduce the dimensionality of the dataset and assist in selecting the best significant features.
4. Classification model training-After selecting the relevant features, classification model based on GRU is trained against the training dataset.
5. Attack classification-The trained model is tested with testing dataset to come up with classification decisions (attack or normal)
6. Performance evaluation-The model is tuned until it gives high accuracy and low false alarm rate.

An ensemble feature selection is proposed with M-GRU classification model to constitute efficient IDS. In data preprocessing the NSL-KDD (TrainD) dataset is label encoded and normalization of dataset is done. After preprocessing of training data, the dataset is shuffled randomly within each class to reduce variance and overfitting.

Then ensemble feature selection process i.e. multiple individual feature estimators is solicited. The base selectors (F_i) for feature selection are LightGBM, Catboost, XGBoost, Mutual information, Extra tree classifier, RFE. Each base algorithms produces a ranking list (FR) based on the Feature important score (FIS) where the ranking is done according to the highest feature important score. i.e. the feature with highest FIS is ranked first and so on.

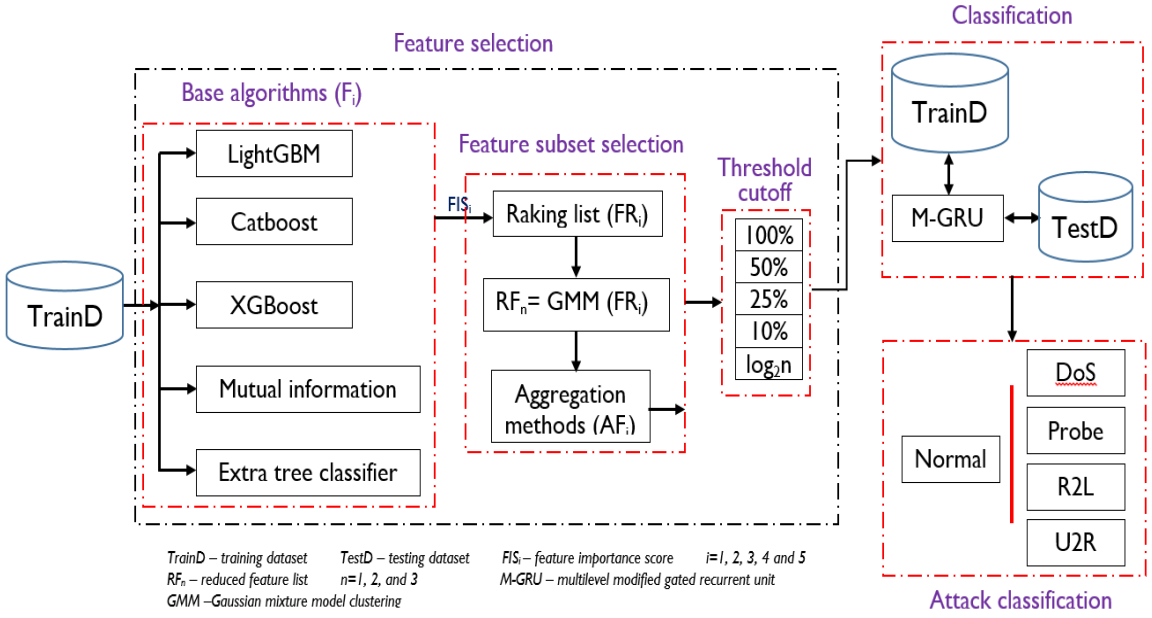


Figure 1 Overall framework of proposed system

The diversity of the base algorithms are ensured by applying a clustering algorithm. GMM clustering is used in this work to eradicate the base algorithms which likely to work similar. After eliminating the related base selectors, the remaining FR_i is aggregated to a single feature list. Different aggregation methods such as Arithmetic mean, Geometric Mean, MC4, MCT, Random Dictator, Score Voting, Borda, and Min have been tried to obtain a best feature subset list (AF_i). The optimal feature subset is decided by selecting various thresholds of 100%, 50%, 25%, 10%, log_2n. With each subsets formed, M-GRU based classification model has been trained. Evaluating the obtained model with NSL KDD testing dataset, the attack classification is

done. The proposed model's performance is analyzed and compared with other existing classification model GRU, LSTM, RF, Naïve Bayes.

4.1 Feature Selection Process

Feature selection process consists of four steps to find the optimal feature subset. The steps are,
1.Ensemble Feature Selection
2. Clustering
3.Rank Aggregation
4.Threshold selection

4.1.1 Ensemble Feature Selection

4.1.1.1 Light GBM

Light GBM is a tree based model (Light Gradient boosting machine) proposed by Microsoft in 2017 [24], an effective employment of GBDT. Light GBM uses Gradient based one side sampling which reduces the avoidable data. It preserves only the discrete values of features there by reducing the memory consumption. LightGBM has two feature importance measures split and gain and weight. Gain is the default measure that provides the average gain of feature while it is utilized in trees. Split measure gives the total number of times a feature is employed to split the data through all trees.

4.1.1.2 Catboost

Catboost follows ordered boosting [25] and GPU accelerated which helps it to predict faster [26]. The feature importance measure PredictionValuesChange, LossFunctionChange, InternalFeatureImportance, PredictionDiff. PredictionValuesChange indicates for each feature how much the prediction changes on average when the feature value changes. Feature importance is calculated for prediction values change by the following equations 1 and 2.

$$F=\sum_{t,l}(s_1 - asr)^2 * w1+(s_2 - asr)^2 * w2 \quad \text{---------- (1)}$$

$$asr = \frac{s1.w1 + s2.w2}{w1+w2} \quad \text{---------- (2)}$$

The total weight of objects in the left and right leaves is represented by w1 and w2. If no weights are specified for the dataset, this weight is equal to the number of objects in each leaf. s_1 and s_2 denote the value in the left leaves and right leaves.

4.1.1.3 XGBoost

XGBoost is a Gradient Boosting Decision Tree (GDBT) which makes use of regularized learning. Gain, weight, cover is the feature importance measure of XGBoost. Gain signifies the gain score for each split of a tree, and the average gain is used to calculate the final feature importance score. The average gain is calculated by dividing the total gain of all trees by the total number of splits for each feature. The higher XGBoost's feature importance score, the more significant the corresponding feature [27].Gain is calculated by the following,

$$Gain=\frac{1}{2}\left[\frac{(\sum_{i\in S_l} g_i)^2}{\sum_{i\in S_l} h_i+\lambda}+\frac{(\sum_{i\in S_R} g_i)^2}{\sum_{i\in S_R} h_i+\lambda}-\frac{(\sum_{i\in I_l} g_i)^2}{\sum_{i\in I_l} h_i+\lambda}\right]-\boldsymbol{\gamma} \quad \text{---------- (3)}$$

S_L and S_R denote the left and right nodes after the split. $\boldsymbol{\gamma}$ represents the penalty parameter. $g_i$and h_i represents the first order and second gradients respectively.

4.1.1.4 Mutual Information

Mutual Information (MI) employs a greedy strategy to find a feature subset that has the most information about the class label. MI between Aj and B is given by[28],

$$I(B, Aj) = H(Y) - H(Y|Aj) = \sum_{Aj,B} p(A_j, B)\log \frac{p(Aj,B)}{p(Aj)p(B)} \quad \text{---------- (4)}$$

4.1.1.5 Extra Trees Classifier

Extra Trees Classifier is a sort of ensemble learning approach that aggregates the classification results of multiple de-correlated decision trees collected in a "forest" to produce its classification result. To implement feature selection with extra tree classifier, split decision measure Gini Index is computed for each feature during the forest's construction . To execute feature selection, each feature is sorted into descending order based on its Gini Importance, and the top k features are selected.

$$Gini=1-\sum ni=1(pi)^2 \quad \text{---------- (5)}$$

where pi is the probability of a feature being categorized into a specific label.

4.1.1.6 Recursive Feature Selection (RFE)

RFE is a wrapper based feature selection algorithm. The objective of recursive feature elimination (RFE) is to select features by looking at smaller sets of features in a recursive manner . To begin with, the estimator is trained on a small set of features, and the importance of each feature is calculated by any attribute. Then, from the current set of features, the least important features are pruned. On this set, this technique is recurring recursively till the preferred number of features to select is reached. The estimator used is multilayer perceptron.

The feature importance score (FIS) is engendered from each base algorithm. Features are ranked according to FIS. For example the highest FIS of Light GBM is 2394 of feature src_bytes, therefore it is ranked 1 and so on. The feature importance score is given in Table 1.

4.1.2 Clustering for ensemble Feature selection using Gaussian mixture model (GMM)

Similar fundamental assumptions in FS approaches tend to yield similar results. If numerous feature selection approaches are merged, and several of them are identical, they might dominate in aggregation, and the obtained output will be heavily skewed with their choice. This consequence can be avoided by carefully selecting feature selection approach. But, it is hard to determine which feature selection approaches have similar characteristics, and it is difficult to develop a robust aggregation mechanism that works in all instances.

So a clustered ensemble approach is proposed as an alternative to traditional ensemble FS. Similar rankers' preferences are first clustered and then used to create ensembles. Gaussian Mixture Model (GMM) is used for clustering the ranked outputs from different algorithms. After the ranker blocks, clustering is used to limit the number of ranked outputs that are fed into the aggregation block. [29]GMMs presuppose a given number of Gaussian distributions, each of which represents a cluster. As a result, the data points from a single distribution are grouped together in a Gaussian Mixture Model. GMM would calculate the probability of every data point fitting into the distributions for a given collection of data points. Estimation Maximization(EM) technique is used to estimate the parameters for a GMM with a certain number of K components. [30]

Estimation Step

Determine the expected value of the log-likelihood of the datapoints based on current parameter estimates, to assign each sample to a GMM component. Estimate the probability for each point x_i which belongs to one of the distributions c_1, c_2... c_k.

$$r_{ic}=\frac{Probability\ Xi\ belongs\ to\ c}{SumofProbabilityXibelongst0\ c1,c2,\ldots ck} \quad \text{---------- (6)}$$

Maximization step

Based on Estimation step, Π, μ and Σ values are updated.

$$\Pi=\frac{Number\ of points\ assigned\ to\ cluster}{Totalnumber\ of\ points} \quad \text{---------- (7)}$$

$$\mu = \frac{1}{Number\ of\ points\ assigned\ to\ cluster} \Sigma_i r_{ic} X_i \quad \text{---------- (8)}$$

$$\Sigma_c = \frac{1}{Number\ of\ points\ assigned\ to\ cluster} \Sigma_i r_{ic} (X_i - \mu_c)^T (X_i - \mu_c) \quad \text{---------- (9)}$$

Where μ- mean ,Σ-covariance, Π-density of the distribution.
New probabilities for each data point are determined in this step, and the values are updated iteratively. This is performed until the log-likelihood function is maximized.

Data samples containing the output of several rankers are sent to the clustering algorithm as input. GMM algorithm is employed for identifying clusters and labeling data point that belong to a certain cluster. GMM clustering identified and grouped related feature selection outputs. As similar feature selection outputs are grouped together, they have a lower probability of outvoting the other approaches, promoting diversity. The GMM prediction probability is [[1. 0. 0.] [0. 0. 1.] [1. 0. 0.][0. 1. 0.] [0. 1. 0.][0. 1. 0.]

Table 1 Feature importance score for each base algorithms

Features	Feature Selection methods					
	Light BGM	Cat boost	XG Boost	Mutual Information	Extra tree Classifier	RFE
Duration						
protocol_type	318	12.963	0.004	0.263	0.032	6
Service	640	0	0.045	0.761	0.016	3
Flag	191	0	0.009	0.680	0.068	4
src_bytes	2394	44.030	0.056	0.931	0.030	1
dst_bytes	1206	2.832	0.007	0.502	0.010	1
Land	0	0	0.000	0.004	0.000	34
wrong_fragment	146	0.135	0.014	0.053	0.016	23
urgent	0	0.000	0.000	0.000	0.000	35
Hot	426	1.699	0.011	0.062	0.009	12
num_failed_logins	64	0.000	0.002	0.005	0.000	31
logged_in	196	0.390	0.001	0.350	0.046	8
num_compromised	169	0.532	0.012	0.041	0.005	11
root_shell	144	0.000	0.000	0.004	0.000	30
su_attempted	3	0.002	0.000	0.001	0.000	32
num_root	131	0.000	0.000	0.003	0.000	10
num_file_creations	112	0.000	0.000	0.002	0.000	24
num_shells	0	0.000	0.000	0.001	0.000	33
num_access_files	39	0.000	0.001	0.003	0.000	28
num_outbound_cmds	0	0.000	0.000	0.000	0.000	37
is_host_login	0	0.000	0.000	0.000	0.000	36
is_guest_login	57	0.000	0.007	0.011	0.002	26
count	701	12.360	0.021	0.562	0.028	1
srv_count	270	0.801	0.002	0.294	0.023	2
serror_rate	178	0.827	0.012	0.406	0.046	16
srv_serror_rate	88	0.000	0.002	0.413	0.019	13
rerror_rate	155	0.000	0.001	0.073	0.018	19
srv_rerror_rate	45	0.347	0.000	0.085	0.021	18
same_srv_rate	186	0.884	0.460	0.511	0.179	7
diff_srv_rate	303	1.578	0.039	0.052	0.016	25
srv_diff_host_rate	44	0.000	0.000	0.080	0.007	20
dst_host_count	747	0.000	0.001	0.292	0.023	1
dst_host_srv_count	1094	0.251	0.005	0.540	0.078	1
dst_host_same_srv_rate	831	1.904	0.005	0.277	0.050	9
dst_host_diff_srv_rate	776	12.506	0.025	0.046	0.010	27
dst_host_same_src_port_rate	862	3.869	0.025	0.134	0.053	22
dst_host_srv_diff_host_rate	454	0.617	0.004	0.015	0.002	29
dst_host_serror_rate	497	0.000	0.008	0.418	0.117	15
dst_host_srv_serror_rate	248	0.687	0.003	0.419	0.037	14
dst_host_rerror_rate	481	0.784	0.004	0.045	0.014	21
dst_host_srv_rerror_rate	134	0.000	0.004	0.102	0.020	17

4.1.3 Rank Aggregation

Multiple rank lists are generated by individual feature selection methods. These multiple lists of ranks have to be combined and top-ranked features are selected to use in the classification process [31].The clustered following aggregation methods are used in this paper,

4.1.3.1Arithmetic Mean

Aggregation is done by simple arithmetic mean operation. This gives the average of the values produced by the rankings [32].

$$AM = (1/n) * (x_1 + x_2 + \ldots + x_n) \quad \text{---------- (10)}$$

4.1.3.2 Geometric Mean

In this aggregation is performed by taking the geometric average of the values produced by the rankings

$$GM = n\text{-root}(x_1 * x_2 * \ldots * x_n) \quad \text{---------- (11)}$$

4.1.3.2 MC4

MC4 (Markov Chain 4) is graph theory based rank aggregation method.MC4 based on simple majority voting. The MC4 algorithm's purpose is to generate an aggregate rating that ignores items that are erroneously highly ranked in a small number of lists.

4.1.3.2 MCT

MCT is rank aggregation method based on Thurstone's order-statistics algorithm.

4.1.3.2 Random Dictator

Uniform random dictatorship goes as follows, given the set of elements I and the appropriate preference relations $>=i$ for $i \in I$ on the alternatives A. It chooses an element I in I based on a uniform distribution on I, and then uses $>=_i$ to choose the most desired option by agent I as the outcome.

4.1.3.2 Score Voting

Each ranked feature score list is given by different algorithms, and the scores are added for each feature. The one with the highest total is selected as the aggregated value for that feature.

4.1.3.2 Borda

For entire ranked lists, aggregate ranks were calculated using an arithmetic average. Many more aggregation functions and changes have been proposed and used, and they can be applied to top-k lists as well. When employing the Borda Method with v voters and c candidates. Allow r1, r2,...,rv to denote the ranks granted to a contender by each voter for every given candidate. Let's call their total s = r1 + r2 + + rv. The Borda count for that candidate is then provided.

$$b = v(c + 1) – s \quad \text{---------- (12)}$$

4.1.3.2 Minimum aggregation method

The purpose of the minimal aggregation technique is to determine the minimum of the importance values provided by the rankings using basic mathematical operations. This method chooses the highest position provided by each ranking from different ranking methods. The aggregated values for each method of all the features are tabulated in Table 2.

4.1.4 Threshold

Different thresholds select different percentage of features[33]. There are many threshold methods used in the literature. Fixed threshold, Maximum Fisher's discriminant ratio, Volume of overlap region, maximum feature efficiency, and complexity fusion is some of the threshold methods that provide different subset of features. Setting thresholds of 50%, 25%, 10% and $\log_2$ (n) were employed to choose the top 50%, 25%, 10% and $\log_2$ (n) of the features after ranking by different ranking methods, correspondingly.

Table 3a,3b,3c,3d denotes the list of features selected after applying the thresholds of 50%, 25%, 10% and $\log_2$ (n).

Table 2 Aggregation of features after clustering

Features	Aggregation Methods							
	A.M	**G.M**	**MC4**	**MCT**	**Random Dictator**	**Score Voting**	**Borda**	**Min**
Duration	17.333	16.08	16	18	21	23.5	24.667	9
protocol_type	11.000	7.83	9	9	16	35	31.000	2
Service	11.667	7.719	8	8	2	33.5	30.333	2
Flag	15.667	11.292	18	18	3	27	26.333	3
src_bytes	1.000	1.000	1	1	1	41	41.000	1
dst_bytes	5.000	4.380	3	3	7	40	37.000	2
Land	31.667	31.253	33	34	33	8	10.333	25
wrong_fragment	23.000	22.894	25	23	24	18	19.000	20
urgent	34.333	33.776	37	37	39	4	7.667	26
Hot	15.000	13.708	10	11	23	28	27.000	8
num_failed_logins	29.667	29.605	29	29	31	12	12.333	27
logged_in	16.000	15.708	20	20	12	26	26.000	12
num_compromised	22.333	21.760	22	22	28	19	19.667	16
root_shell	28.667	28.560	27	28	32	13.5	13.333	26
su_attempted	31.667	30.627	35	35	38	8	10.333	21
num_root	30.667	30.517	30	30	35	10.5	11.333	28
num_file_creations	31.667	31.522	31	31	36	8	10.333	29
num_shells	35.667	35.498	38	38	37	3	6.333	31
num_access_files	33.667	33.643	36	36	34	5	8.333	32
num_outbound_cmds	37.667	37.516	39	39	40	2	4.333	33
is_host_login	38.667	38.520	40	40	41	1	3.333	34
is_guest_login	32.333	32.269	32	33	30	6	9.667	30
count	5.333	5.040	2	2	4	39	36.667	4
srv_count	14.000	13.842	16	15	13	30	28.000	12
serror_rate	14.667	13.859	17	16	11	29	27.333	11
srv_serror_rate	25.333	22.104	26	25	10	16	16.667	10
rerror_rate	27.667	26.933	24	26	22	15	14.333	22
srv_rerror_rate	23.333	22.431	23	24	19	17	18.667	18
same_srv_rate	12.333	10.801	12	10	6	32	29.667	6
diff_srv_rate	16.667	15.326	13	13	25	25	25.333	9
srv_diff_host_rate	30.667	29.564	34	32	20	10.5	11.333	20
dst_host_count	20.000	15.635	14	17	14	20.5	22.000	7
dst_host_srv_count	9.000	6.581	4	4	5	36.5	33.000	3
dst_host_same_srv_rate	9.000	8.067	6	6	15	36.5	33.000	5
dst_host_diff_srv_rate	11.667	7.764	7	7	26	33.5	30.333	3
dst_host_same_src_port_rate	8.667	6.980	5	5	17	38	33.333	4
dst_host_srv_diff_host_rate	19.000	17.816	21	21	29	22	23.000	13
dst_host_serror_rate	20.000	15.821	11	12	9	20.5	22.000	9
dst_host_srv_serror_rate	13.333	12.633	15	14	8	31	28.667	8
dst_host_rerror_rate	17.333	16.150	19	19	27	23.5	24.667	12
dst_host_srv_rerror_rate	28.667	27.111	28	27	18	13.5	13.333	18

Table 3a Feature subsets after applying threshold of 50%, 25%, 10% and $\log_2$ (n)

Threshold-50%							
A.M	**GM**	**MC4**	**MCT**	**R.D**	**S.V**	**Borda**	**Min**
src_bytes	dst_host_srv_count	is_guest_login	dst_host_serror_rate	srv_rerror_rate	dst_host_serror_rate	dst_host_serror_rate	src_bytes
dst_bytes	dst_host_same_srv_rate	serror_rate	same_srv_rate	protocol_type	dst_host_srv_count	dst_host_srv_count	protocol_type
Count	num_compromised	count	protocol_type	num_access_files	num_shells	num_shells	service
dst_host_same_src_port_rate	land	flag	num_access_files	num_root	dst_host_same_srv_rate	dst_host_same_srv_rate	dst_bytes
dst_host_srv_count	is_guest_login	dst_host_serror_rate	srv_rerror_rate	num_failed_logins	num_outbound_cmds	num_outbound_cmds	flag
dst_host_same_srv_rate	count	protocol_type	num_file_creations	logged_in	service	service	dst_host_srv_count
protocol_type	num_file_creations	diff_srv_rate	dst_host_srv_serror_rate	num_file_creations	num_file_creations	num_file_creations	dst_host_diff_srv_rate
Service	dst_bytes	num_compromised	count	root_shell	dst_host_rerror_rate	dst_host_rerror_rate	count
dst_host_diff_srv_rate	su_attempted	dst_host_same_srv_rate	serror_rate	dst_host_same_srv_rate	duration	duration	dst_host_same_src_port_rate

same_srv_rate	dst_host_srv_rerror_rate	dst_host_srv_rerror_rate	dst_host_count	dst_bytes	same_srv_rate	same_srv_rate	dst_host_same_srv_rate
dst_host_srv_serror_rate	dst_host_same_src_port_rate	dst_host_diff_srv_rate	dst_host_srv_rerror_rate	wrong_fragment	dst_host_srv_rerror_rate	dst_host_srv_rerror_rate	same_srv_rate
srv_count	num_shells	hot	su_attempted	su_attempted	wrong_fragment	wrong_fragment	dst_host_count
serror_rate	same_srv_rate	num_root	dst_host_same_src_port_rate	num_compromised	srv_serror_rate	srv_serror_rate	hot
Hot	wrong_fragment	logged_in	srv_count	srv_serror_rate	hot	hot	dst_host_srv_serror_rate
Flag	srv_rerror_rate	duration	is_host_login	dst_host_rerror_rate	dst_host_srv_diff_host_rate	dst_host_srv_diff_host_rate	duration
logged_in	num_failed_logins	dst_host_srv_count	rerror_rate	same_srv_rate	diff_srv_rate	diff_srv_rate	diff_srv_rate
diff_srv_rate	srv_diff_host_rate	srv_serror_rate	srv_diff_host_rate	duration	dst_host_same_src_port_rate	dst_host_same_src_port_rate	dst_host_serror_rate
Duration	srv_count	root_shell	is_guest_login	num_shells	root_shell	root_shell	srv_serror_rate
dst_host_rerror_rate	urgent	land	flag	num_outbound_cmds	num_compromised	num_compromised	serror_rate
dst_host_srv_diff_host_rate	src_bytes	dst_host_srv_diff_host_rate	src_bytes	dst_host_srv_count	srv_count	srv_count	logged_in
dst_host_count	num_outbound_cmds	num_shells	hot	dst_host_serror_rate	is_host_login	is_host_login	srv_count

Table 3b **Feature subsets after applying threshold of 25%**

Threshold-25%							
A.M	GM	MC4	MCT	R.D	S.V	Borda	Min
src_bytes	dst_host_srv_count	is_guest_login	dst_host_serror_rate	srv_rerror_rate	dst_host_serror_rate	dst_host_serror_rate	src_bytes
dst_bytes	dst_host_same_srv_rate	serror_rate	same_srv_rate	protocol_type	dst_host_srv_count	dst_host_srv_count	protocol_type
Count	num_compromised	count	protocol_type	num_access_files	num_shells	num_shells	service
dst_host_same_src_port_rate	land	flag	num_access_files	num_root	dst_host_same_srv_rate	dst_host_same_srv_rate	dst_bytes
dst_host_srv_count	is_guest_login	dst_host_serror_rate	srv_rerror_rate	num_failed_logins	num_outbound_cmds	num_outbound_cmds	flag
dst_host_same_srv_rate	count	protocol_type	num_file_creations	logged_in	service	service	dst_host_srv_count
protocol_type	num_file_creations	diff_srv_rate	dst_host_srv_serror_rate	num_file_creations	num_file_creations	num_file_creations	dst_host_diff_srv_rate
Service	dst_bytes	num_compromised	count	root_shell	dst_host_rerror_rate	dst_host_rerror_rate	count
dst_host_diff_srv_rate	su_attempted	dst_host_same_srv_rate	serror_rate	dst_host_same_srv_rate	duration	duration	dst_host_same_src_port_rate
same_srv_rate	dst_host_srv_rerror_rate	dst_host_srv_rerror_rate	dst_host_count	dst_bytes	same_srv_rate	same_srv_rate	dst_host_same_srv_rate

Table 3c Feature subsets after applying threshold of 10%

Threshold-10%							
A.M	GM	MC4	MCT	R.D	S.V	Borda	Min
src_bytes	dst_host_srv_count	is_guest_login	dst_host_serror_rate	srv_rerror_rate	dst_host_serror_rate	dst_host_serror_rate	src_bytes
dst_bytes	dst_host_same_srv_rate	serror_rate	same_srv_rate	protocol_type	dst_host_srv_count	dst_host_srv_count	protocol_type
count	num_compromised	count	protocol_type	num_access_files	num_shells	num_shells	service
dst_host_same_src_port_rate	land	flag	num_access_files	num_root	dst_host_same_srv_rate	dst_host_same_srv_rate	dst_bytes

Table 3d Feature subsets after applying threshold of 10%

Threshold-$\log_2(n)$							
A.M	GM	MC4	MCT	R.D	S.V	Borda	Min

src_bytes	dst_host_srv_count	is_guest_login	dst_host_serror_rate	srv_rerror_rate	dst_host_serror_rate	dst_host_serror_rate	src_bytes
dst_bytes	dst_host_same_srv_rate	serror_rate	same_srv_rate	protocol_type	dst_host_srv_count	dst_host_srv_count	protocol_type
count	num_compromised	count	protocol_type	num_access_files	num_shells	num_shells	service
dst_host_same_src_port_rate	land	flag	num_access_files	num_root	dst_host_same_srv_rate	dst_host_same_srv_rate	dst_bytes
dst_host_srv_count	is_guest_login	dst_host_serror_rate	srv_rerror_rate	num_failed_logins	num_outbound_cmds	num_outbound_cmds	flag

4.2 Multi-Class Classification using Multi level Gated Recurrent model (M-GRU)

The features selected are provide as input into Multi level Gated Recurrent model (M-GRU) classification model. GRU helps to alleviate the vanishing gradient problem of recurrent neural network. It consists of UPDATE and RESET gates. The update gate aids the model in identifying how much historic data must be passed along through the future. The reset gate is mostly used to determine how much information from the past should be discarded [34]. In Multi-level GRU model multiple GRU cells are stacked over each other that improve the accuracy of the model.

The input is fed into the stacked GRU layers; the output layer goes through a softmax function for performing multi-class classification. In each GRU layer activation function sigmoid is replaced by ReLu activation function. Sigmoid function has vanishing gradient problem and slow convergence but ReLu provides improvement in convergence. The multilevel GRU is depicted in Figure 2.

The update gate in a GRU cell for time t is calculated by

$$z_t=R(W^{(z)}x_t+U^{(z)}h_{t-1}) \quad \text{---------- (13)}$$

Where x_t denotes input value at time t, $W^{(z)}$ represents weight of x_t, h_{t-1} represents information at t-1, $U^{(z)}$ denotes Weight of h_{t-1} and R denotes ReLu Activation function.

The reset gate can be used in the model to forget some information. It is given by

$$r_t=R(W^{(r)}x_t+U^{(r)}h_{t-1}) \quad \text{---------- (14)}$$

New memory content is added and reset gate will store most important information

$$h'_t=\tanh (Wx_t+[r_t*Uh_{t-1}]) \quad \text{---------- (15)}$$

where tanh –activation function.

h_t is calculated which gives the current information and passes to the network.

$$h_t=[Z_t* h_{t-1}] +[(1-Z_t)*h_t] \quad \text{---------- (16)}$$

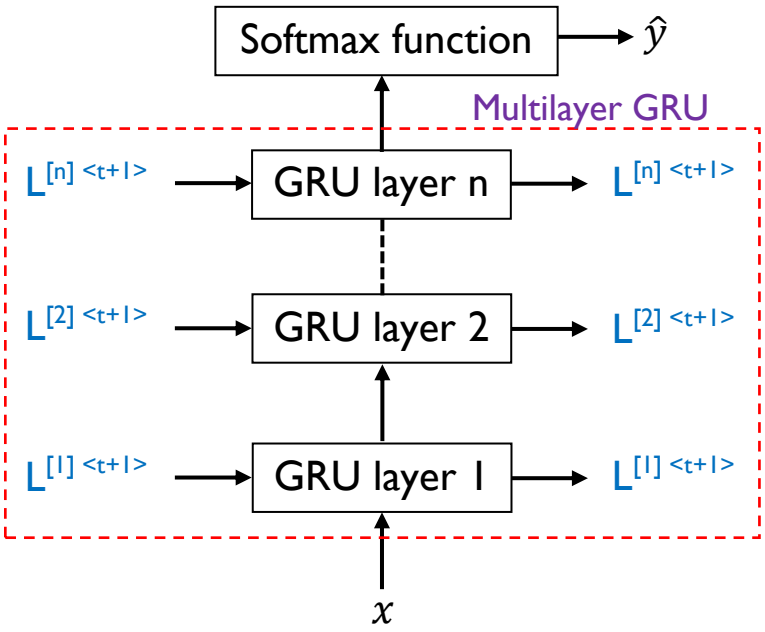


Figure 2 Multi level Gated Recurrent model (M-GRU)

4.3 Algorithm

FS-Ensemble
Input: Training dataset TrainD = {u_1, u_2, ... , u_{n-1}, L_n}, Testing dataset TestD={v_1, v_2, ... , v_{n-1}, L_n}
Output: Attack Classification result of TrainD
Step 1: Obtain feature sets by rankingFS algorithms
1: TrainD=rand.shuffle(Train_D)
1: for Algorithm $F_{(i)}$ in {LightBGM,Catboost,XGBOOST,MutualInformation,Extra tree Classifier}
2: Give TrainD datasetin feature selection algorithms$F_{(i)}$
3: Ranked features with the result of $F_{(i)}$
4: Return the ranked feature list $FR_{(i)}$
Step 2: Obtain$reducedfeaturelistRF_{(i)}$
5: For $FR_{(i)}$ in GMM clustering
6: Return the reduced feature list $RF_{(i)}$
Step 3: Obtain the aggregated feature subsetAF_i
7: $RF_{(i)}$ in Aggregation algorithm={Arithmetic mean, Geometric Mean, MC4, MCT,Random Dictator, Score Voting,Borda,Min}
8: Return the Aggregated feature subsetAF_i
Step 4:Acquire best feature subset according to threshold
9: Threshold (T) = {100%, 50%, 25&, 10%, $\log_2(n)$} in AF_i
10: Put the first T % features in AF to FSbestsubset (FS → FSbestsubset)
11: Return FSbestsubset
Step 5: Obtain attack classification in TestD
12: for classifiers C=M-GRU inTestD do
13: Choose the T% features from FSbestsubset
14: learn C based on TestD
15: return the attack classification
Step 6: Acquire comparison result
18: for classifiers C_i= {GRU,LSTM,RF,Naïve Bayes} inTestD do
19: Choose the T% features from FSbestsubset
20:learn C_i based on TestD
21: Attack classification obtained from C_i compared with C

5 Performance Analysis

The experimental investigation was carried out with python which includes many packages of scikit, keras with the testing dataset of NSL-KDD. The test dataset consists of 22543 records in which 9710 records are normal,7460 are denial of service attack,2421 are probing attack, 2885 R2L attacks and 67 are U2R attacks.

Performance metrics

The performance of IDS is assessed by its ability to classify dataset from network traffic into its correct type. The performance metric is generally derived from the following attributes of confusion matrix denoting the actual and predicted class. True Positive, False Positive, True Negative, False Negative are the four attributes of a confusion matrix [35].

$$\text{Accuracy}=\frac{True\ Positive+True\ Negative}{True\ Positive+True\ Negative+False\ Positive+False\ Negative} \quad \text{---------- (17)}$$

$$\text{False Alarm Rate}=\frac{True\ \ Negative}{True\ Negative+False\ Positive} \quad \text{---------- (18)}$$

$$\text{Precision}=\frac{True\ Positive}{True\ Positive+False\ Positive} \quad \text{---------- (19)}$$

$$\text{Recall}=\frac{True\ Positive}{True\ Positive+False\ Negative} \quad \text{---------- (20)}$$

$$\text{F1-score}=2.\frac{Precision.Recall}{Precision.Recall} \quad \text{---------- (21)}$$

In the feature selection stage, important components are determined by using the suggested Clustered Ensemble Feature Selection technique to obtain reduced dataset. Next, from the original features, candidate features are selected for the following stage. Table 3 lists the numbers and names of the obtained features of NSL-KDD dataset. The method appears to dramatically reduce dimensionality and eradicate the dataset's unnecessary features. Eventually, an M-GRU classifier which has multiple layers of modified GRU classifier is employed that dramatically increases IDS prediction performance.

The classification results are assessed to see how well our ensemble feature selection strategy has performed. The hyperparameter for M-GRU are tabulated in table 4.

Table 4 Hyper-parameters of M-GRU

Hyper-Parameter	**Multi-Class Classification**
Learning Rate	0.01
Activation function	Softmax
Optimizer	Stochastic Gradient Descent
Epoch	500
Loss function	categorical_crossentropy

The confusion matrix is obtained for the NSL KDD dataset using the M-GRU algorithm with different thresholds 100%,50%,25%,10%,log2 (n) of aggregation methods A.M,G.M, MC4, MCT, R.D, S.V, Borda, Min and it compares the effectiveness of the proposed technique to several state-of-the-art techniques in terms of performance parameters such as accuracy, false alarm rate, precision and recall. The mathematical computations of the used assessment metrics are explained in equation 17 to 20.

Table 5 Confusion matrix for M-GRU with different thresholds of A.M aggregation

Threshold level	**Attack Class**	**Normal**	**DoS**	**Probing**	**R2L**	**U2R**
100%	Normal	9526	17	75	58	34
	DoS	510	6224	355	112	259
	Probing	511	107	1644	66	93
	R2L	525	155	139	1964	102
	U2R	11	3	7	8	38
50%	Normal	9366	49	161	75	59
	DoS	562	6224	186	296	192
	Probing	529	65	1664	88	75
	R2L	219	44	39	2564	19
	U2R	13	2	7	16	29
25%	Normal	9648	19	11	21	11
	DoS	153	7001	143	96	67

	Probing	119	44	2101	93	64
	R2L	56	45	37	2702	47
	U2R	13	2	0	1	51
10%	Normal	9462	49	85	83	31
	DoS	586	6430	166	159	119
	Probing	434	112	1654	128	93
	R2L	153	35	9	2684	4
	U2R	11	3	9	3	41
$\log_2$ (n)	Normal	9462	49	85	83	31
	DoS	477	6389	179	259	156
	Probing	797	6	1607	2	9
	R2L	236	37	29	2564	19
	U2R	15	0	8	6	38

The confusion matrix for all five classes with threshold values 100%,50%,25%,10%,log2 (n) of Arithmetic mean aggregation of M-GRU algorithm are generated which is tabulated in table 5. Tables 6 and 7 highlight the average accuracy and false alarm rate performance of M-GRU using the NSL KDD dataset for various threshold levels. It's been claimed that the M-GRU classifier is best in various measures if feature selection of 25% threshold is used.

Table 6 Accuracy of M-GRU for different thresholds of A.M aggregation

Threshold level	**Attack Class**	**Accuracy**				
100%	Normal	0.923	0.911	0.889	0.781	0.768
	DoS	0.933	0.924	0.912	0.823	0.813
	Probing	0.940	0.943	0.926	0.853	0.851
	R2L	0.948	0.941	0.926	0.881	0.882
	U2R	0.977	0.980	0.964	0.983	0.985
50%	Normal	0.926	0.896	0.894	0.748	0.769
	DoS	0.938	0.939	0.922	0.808	0.814
	Probing	0.949	0.945	0.941	0.847	0.851
	R2L	0.965	0.972	0.962	0.870	0.882
	U2R	0.983	0.989	0.982	0.977	0.984
25%	Normal	0.982	0.947	0.942	0.890	0.817
	DoS	0.975	0.957	0.954	0.910	0.826
	Probing	0.977	0.963	0.962	0.927	0.859
	R2L	0.982	0.979	0.972	0.924	0.902
	U2R	0.991	0.986	0.985	0.963	0.984
10%	Normal	0.936	0.940	0.907	0.756	0.758
	DoS	0.945	0.945	0.938	0.779	0.781
	Probing	0.954	0.947	0.946	0.836	0.834
	R2L	0.975	0.971	0.973	0.880	0.881
	U2R	0.988	0.980	0.984	0.966	0.978
$\log_2$ (n)	Normal	0.921	0.915	0.904	0.759	0.760
	DoS	0.948	0.941	0.922	0.781	0.758
	Probing	0.951	0.950	0.929	0.836	0.821
	R2L	0.970	0.971	0.956	0.879	0.820
	U2R	0.989	0.989	0.979	0.974	0.949

Table 7 False Alarm Rate of M-GRU for different thresholds of A.M aggregation

Algorithms	Attack Class	False Alarm Rate				
100%	Normal	0.121	0.129	0.108	0.311	0.318
	DoS	0.019	0.031	0.042	0.102	0.113
	Probing	0.029	0.024	0.042	0.069	0.071
	U2L	0.012	0.019	0.035	0.021	0.020
	R2L	0.022	0.018	0.034	0.014	0.013
50%	Normal	0.103	0.142	0.121	0.316	0.318
	DoS	0.011	0.009	0.033	0.119	0.112
	Probing	0.020	0.023	0.028	0.074	0.071
	U2L	0.024	0.013	0.020	0.033	0.021
	R2L	0.015	0.010	0.016	0.020	0.013
25%	Normal	0.005	0.037	0.053	0.089	0.095
	DoS	0.007	0.018	0.032	0.047	0.090
	Probing	0.009	0.015	0.016	0.041	0.060
	U2L	0.011	0.015	0.018	0.034	0.046
	R2L	0.008	0.013	0.014	0.035	0.014
10%	Normal	0.021	0.026	0.059	0.188	0.182
	DoS	0.013	0.015	0.024	0.138	0.137
	Probing	0.013	0.020	0.021	0.090	0.092
	U2L	0.019	0.020	0.018	0.047	0.048
	R2L	0.011	0.018	0.015	0.032	0.020
$\log_2$ (n)	Normal	0.021	0.022	0.066	0.183	0.196
	DoS	0.006	0.008	0.030	0.135	0.141
	Probing	0.015	0.015	0.034	0.091	0.105
	U2L	0.018	0.017	0.030	0.051	0.087
	R2L	0.010	0.009	0.019	0.024	0.048

It is evident that the arithmetic mean aggregation method outperforms all the other aggregation methods.

Figure 3 Accuracy of M-GRU for different thresholds of various aggregation methods

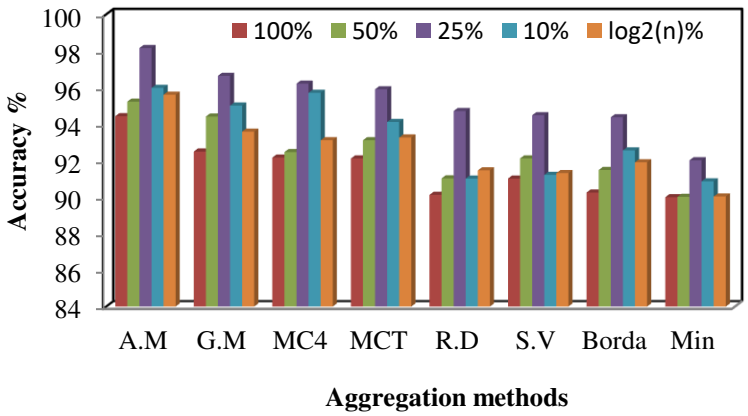


Figure 4 FAR of M-GRU for different thresholds of various aggregation methods

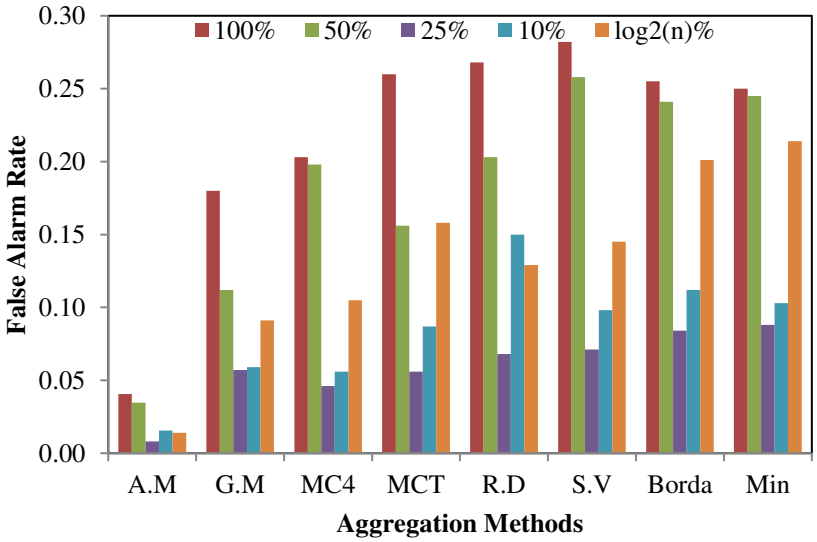


Figure 5 Accuracy of GRU for different thresholds of various aggregation methods

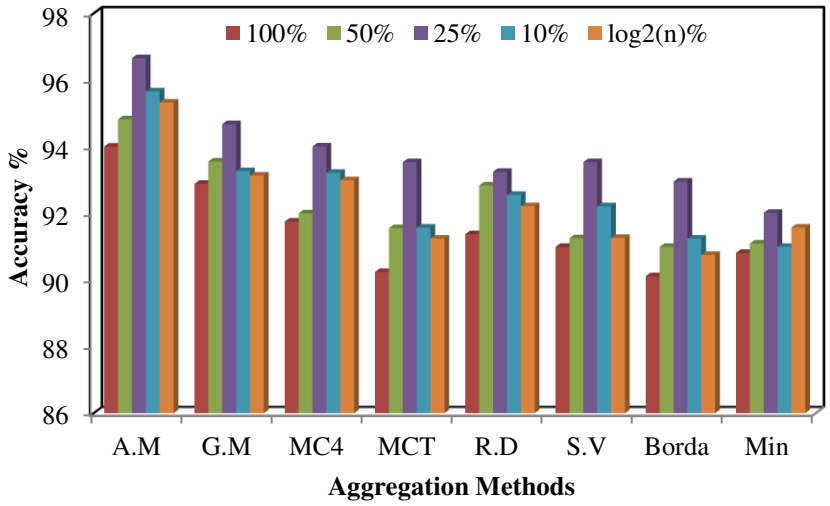


Figure 6: FAR of GRU for different thresholds of various aggregation methods

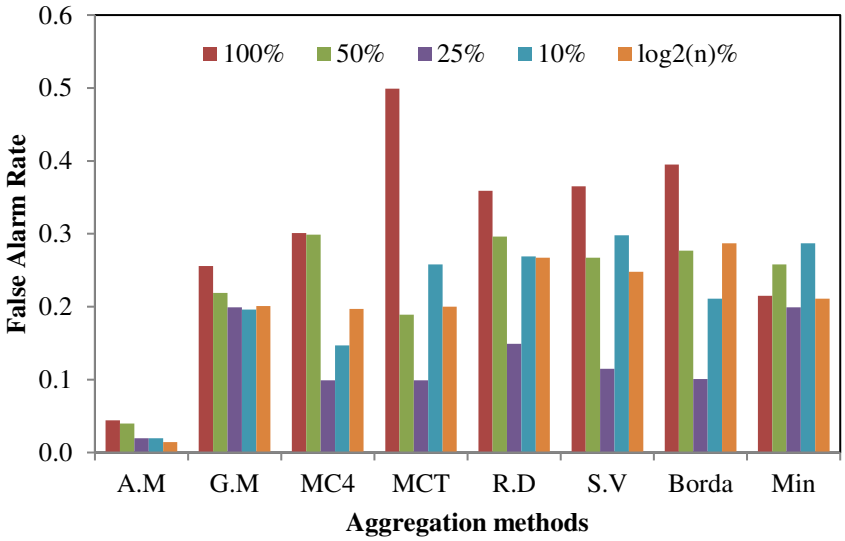


Figure 7 Accuracy of LSTM for different thresholds of various aggregation methods

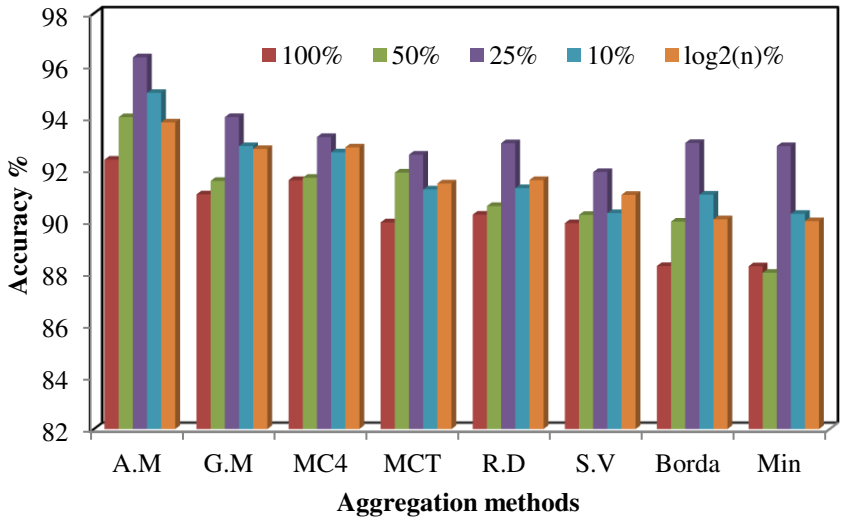


Figure 8 FAR of LSTM for different thresholds of various aggregation methods

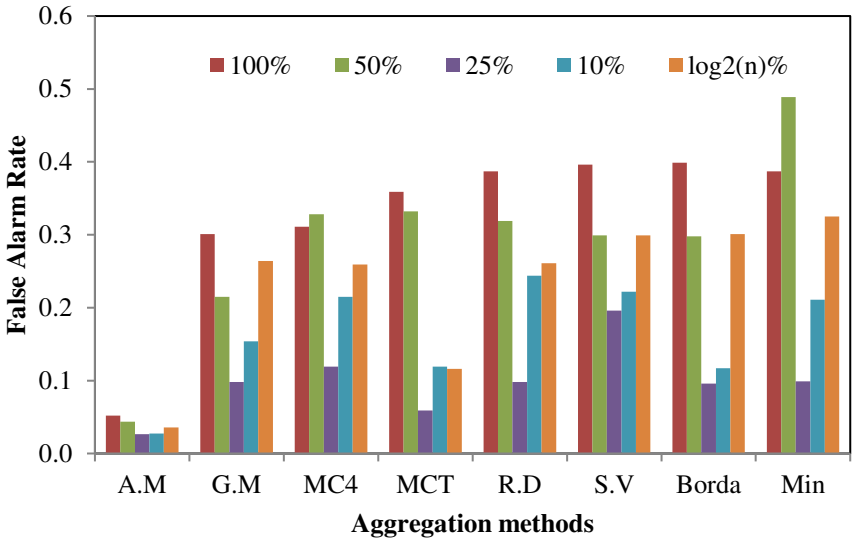


Figure 9 Accuracy of RF for different thresholds of various aggregation methods

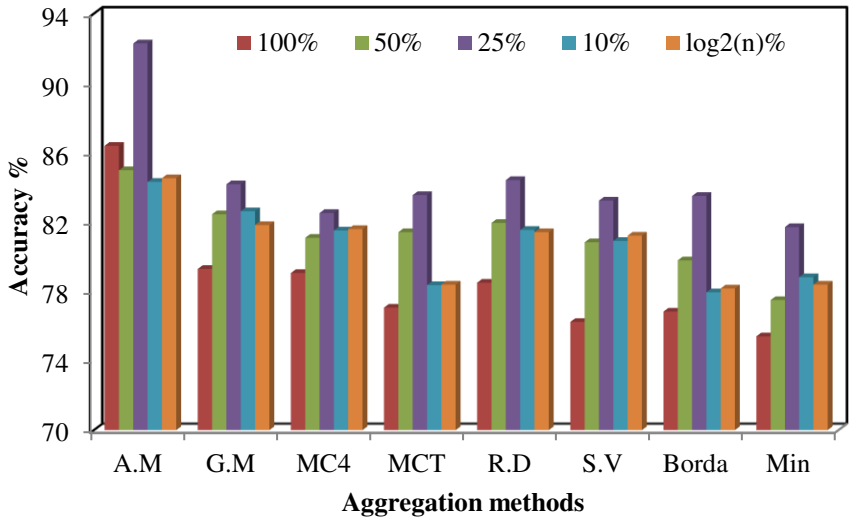


Figure 10 FAR of RF for different thresholds of various aggregation methods

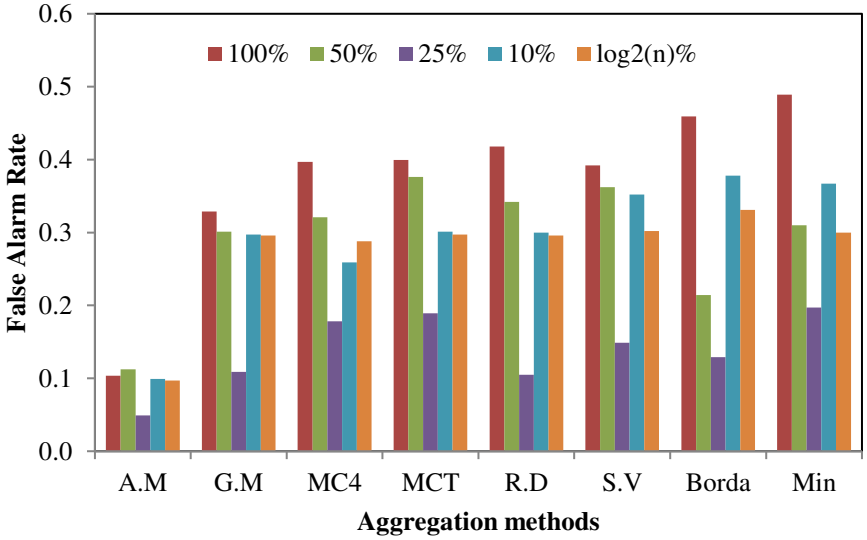


Figure 11 Accuracy of Naïve Bayes for different thresholds of various aggregation methods

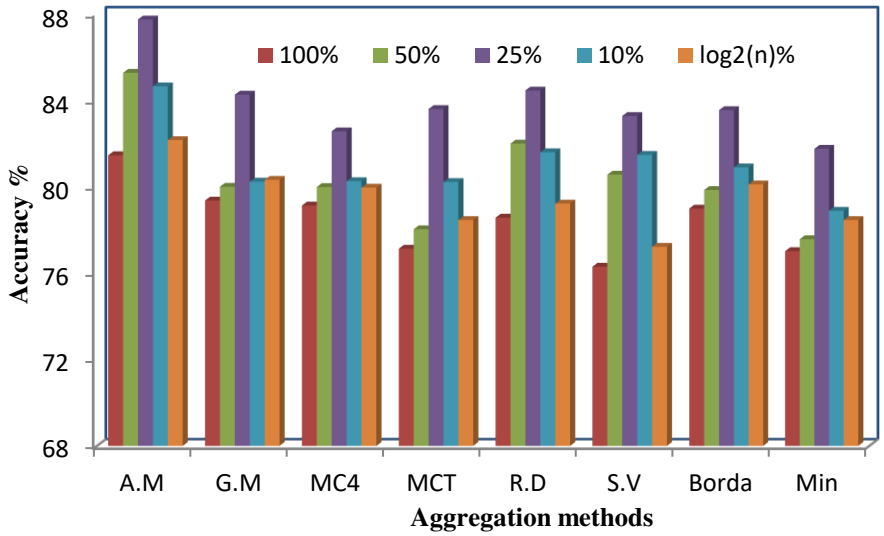
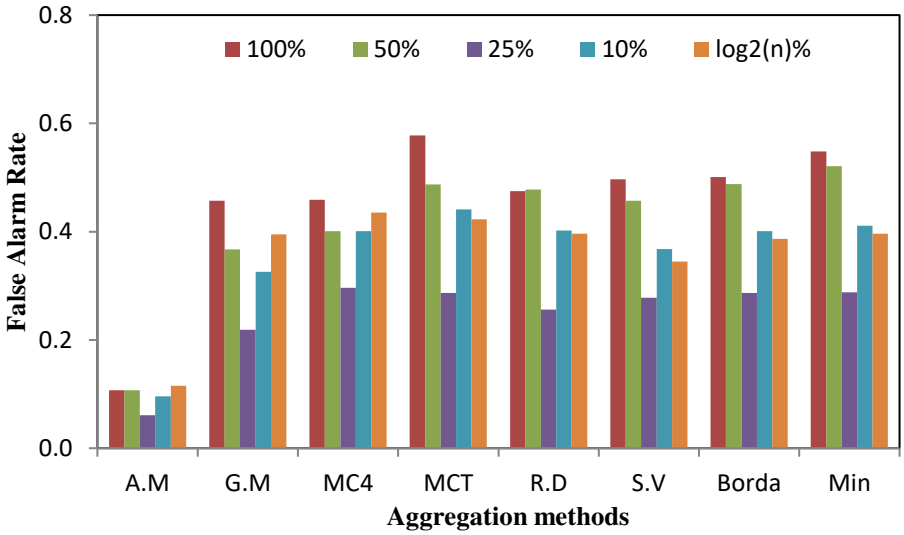


Figure 12 FAR of Naïve Bayes for different thresholds of various aggregation methods



To assess the efficacy of the proposed IDS, we compare the M-GRU classification model to other machine learning models in order to identify malicious from benign occurrences. The average values of key metrics, such as Acc, accuracy, DR, F-Measure, and ADR, have increased dramatically as a result of the suggested CFS-BA algorithm's identification of relevant characteristics.

Figure 13 Accuracy % of different aggregation algorithm

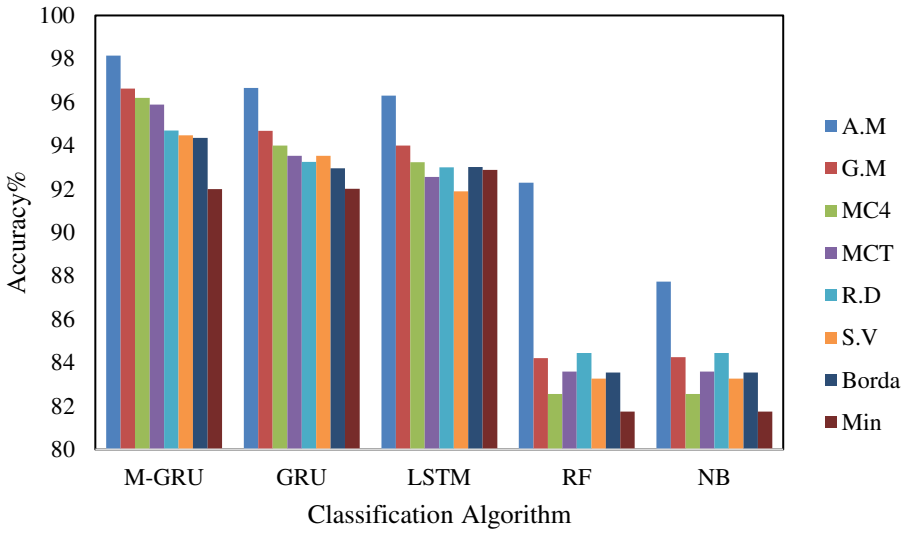


Figure 14 FAR of different aggregation algorithm

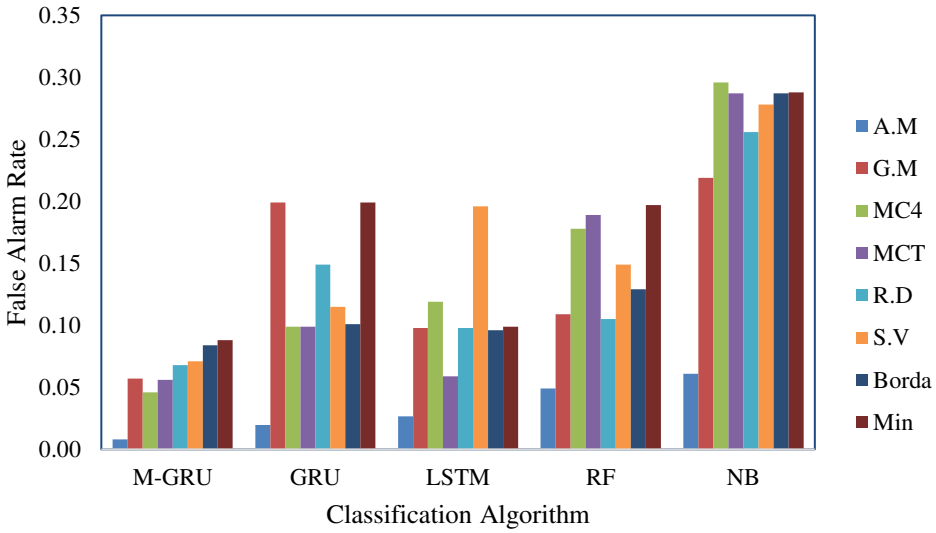


Figure 15 Comparison of Accuracy% of M-GRU, GRU, LSTM, RF, NB

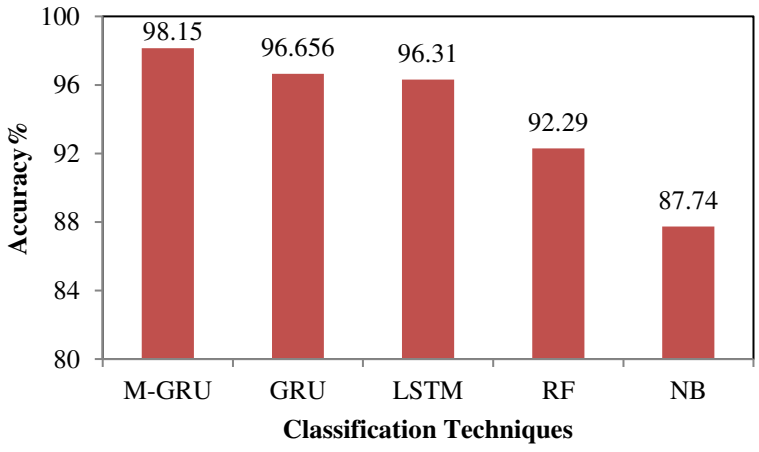
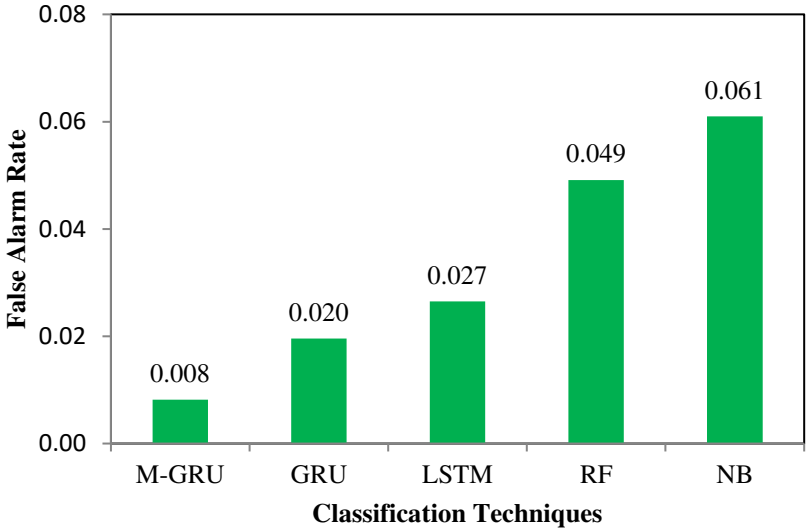


Figure 16 FAR of M-GRU, GRU, LSTM, RF, NB



The following are the inferences from the experimental analysis,

Finding1:
Figure 3,5,7,9,11 illustrates the average accuracy% of M-GRU, GRU,LSTM,RF,NB algorithms for A.M,G.M, MC4, MCT, R.D, S.V, Borda, Min aggregation methods with different thresholds. It is inferred from the M-GRU multi-classification model validation that 25% threshold of Arithmetic mean aggregation produces higher accuracy of 98.21% for normal class, 97.476 % for DoS, 97.733% for Probe. 98.243 % for R2L and 99.09% for U2R.The 25% threshold feature list of M-GRU is compared with all the other threshold values and it is evident that 25% produces higher accuracy of 5.935%, 4.210%, 3.735%,3.411%,1.384% than 100% threshold of normal, DoS, Probe, R2L, U2R classes respectively,5.607%, 3.669%, 2.835%, 1.775%, 0.790% higher accuracy than 50% threshold of normal, DoS, Probe, R2L, U2R classes respectively. 4.565%, 2.928%, 2.329%, 0.790%, 0.302% higher accuracy than 10% threshold of normal, DoS, Probe, R2L, U2R classes respectively. 6.077%, 2.635%, 2.680%, 1.220%, 0.173% higher accuracy than $\log_2$ (n) threshold of normal, DoS, Probe, R2L, and U2R classes respectively.

Finding 2:
Figure 4,6,8,10,12 illustrates the average accuracy% of M-GRU, GRU, LSTM, RF, NB algorithms for A.M, G.M, MC4, MCT, R.D, S.V, Borda, Min aggregation methods with different thresholds. It is evident that the M-GRU multi-classification model validation that 25% threshold of Arithmetic mean aggregation produces low false alarm rate of 0.005 for normal class, 0.007 for DoS, 0.009 for Probe. 0.003 for R2L and 0.008 for U2R class. The 25% threshold feature list of M-GRU is compared with all the other threshold values which produces low false alarm rate with difference of 0.116,0.011,0.019,0.002,0.013for 100% threshold of normal, DoS, Probe, R2L, U2R classes respectively. 0.098 ,0.003 ,0.010 ,0.013 ,0.007 low false alarm rate with difference for 50% threshold of normal, DoS, Probe, R2L, U2R classes respectively. 0.016, 0.006, 0.004, 0.008, 0.003 low false alarm rate with difference for 10% threshold of normal, DoS, Probe, R2L, U2R classes respectively. 0.017,0.001,0.005,0.007,0.001low false alarm rate with difference for $\log_2$ (n) threshold of normal, DoS, Probe, R2L, U2R classes respectively.

Finding 3:
Accuracy % of and FAR of 25% threshold for M-GRU, GRU, LSTM, Random Forest, Naïve Bayes classification with different aggregation such as A.M, G.M, MC4, MCT, Random Dictator, Score Voting, Borda, Min is depicted in graph 13 and 14. M-GRU produces 98.151% accuracy for A.M aggregation and 96.630%, 96.200%, 95.890%, 94.700%, 94.470%, 94.360%, 92% for G.M, MC4, MCT, Random Dictator, Score Voting, Borda, Min methods. GRU produces 96.656% accuracy for A.M aggregation and 94.68%, 94.01%, 93.54%, 93.25%, 93.54%, 92.96%, 92.02% for G.M, MC4, MCT, Random Dictator, Score Voting, Borda, Min methods. LSTM produces 96.31% accuracy for A.M aggregation and 96.31%, 94.01%, 93.24%, 92.56%, 93%, 91.89%, 93.01%, 92.89% for G.M, MC4, MCT, Random Dictator, Score Voting, Borda, Min methods. RF produces 92.29 accuracy for A.M aggregation and 84.21%, 82.56%, 83.59%, 84.45%, 83.27%, 83.54%, 81.75 % for G.M, MC4, MCT, Random Dictator, Score Voting, Borda, Min methods. NB produces 87.74% accuracy for A.M aggregation and 84.26%, 82.56%, 83.59, 84.45%, 83.27%, 83.54%, 81.75% for G.M, MC4, MCT, Random Dictator, Score Voting, Borda, Min methods.
M-GRU produces 0.008 FAR for A.M aggregation and 0.057, 0.046, 0.056, 0.068, 0.071, 0.084, 0.088 for G.M, MC4, MCT, Random Dictator, Score Voting, Borda, Min methods. GRU produces 0.020 FAR for A.M

aggregation and 0.199, 0.099, 0.099, 0.149, 0.115, 0.101, 0.199 for G.M, MC4, MCT, Random Dictator, Score Voting, Borda, Min methods. LSTM produces 0.027 FAR for A.M aggregation and 0.098, 0.119, 0.059, 0.098, 0.196, 0.096, 0.099 for G.M, MC4, MCT, Random Dictator, Score Voting, Borda, Min methods. RF produces 0.049 FAR for A.M aggregation and 0.109, 0.178, 0.189, 0.105, 0.149, 0.129, 0.197 for G.M, MC4, MCT, Random Dictator, Score Voting, Borda, Min methods. NB produces 0.061 FAR for A.M aggregation and 0.219, 0.296, 0.287, 0.256, 0.278, 0.287, 0.288 for G.M, MC4, MCT, Random Dictator, Score Voting, Borda, Min methods.

It is inferred that the arithmetic aggregation outperforms all the other aggregation methods in terms of accuracy and false alarm rate.

Finding 4:

By comparing M-GRU with GRU, LSTM, Random Forest, Naïve Bayes, it is inferred that 25% threshold of A. M aggregation of M-GRU outperforms all the other algorithms with respect to accuracy and 98.15% and 96.656%, 96.31%, 92.29%, 87.74% for GRU, LSTM, Random Forest, Naïve Bayes respectively. False alarm rate of 0.008 for M-GRU and 0.020, 0.027, 0.049, 0.061 for GRU, LSTM, Random Forest, Naïve Bayes respectively is shown in figure 15 and 16 respectively.

6 Conclusion and Future Enhancement

Despite the fact that various machine learning techniques have indeed been presented to effectiveness of IDSs, existing classification algorithms still struggle to reach good result. This paper first introduces a novel clustered ensemble based feature selection method which helps in picking the favorable feature subset list which indeed improves the performance of classification algorithm. NSL-KDD training and testing dataset used in this work. Then a multi-level modified gated recurrent unit algorithm (M-GRU) is proposed which with the help ensemble feature selection produces better accuracy and low false alarm rate. The experiments show promise in terms of classification accuracy and average false alarm rate.

Results show that M-GRU, with average accuracy of 98.15% and 0.008 average false alarm rate with the subset of 10 features (25% threshold) when arithmetic mean aggregation is employed. In M-GRU different threshold of feature subsets were studied with different aggregate methods and the analysis shows that 25% threshold subset of arithmetic mean aggregation method outperforms the other threshold subset of all the other aggregation methods. M-GRU algorithm outperforms other classification algorithm in performance on criteria such as accuracy and false alarm rate. For future research real time intrusion detection is with real industrial control systems dataset can be made use of.

Author contribution M Karthigha: Conceptualization, Methodology, Software, Validation, Investigation, Resources, Data curation, Writing - original draft, Visualization, Dr. L Latha: Supervision

Declaration of competing interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding The author(s) received no financial support for the research, authorship, and/or publication of this article.

7 References

[1] K. Yang, Q. Li, X. Lin, X. Chen, and L. Sun, "IFinger: Intrusion Detection in Industrial Control Systems via Register-Based Fingerprinting," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 955–967, 2020, doi: 10.1109/JSAC.2020.2980921.

[2] M. Kaouk, J. M. Flaus, M. L. Potet, and R. Groz, "A review of intrusion detection systems for industrial control systems," *2019 6th Int. Conf. Control. Decis. Inf. Technol. CoDIT 2019*, pp. 1699–1704, 2019, doi: 10.1109/CoDIT.2019.8820602.

[3] Y. Hu, A. Yang, H. Li, Y. Sun, and L. Sun, "A survey of intrusion detection on industrial control systems," *Int. J. Distrib. Sens. Networks*, vol. 14, no. 8, 2018, doi: 10.1177/1550147718794615.

[4] A. Ayodeji, Y. kuo Liu, N. Chao, and L. qun Yang, "A new perspective towards the development of robust data-driven intrusion detection for industrial control systems," *Nucl. Eng. Technol.*, vol. 52, no. 12, pp. 2687–2698, 2020, doi: 10.1016/j.net.2020.05.012.

[5] A. Gül and E. Adali, "A feature selection algorithm for IDS," *2nd Int. Conf. Comput. Sci. Eng. UBMK 2017*, pp. 816–820, 2017, doi: 10.1109/UBMK.2017.8093538.

[6] H. Alazzam, A. Sharieh, and K. E. Sabri, "A feature selection algorithm for intrusion detection system based on Pigeon Inspired Optimizer," *Expert Syst. Appl.*, vol. 148, p. 113249, 2020, doi: 10.1016/j.eswa.2020.113249.

[7] F. E. Ayo, S. O. Folorunso, A. A. Abayomi-Alli, A. O. Adekunle, and J. B. Awotunde, "Network intrusion detection based on deep learning model optimized with rule-based hybrid feature selection," *Inf. Secur. J.*, vol. 29, no. 6, pp. 267–283, 2020, doi: 10.1080/19393555.2020.1767240.

[8] M. Alirezanejad, R. Enayatifar, H. Motameni, and H. Nematzadeh, "Heuristic filter feature selection methods for medical datasets," *Genomics*, vol. 112, no. 2, pp. 1173–1181, 2020, doi: 10.1016/j.ygeno.2019.07.002.

[9] M. Reazul, A. Rahman, and T. Samad, "A Network Intrusion Detection Framework based on Bayesian Network using Wrapper Approach," *Int. J. Comput. Appl.*, vol. 166, no. 4, pp. 13–17, 2017, doi: 10.5120/ijca2017913992.

[10] H. Liu, M. Zhou, and Q. Liu, "An embedded feature selection method for imbalanced data classification," *IEEE/CAA J. Autom.*

Sin., vol. 6, no. 3, pp. 703–715, 2019, doi: 10.1109/JAS.2019.1911447.
[11] V. Bolón-Canedo and A. Alonso-Betanzos, "Ensembles for feature selection: A review and future trends," *Inf. Fusion*, vol. 52, pp. 1–12, 2019, doi: 10.1016/j.inffus.2018.11.008.
[12] M. Abdullah, A. Balamash, A. Al-Shannaq, and S. Almabdy, "Enhanced Intrusion Detection System using Feature Selection Method and Ensemble Learning Algorithms," *Int. J. Comput. Sci. Inf. Secur.*, vol. 16, no. December, pp. 48–55, 2018.
[13] C. Khammassi and S. Krichen, "A GA-LR wrapper approach for feature selection in network intrusion detection," *Comput. Secur.*, vol. 70, pp. 255–277, 2017, doi: 10.1016/j.cose.2017.06.005.
[14] A. J. Malik, W. Shahzad, and F. A. Khan, "Network intrusion detection using hybrid binary PSO and random forests algorithm," *Secur. Commun. Networks*, vol. 8, no. 16, pp. 2646–2660, 2015, doi: 10.1002/sec.508.
[15] N. T. Pham, E. Foo, S. Suriadi, H. Jeffrey, and H. F. M. Lahza, "Improving performance of intrusion detection system using ensemble methods and feature selection," *ACM Int. Conf. Proceeding Ser.*, 2018, doi: 10.1145/3167918.3167951.
[16] A. K. Shukla, "Building an effective approach toward intrusion detection using ensemble feature selection," *Int. J. Inf. Secur. Priv.*, vol. 13, no. 3, pp. 31–47, 2019, doi: 10.4018/IJISP.201907010102.
[17] W. He, H. Li, and J. Li, "Ensemble feature selection for improving intrusion detection classification accuracy," *ACM Int. Conf. Proceeding Ser.*, pp. 28–33, 2019, doi: 10.1145/3349341.3349364.
[18] S. Krishnaveni, S. Sivamohan, S. S. Sridhar, and S. Prabakaran, "Efficient feature selection and classification through ensemble method for network intrusion detection on cloud computing," *Cluster Comput.*, vol. 24, no. 3, pp. 1761–1779, 2021, doi: 10.1007/s10586-020-03222-y.
[19] F. Salo, A. B. Nassif, and A. Essex, "Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection," *Comput. Networks*, vol. 148, pp. 164–175, 2019, doi: 10.1016/j.comnet.2018.11.010.
[20] J. Ling, Z. Zhu, Y. Luo, and H. Wang, "An intrusion detection method for industrial control systems based on bidirectional simple recurrent unit," *Comput. Electr. Eng.*, vol. 91, no. February 2020, p. 107049, 2021, doi: 10.1016/j.compeleceng.2021.107049.
[21] M. Ge, N. F. Syed, X. Fu, Z. Baig, and A. Robles-Kelly, "Towards a deep learning-driven intrusion detection approach for Internet of Things," *Comput. Networks*, vol. 186, 2021, doi: 10.1016/j.comnet.2020.107784.
[22] B. Ingre and A. Yadav, "Performance analysis of NSL-KDD dataset using ANN," *Int. Conf. Signal Process. Commun. Eng. Syst. - Proc. SPACES 2015, Assoc. with IEEE*, pp. 92–96, 2015, doi: 10.1109/SPACES.2015.7058223.
[23] N. Paulauskas and J. Auskalnis, "Analysis of data pre-processing influence on intrusion detection using NSL-KDD dataset," *2017 Open Conf. Electr. Electron. Inf. Sci. eStream 2017 - Proc. Conf.*, pp. 1–5, 2017, doi: 10.1109/eStream.2017.7950325.
[24] G. Ke *et al.*, "LightGBM: A highly efficient gradient boosting decision tree," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, no. Nips, pp. 3147–3155, 2017.
[25] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: Unbiased boosting with categorical features," *Adv. Neural Inf. Process. Syst.*, vol. 2018-Decem, no. Section 4, pp. 6638–6648, 2018.
[26] A. V. Dorogush, V. Ershov, and A. Gulin, "CatBoost: gradient boosting with categorical features support," pp. 1–7, 2018, [Online]. Available: http://arxiv.org/abs/1810.11363.
[27] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, vol. 13-17-August-2016, pp. 785–794, 2016, doi: 10.1145/2939672.2939785.
[28] W. Gao, L. Hu, and P. Zhang, "Feature redundancy term variation for mutual information-based feature selection," *Appl. Intell.*, vol. 50, no. 4, pp. 1272–1288, 2020, doi: 10.1007/s10489-019-01597-z.
[29] C. M. Weber, D. Ray, A. A. Valverde, J. A. Clark, and K. S. Sharma, "Gaussian mixture model clustering algorithms for the analysis of high-precision mass measurements," *Nucl. Instruments Methods Phys. Res. Sect. A Accel. Spectrometers, Detect. Assoc. Equip.*, vol. 1027, no. 1, pp. 1–12, 2022, doi: 10.1016/j.nima.2021.166299.
[30] Y. Lu, Z. Tian, P. Peng, J. Niu, W. Li, and H. Zhang, "GMM clustering for heating load patterns in-depth identification and prediction model accuracy improvement of district heating system," *Energy Build.*, vol. 190, pp. 49–60, 2019, doi: 10.1016/j.enbuild.2019.02.014.
[31] A. O. Balogun *et al.*, "Empirical analysis of rank aggregation-based multi-filter feature selection methods in software defect prediction," *Electron.*, vol. 10, no. 2, pp. 1–16, 2021, doi: 10.3390/electronics10020179.
[32] X. Li, X. Wang, and G. Xiao, "A comparative study of rank aggregation methods for partial and top ranked lists in genomic applications," *Brief. Bioinform.*, vol. 20, no. 1, pp. 178–189, 2019, doi: 10.1093/bib/bbx101.
[33] A. Hashemi, M. B. Dowlatshahi, and H. Nezamabadi-pour, "MFS-MCDM: Multi-label feature selection using multi-criteria decision making," *Knowledge-Based Syst.*, vol. 206, p. 106365, 2020, doi: 10.1016/j.knosys.2020.106365.
[34] S. Jung, J. Moon, S. Park, and E. Hwang, "An attention-based multilayer gru model for multistep-ahead short-term load forecasting," *Sensors*, vol. 21, no. 5, pp. 1–20, 2021, doi: 10.3390/s21051639.
[35] H. Ji, Y. Wang, H. Qin, Y. Wang, and H. Li, "Comparative performance evaluation of intrusion detection methods for In-Vehicle networks," *IEEE Access*, vol. 6, pp. 37523–37532, 2018, doi: 10.1109/ACCESS.2018.2848106.