Intelligent Automation And Soft Computing, 2020
Vol. 26, no. 6, 795–805
DOI: 10.32604/iasc.2020.010114

A Perspective of the Machine Learning Approach for the Packet Classification in the Software Defined Network

B. Indira[1], K. Valarmathi[2]

[1]Department of Computer Science and Engineering, P.S.R. Engineering College, Sivakasi-626 140
[2]Department of Electronics and Communication Engineering, P.S.R. Engineering College, Sivakasi-626 140

ABSTRACT
Packet classification is a major bottleneck in Software Defined Network (SDN). Each packet has to be classified based on the action specified in each rule in the given flow table. To perform classification, the system requires much of the CPU clock time. Therefore, developing an efficient packet classification algorithm is critical for high speed inter networking. Existing works make use of exact matching, range matching and longest prefix matching for classification and these techniques sometime enlarges rule databases, thus resulting in huge memory consumption and inefficient searching performance. In order to select an efficient packet classification algorithm with less memory consumption and high classification accuracy, Machine Learning (ML) algorithms are used. For performance comparison, ML algorithms are used, namely Multi-layer Perceptron (MLP), K-Nearest Neighbor (KNN), Decision Tree (DT), Random Forest (RF), AdaBoost classifier (AB) and Support Vector Machine (SVM). All these algorithms build network for packet classification and train the network with the use of Access Control List (ACL) netbench dataset. 5-features of IPv4 packet header are used and the algorithms classify the packets based on action/flow of each packet. Experimental results show that among six algorithms, RF algorithm gives better improvement in accuracy performance for permitted packets.

KEYWORDS: Machine learning algorithm, min-max normalization, IPv4 packet header, packet classification.

1 INTRODUCTION

IN the SDN, the Packet classification is a key process hired by Internet routers to classify the packets for identifying various applications like traffic engineering, security monitoring and quality of services, (You, 2017). Gupta & Mckeown (2001) describe that the packet classification for the routing lookup concentrated in both the one dimensional and multi-dimensional fields. Based on the packet destination address (one dimension), the router forwards an incoming packet to the next hop. This is generally done by applying the longest prefix matching algorithms in the designated IP address of the forwarding table. Tactlessly, this matching process is time consuming and costly. In a network, when the packet arrival rate increases, the complexity of the IP lookup and size of the lookup table also increases. Hence, Perex, et al (2014) and Dixit, et al (2012) specified that the IP lookup is a major problem in high speed networks. In this paper instead of a single dimension, the multi-dimensional fields of the IPv4 packet header is chosen for the classification. So, the IP lookup problem is transformed to the packet classification with the use of multiple fields.

In data networks, the packet classification plays an important role and is achieved through some rules stored in a table. Each rule has five fields of the IPv4 packet header and the action must be applied to each packet to match the rule and is explained in Gupta, et al. (1999). Sometimes different rules may match an identical packet and the highest priority rule is applied on the incoming packet to choose the correct action. However, these existing approaches lead to the problem of huge memory consumption, searching time complexity and poor worse case performance (Avudaiammal, et al. (2017)). The packet classification is often the performance bottleneck for internet routers. To overcome the problems, Liu, at al. (2012) and Mrudul, et al. (2017) recommended that,

CONTACT B. Indira aridnib@gmail.com

the Machine learning based packet classification has become the widely adopted solution. Because, the ML approaches offer a better choice in classifying the traffic in the SDN and will overcome some of the limitations of the existing approaches like performance accuracy. Elmahgiubi, et al. (2016) explained that the ML leads to low computational cost and classification time. In this paper, the machine learning algorithm is considered in classifying the packets based on the action or flow of any given incoming packet.

1.1 The Problem Statement

- Assume: Rule set R (The number of rules in the given netbench dataset).
- Select: Feature F (Packet header fields F are shown in Table 1).
- Predict best: Machine Learning Algorithm A (Machine learning classification algorithms).
- Analyse: Performance P (Performance metrics measured when applying the machine learning algorithms to the rule sets).

Table 1. Selection of the 5-features with Flows using the ACL Rule Set.

Rule set	Source IP address	Destination IP address	Source Port	Destination Port	Destination Port	Flow / Action
R1	67.37.131.207	67.37.131.84	1024:65535	161	0x06/0xFF	Permit
R2	16.242.207.72	64.58.35.44	1024:65535	1024:65535	0x06/0xFF	Deny
R3	31.131.76.216	31.131.79.50	1024:65535	1024:65535	0x06/0xFF	Deny
R4	31.248.18.219	64.58.35.44	1024:65535	1024:65535	0x06/0xFF	Permit
R5	171.95.183.11	202.76.70.56	1024:65535	1024:65535	0x06/0xFF	Permit

The problem of the packet classification is achieved such that, for a given ruleset $r \in R$ with five features $\mathbf{f}(r) \in F$, find the best classification model *f(r)* by applying ML algorithms A, such that the selected algorithm $a \in A$ maximizes the performance metrics $p \in P$.

For the neural network, the epochs and the hidden layers are the key parameters to improve the accuracy. The hidden nodes are varied for each epoch. For the K-NN algorithm, *k* is the key parameter to choose the best value to obtain the accuracy of the packets. For the decision tree, the kernel function and number of splits are the key parameters to improve the classification accuracy. In the random forest, choosing the number of trees is a key parameter to build many trees. This method reduces over fitting and is therefore more accurate when compared to other algorithms. The role of the AdaBoost classifier is to boost the performance of the decision tree by choosing the appropriate number of trees and learning rate. In the SVM, the kernel function and gamma are the key parameter to obtain the best accuracy. All these algorithms are implemented using the python. By using the 5-tuple selected features of an ACL (Access Control List) the netbench dataset, this work attempt to find the best classifier from the existing ML algorithms and develops the classification model for the forthcoming intelligent classification system.

1.2 Contribution

The contribution of this paper is to classify the packets in low computation time and improve the performance of the classification rate. The classification takes place based on the exact matching than prefix or range matching. Since the prefix or range matching requires data conversion, which leads to the problem of time and space complexity. So exact matching is chosen since no conversion of data takes place and by using the ML classifiers, it is achieved by selecting and tuning the appropriate key parameters in each of the classifiers used.

To achieve the packet classification based on each flow, four steps are generally involved in the ML techniques, which include (i) define the features of the IPv4 packets, (ii) develop the machine learning model, (iii) train the model to manage the ML classifier in connecting a group of features with the known class labels, (iv) using the trained model to classify the unknown flows in the given classification system.

The objective of this paper includes (i) to evaluate the performance of the six ML classifiers applied to the IPv4 routing table (ii) to evaluate the effects of the training samples size and accuracy of the classification results (iii) to tune the key parameters in each classifier to improve the accuracy.

2 RELATED WORK

THE process of the classifying packets into flows requires multiple fields of the packet header. Each rule has fields, which are defined either as ranges or prefixes and the matching is done by applying the range matching, prefix matching and longest prefix matching. All these existing matching algorithms leads to the problem of rule overlapping, high computation cost, and poor performance in terms of the accuracy of the packets classified. All the above limitations of matching algorithms are solved by the machine learning based methods.

Liu, at al. (2012) clarified that the few ML algorithms used supervised the learning method and this required a prior knowledge of the output classes. Therefore, this type of learning algorithm makes the system to perform the classification with less computation time and high classification performance.

Due to this reason, the ML algorithms are chosen in this paper.

Currently, the ML based classification methods attracted few researchers like Mrudul, et al. (2017) and Wang, et al. (2016) for the traffic classification and are widely applicable in the intrusion detection system, the quality of service, and the security and monitoring system. Recent related works focused on the feed forward neural network (Qin, et al. (2012), the radial basis function (Namdev, et al. (2015), the decision tree K Singh, et al. (2011), the Support Vector Machine (Erman, et al. (2006) the Random forest (Singh A, et al. (2017), the Naive Bayes classifier (Ashari, et al. (2013), the K-nearest neighbors, and the Adaptive Nearest Neighbors (Li, 2011; Thanh, 2017; Lin, 2006) to classify the packets. All the researchers investigated and compared the performance of the algorithms. Some algorithms like the feed forward and the redial basis function network may require more time to learn the training samples. But few algorithms like the decision tree, the KNN and the Naive Bayes requires less learning time and low computation time.

Wong, et al. (2017) and Couronn, et al. (2018) concentrated on several studies by comparing the performance of the classifier with other classification algorithms by varying the training sample sizes. Similarly, the performance of the IPv4 packet classification is analyzed based on the classification accuracy and computation time. Powers, et al. (2011) and Davis, et al. (2006) explained that each algorithm has a key parameter, and these parameters are tuned to produce the best accuracy. The existing ML algorithms whose accuracy and execution time is said to be compared with the proposed machine learning algorithm used in this paper. To reduce the learning time and to predict the class more accurately the random forest algorithm is chosen. The key parameters of each algorithm are tuned to get achieve high accuracy of classification.

3 METHODOLOGY

THE overall methodology of the packet classification problem is shown in Figure 1, which consists of two important phases, the training and testing phase. A detailed description of each block is explained in the following subsections.

3.1 The Netbench Rule Set Description

To achieve the objectives of (i) and (ii), this work concentrated on choosing the standard netbench rule set (the ACL rule set) showing information about the IPv4 packet header. You, et al. (2017), Perez, et al. (2014) and Gupta, et al. (1999) are used the 5-fields of the packet header information and in this work, the same fields are selected, namely the Source IP ($f1$), the Destination IP address ($f2$), the Source port ($f3$), the Destination port ($f4$) and the Protocol type ($f5$) of each packet. The sample dataset is shown in Table 1. For our classification problem, the 5-fields are considered as input and the action as output, which are widely used in papers such as Dixit, et al. (2012) and Mrudul, et al. (2017). Table 1 shows the input features with the output flow/action for each rule. Each flow in the table is specified by a rule and each rule consists of five fields. All the five input features are normalized using the min-max normalization.

3.2 The Training and Testing Sample Dataset

For an accurate evaluation of the classification, the five features are chosen and are shown in Table 1. In order to evaluate the effect of the training sample sizes, three different data sizes are chosen and are shown in Table 2. Based on the sample sizes given in Table 2, the training and testing are done by applying the ML algorithms. The performance of each training sample is noted to choose the correct sample sizes for both training and testing.

Table 2. The Training and Testing Sample Sizes.

Rule set	Training size	Testing size
ACL1_1K	60%	40%
	70%	30%
	80%	20%

3.3 The Classification Algorithms

Each ML algorithm has one or more key parameters and the parameters are said to be tuned to produce high accuracy. Six types of the ML algorithms, namely the Multi-layer perceptron, the K-Nearest Neighbor, the Decision Tree, the Random Forest, the AdaBoost classifier and the Support Vector Machine are compared in this work. The training samples are first used to learn all the algorithms by assigning the sequence of the values for the parameters.

During the training stage, for each parameter whose optimal value accuracy high is selected, and those optimal values are used for testing. The testing samples are applied to the trained models to classify the given packets and compare the accuracy performance. Table 3 illustrates the most specific key parameters used in this work for each classifier.

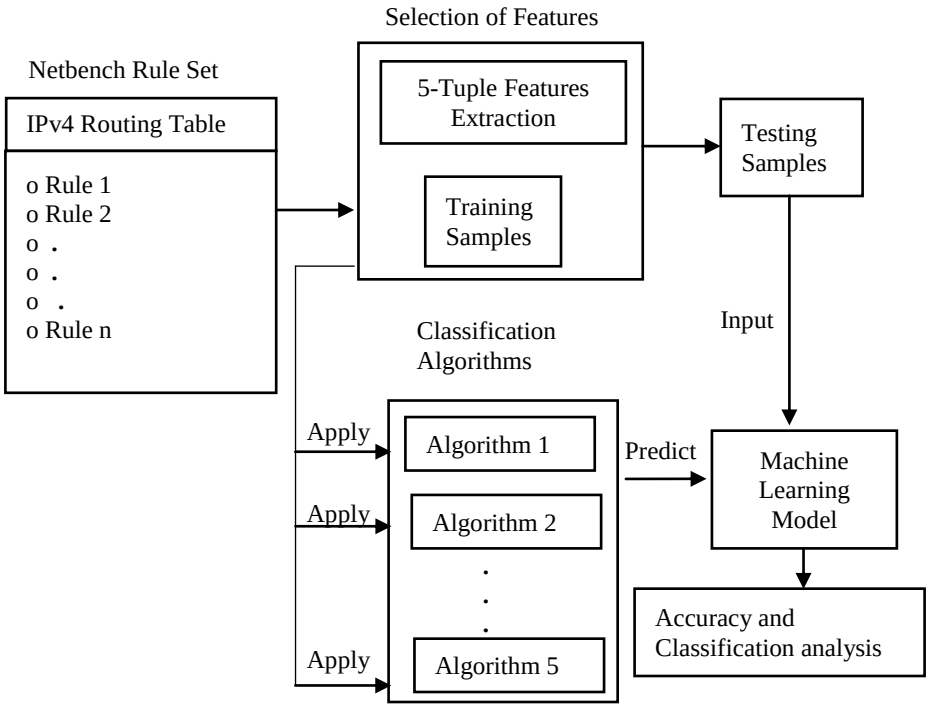

Figure 1. **The Training and Testing Phases of the Machine Learning Mode.**

Table 3. **Parameter tuning of machine learning classifier.**

Classification Algorithm	Key Parameters	Values
Multi-layer perceptron	Hidden nodes	10 to 50
	Epochs	1000 to 5000
K-Nearest Neighbor	K-value	1 to 25
Decision tree	Selection criteria	Gini
	min_sample_leaf	1
	min_sample_split	2
Random forest	n-estimators (ntree)	100 to 500
	n-jobs (step size)	1
	mtry	1 to 10
AdaBoost	Learning rate	1
	n-estimators (ntree)	100 to 500
Support Vector Machine	Kernel	Linear
	Cost (c)	100
	Cache size	200

3.4 The Accuracy and Comparison Analysis

All the objectives are achieved by applying the six different types of classification algorithms and the accuracy of each algorithm is compared with the three different training sample sizes 60%, 70% and 80%. In order to measure the classification performance, a few metrics are considered which are presented in articles by Liu, et al. (2012) and Singh K, et al. (2011). Accuracy of the classification, precision, recall, F1score, false positive rate, and true positive rate for each classification algorithms are measured, and the results are compared with each other to select the best sample size. The formula for calculating above metrics are given in the following equations.

The accuracy is defined as the ratio of the number of correct predictions made by the overall predictions made.

$$\text{Accuracy} = \frac{\text{Correct Prediction}}{\text{Total Prediction}} * 100 \quad (1)$$

The precision is defined as the ratio of the true positive to the true and false positive. This decides how many objects are correctly identified. The true positive represents the percentage of the correct classified packet as targeted packets and the false positive denotes the percentage of other types of packets classified as targeted packets.

$$\text{Precision} = \frac{TruePositive}{TruePositive + FalsePositive} \quad (2)$$

The recall is defined as the ratio of the true positive to the number of true positives and false negatives. This shows how many packets in a class are misclassified. The false negative represents the percentage of the targeted packets incorrectly classified as other packets.

$$\text{Recall} = \frac{\text{TruePositive}}{\text{TruePositive} + \text{FalseNegative}} \quad (3)$$

The F1Score is a measure of the test's accuracy. It is concentrated on both precision and recall to compute.

$$\text{F1score} = \frac{2*(\text{Recall}*\text{Precision})}{\text{Recall}+\text{precision}} \quad (4)$$

The training time is measured as the total time taken for training of a machine learning classifier. In this paper, it is measured in seconds.

4 RESULT ANALYSIS

THE ML classifier algorithms are implemented in the Python3 using the Scikit-learn packages and are executed by using an Intel i5-2400 processor with a 3.10GHz clock frequency. The Scikit-learn offers libraries that implement many machine learning algorithms. In this work, the Netbench dataset is taken for the IPv4 packet classification, which consists of five input features and two output class labels. The machine learning algorithms are applied to classify the packets and depends on the action or flow with the different training samples. The classification accuracy, recall, precision, and f1score are evaluated by varying the ranges of the key parameters.

4.1 The Multi-layer Perceptron

A single layer perceptron, which consists of the input layer having neurons equals to the number of features in an input sample. The user defined neurons in the hidden and output layers having neurons equal to the output of the network. In this network, the error between the actual and expected output is propagated in a backward direction. This single layer perceptron network gives a 72% accuracy by keeping the learning rate of 0.01. The learning time required for this network is 1262 seconds, which is high when compared to other classifiers.

4.2 The K-Nearest Neighbor

The K-nearest neighbor classifier is a supervised algorithm, which captures the relationship between the given training samples of (x, y), where x is the input feature and y is the output class. Singh A, et al. (2017) and Thanh, et al. (2017) describe that, the packets are classified based on a majority of classes among its k-neighbors, so the 'k' (tuning parameter) value plays a vital role in the classification performance of the KNN and hence the 'k' value is chosen as a tuning parameter. A range of the 'k' value is tested from 1 to 25 for finding the optimal value for 'k'.

Figure 2 shows the comparison between the 'K' value and the classification accuracy. When the k-value increases, the accuracy of the corrected classified packets decreases. The training sample is 60% of data, the accuracy attained is 93.31% to the k value at 1. Similarly, the training sample of 70% and 80%, the highest accuracy 92.7% and 95.15% are obtained with the 'k' value =1. From the analysis of Figure 2 it clearly shows that the satisfactory result was attained for k=1, so the optimal value for the 'k' is chosen as 1 (k=1) for our packet classification problem.

The KNN algorithm for finding the 'k' value is illustrated in Algorithm 1. Though the KNN classifier produces acceptable accuracy, the classifier leads to high computational cost. We need to compute the distance of each query instances to all the training samples.

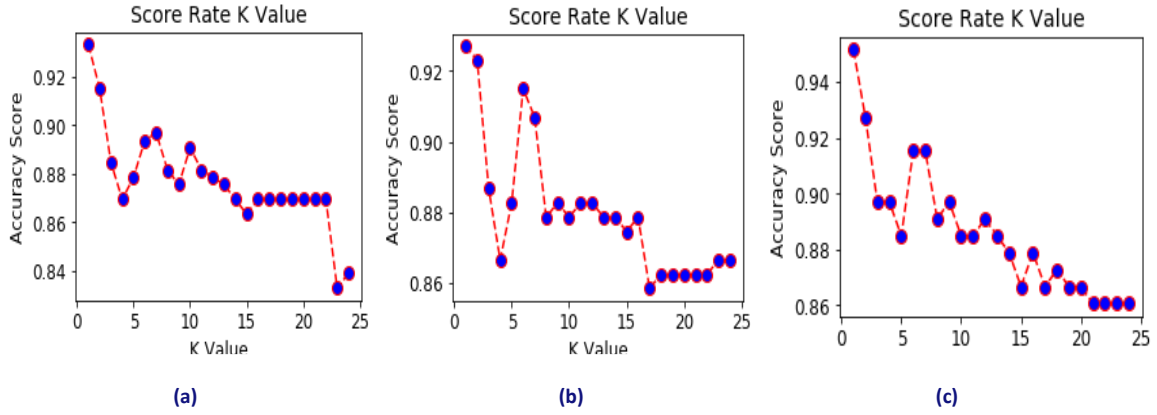

Figure 2. The Relationship between the Classification Accuracy and the k Value (Where k = 1 to 25) for the Varying Training Sample Size of the KNN Classifier (a) The Training Sample Size is 60%, (b) The Training Sample Size is 70%, (c) The Training Sample Size is 80%).

Algorithm 1 Finding the K value in the K-Nearest Neighbor

Classify (X,Y,s)
for i = 1 to k do
  Compute Distance d(X_i,s)
end for
Compute set T containing indices for the k-smallest distances d(X_i,s)
return majority label for Y_i where iϵT

4.3 The Support Vector Machine

Many of the researchers focused on using the radial basis function kernel of the SVM classifier. But, for the IPv4 packet classification problem, the linear kernel is used to implement this SVM algorithm. The key tuning parameter is gamma, which is the kernel width parameter (γ). This parameter distresses the smoothing of the shape of the class partitioning of the hyper plane. The learning of this hyper plane in the linear SVM is done by using linear algebra. For the linear kernel, as the gamma value increases, it may affect the accuracy of the classification results. In order to find the optimal parameters for the SVM, six values of γ (2^{-1},2^0,2^1,2^2,2^3,2^4) are tested and these values are applied to all three different training samples.

Figure 3 shows the relationship between the gamma and mean square error when the value of the gamma is 8, and the error is high and started to remain stable. The optimal value for the gamma chosen for our classification problem is 2. Since starting from gamma=2, the error started increasing. The major drawbacks of the SVM are to select the appropriate kernel function. Computation overheads may occur if we fail to predict the optimal value for the gamma, Tingyao et al. (2015). In this work, we mainly focused on the tree-based classifier like the decision tree, the random forest and the adaboost classifier.

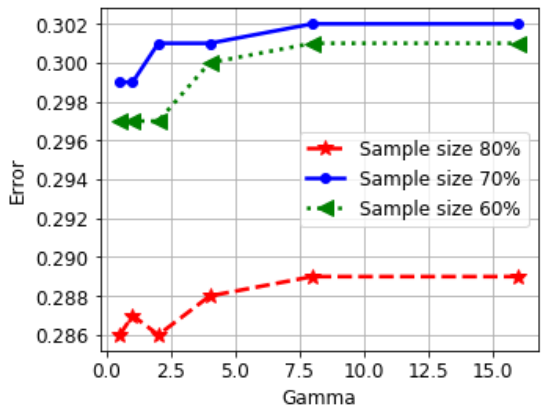

Figure 3. The Relationship between the Gamma and Mean Square Error of the SVM Classifier.

4.4 The Decision Tree

The decision tree classifier is a tree where each internal node denotes the test of an attribute. Each branch node denotes the outcome of the test and the leaf node denotes the class namely permit or deny. Here the CART algorithm is used to build a tree and it uses the Gini coefficient as the selection criteria where the sample leaf is set as 1 and the leaf split is set as 2. Ashari, et al. (2013) stated that each time the attribute chosen with the smallest Gini coefficient is the test attribute for a given sample. This algorithm is very simple and achieved was the accuracy of 85.1%, 85.8% and 89.7% and for the training sample sizes of 60%, 70% and 80% respectively. However, the decision tree classifier has major disadvantages. Since this classifier uses a divide and conquer method, they perform well only if a highly relevant attribute exists. If many complex interactions are present, the accuracy is low. The decision tree classifier works better only if the target attributes have discrete values. The defects of this classifier are overcome by using the random forest algorithm.

4.5 The Random Forest Classifier

The Random Forest is a combination of several decision trees and works as an ensemble model. When compared to the decision tree, this classifier helps in reducing the variance by averaging the prediction of trees used and hence reduces the over fitting problem. In this classifier, the two key tuning parameters namely the number of trees (ntree) and the number of features in each split (mtry) are tuned to improve the accuracy. Several studies focused on using these two parameters for better performance. The parameter namely, the number of trees is used to build many trees to predict the class by taking the average over all the trees.

Figure 4 shows the structure of the simplified random forest model for the IPv4 packet classification. All the 5-tuple information are stored in a separate decision tree and searching takes place in each tree. Based on most of the voting, the rule is chosen and the corresponding action for the given is performed. Hence, in this way the packet classification is achieved, and the searching procedure takes place in parallel to reduce the computation time.

To perform searching in the RF, two parameters the ntree and the mtry play an important role. In order to find the optimal value for these parameters, a range of value is assigned and are tested. In the random forest, each tree is grown using the randomized tree building criteria. The prediction of the ‘ntree’ is averaged to give the final prediction. The tree construction continues until each leaf node contains not more than the ‘k’ training samples for some specified ‘mtree’.

For the ‘ntree’, we have chosen 100 to 500 and the ‘mtry’ is 1 to 10 with the step size set as 1. However, in this analysis, the ‘ntree’ from 300 to 500 retains the

same accuracy, so the 'ntree' range up to 300 is chosen. All these ranges are tested, and the highest results are attained with the 'mtry', which equals to 2. Figure 5 shows the relationship between the number of split features and the accuracy of the classification with the effect of the number of trees used. In 60% of the training samples, 90% accuracy is achieved with the mtry = 2 and the ntree = 100. Similarly, for the 70% and 80% of the training samples, the accuracy obtained is 93% and 96.9%, respectively. The Random Forest classifier is one of the most accurate learning algorithms, which produces a highly accurate classifier.

Similarly, the comparison between the ntree and the Out-Of-Bag (OOB) error is shown in Figure 6. Figure 6 illustrates that the error is decreased when the number of trees increases. For ntree to = 100 to 400, the OBBs are slightly changed for the three different training samples. For all the given training samples, the OOBs error remains constant when the ntree is from 400 to 500, so the tree construction stops when the ntree = 500. Therefore, for our packet classification problem, the ntree = 100 is chosen as an optimal value for the RF classifier.

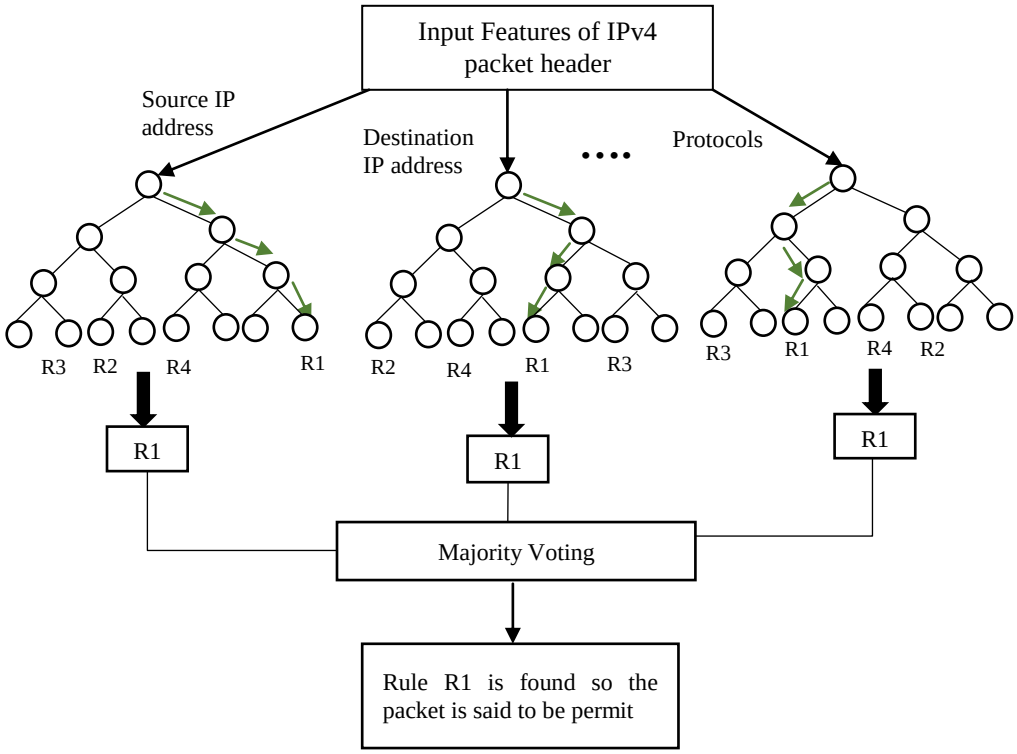

Figure 4. **The Simplified Structure of the Random Forest Model for the Packet Classification.**

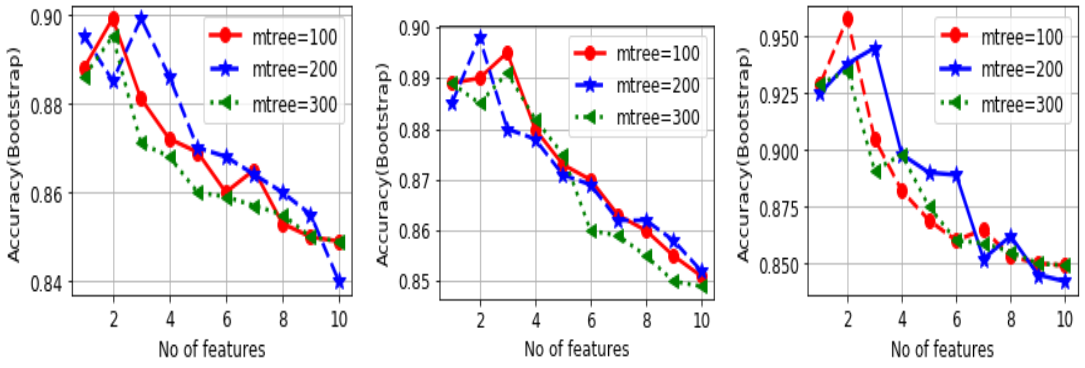

Figure 5. **The Relationship between the Number of Features and Accuracy of the RF Classifier with the Effect of the Number of Trees. (a) The Training Sample Size is 60%, (b) The Training Sample Size is 70%, (c) The Training Sample Size is 80%).**

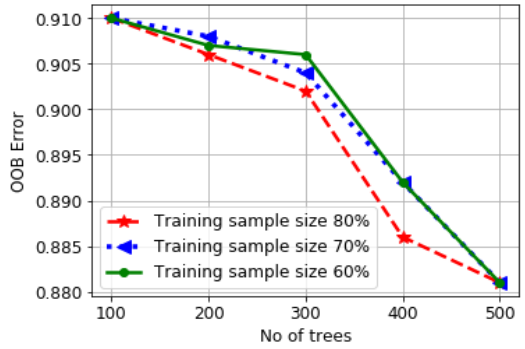

Figure 6. The Comparison between the ntree and the OOB Error using the Different Training Samples.

In this work, the hyper parameters namely the 'n-estimators' and the min_sample_leaf increases the performance accuracy and the model speed is increased by the 'n_job' and the 'oob_score'. The classifier is easy to measure on the relative importance of each feature on the prediction. The Sklearn provides a tool, which measures the above features, so this algorithm is considered easy to use and the optimal value for the hyper parameters in this classification problem produces a good prediction result.

4.6 The AdaBoost Classifier

The AdaBoost classifier is used to improve or boost the performance of the decision tree using the binary class (Permit/Deny) classification. This classifier is used only for binary classification problems, so each decision base makes one decision on the given input variables and the outputs as 0 or 1. In this work, the decision base is prepared with the training samples using the weighted samples. Each feature in the training sample is said to be weighted. The weight is calculated as

$$weight(x_i) = \frac{1}{n} \quad (5)$$

where x_i is the i^{th} input feature and n is the number of training features, (each row has five input features with 1 output). The packet classification accuracy is varied by the tuning the key parameter ntree (number of trees). The range of the ntree [100 to 600] is chosen and the accuracy for the three different training samples are analysed.

Figure 7 shows that when the ntree =100, the accuracy is high showing 90.6%, 93.5%, and 93.7% of all the three training samples, respectively, so the optimal value for the ntree =100 is chosen for our classification problem. When the number of the tree increases, there is a slight variation in the classification rate. Figure 8 shows that the error rate is high when the ntree = 200. From ntree 300 to 500 there is only a slight variation in error for a training sample of the size 70% and 80%, so the optimal value is said to be 100 for this classification problem.

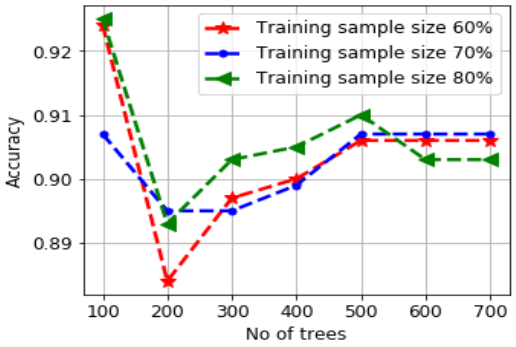

Figure 7. The Relationship between the Classification Accuracy and the Number of Trees.

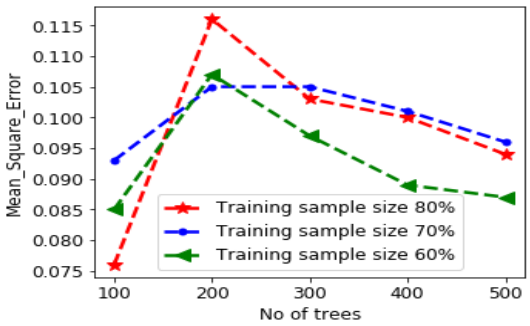

Figure 8. The Relationship between the Mean Square Error and the Number of Trees.

4.7 The Performance of Classifiers

From the analysis of each classifier algorithm, the optimal value of the various tuning parameters is summarized and illustrated in Table 4.

Table 4. The Key Parameter with their Optimal Value

Classification Algorithm	Key Parameters	Values
Multi-layer perceptron	Hidden nodes Epochs	21 3000
K-Nearest Neighbor	K-value	1
Decision tree	min_sample_leaf min_sample_split	1 2
Random forest	n-estimators (ntree) n-jobs (step size) mtry	100 1 2
AdaBoost	Learning rate n-estimators (ntree)	1 100 to 500
Support Vector Machine	Cost (c) Cache size	100 200

By using the parameters, the overall performance of these algorithms is described in Table 5 and it shows that the training sample size of 80% of data is achieved at a higher classification accuracy for all the

Table 5. The Overall Comparison Performance of the Machine Learning Classification Algorithms.

Machine Learning Algorithms	Train data size (%)	Accuracy	Precision	Recall	F1score	False Positive Rate	Time (sec)
MLP	60	64.3	0.854	0.304	0.448	0.062	1012
	70	69.1	0.786	0.573	0.663	0.217	1218
	80	72	0.718	0.692	0.704	0.301	1299
KNN	60	93.31	0.909	0.966	0.937	0.111	0.0174
	70	92.7	0.909	0.966	0.937	0.111	0.0248
	80	95.15	0.943	0.976	0.96	0.062	0.1222
SVM	60	70.3	0.714	0.706	0.71	0.3	0.0594
	70	69.6	0.738	0.689	0.713	0.295	0.0793
	80	71.4	0.733	0.733	0.733	0.307	0.0914
Decision Tree	60	85.1	0.862	0.922	0.89	0.17	0.0258
	70	85.8	0.853	0.948	0.898	0.196	0.0508
	80	89.7	0.844	0.953	0.895	0.188	0.0261
AdaBoost	60	90.6	0.896	0.932	0.914	0.124	0.19284
	70	93.5	0.90	0.933	0.916	0.125	0.18444
	80	93.7	0.92	0.953	0.936	0.087	0.18514
Random forest (proposed)	60	90	0.915	0.977	0.945	0.105	0.1258
	70	93	0.899	0.985	0.94	0.988	0.1266
	80	**96.9**	**0.934**	**1**	**0.966**	**0.075**	**0.1274**

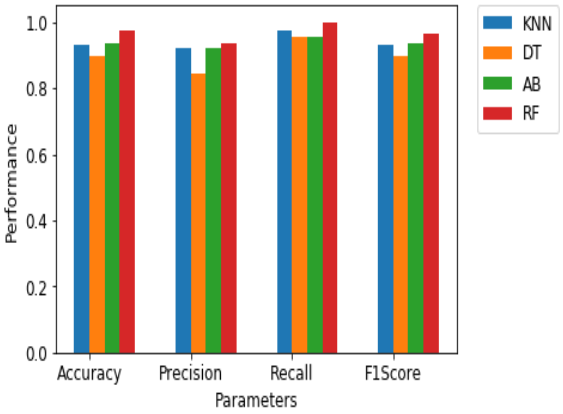

Figure 9. The Overall Comparison of the Accuracy, Precision, Recall and Fscore for the Classifier Algorithms using the Training Sample Size by 80%.

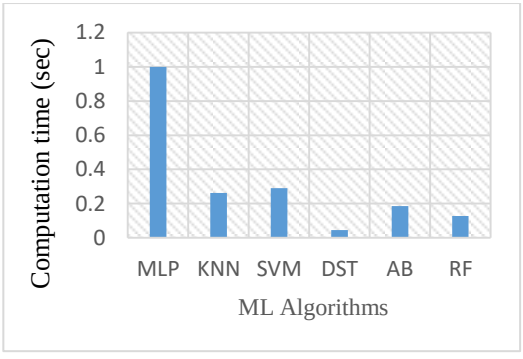

Figure 10. The Comparison of the Computation Time.

classifier algorithms so in this paper, the result analysis and discussion is concentrated on the 80% of the training samples. The results clearly show that the MLP gives a very poor performance in terms of accuracy and execution time. Furthermore, the classification accuracy of the SVM is also below 80%, which is not appropriate for the packet classification.

Therefore, the MLP and SVM algorithms are not taken into consideration for further discussions. When comparing the conventional approaches, Singh K et.al (2011) also suggested that the execution time for the MLP and the radial basis function are high and accuracy low. When compared to the existing conventional approaches, the Random Forest algorithm gives better performance, whose accuracy is 96.4% and the execution time is 0.1274 secs, but other ML algorithms are merely less than the random forest. So, the random forest is used as a good packet classification algorithm due to high accuracy.

The experimental results demonstrated that the accuracy of the classification rises with increasing training samples. For the IPv4 packet classification problem, the RF classifier performed better when compared with other classifier algorithms. The existing machine learning classifier (Sing, et al. 2011) like the multi-layer perceptron, and the radial basis function required an execution time of 205 seconds and 41 seconds, respectively. In this paper, the random forest required 0.1274 seconds for the performing classification with high accuracy than the existing machine learning classifier. This classifier always added additional randomness to the model at the time of growing the trees. While splitting the node, it searches only the best features among a random subset of features, instead of searching for the most important features. Therefore, this random forest classifier is better for the accurate classification than other classifiers.

5 CONCLUSION

A network classifier packet depends on the class and adopts the mechanism to ensure the QoS requirements. To do so, every router needs to classify the packets. To achieve the packet classification, the ML classification algorithms are studied and evaluated. Three objectives in this paper are achieved by using the Random Forest classifier algorithm. Experimental result show that the three algorithms give high accuracy ranging from 93% to 97%, with the RF classifier on the average producing the highest accuracy with a training sample size of 80%. By choosing the optimal value for the ntree and the mtry key parameters, the RF classifier attained the best accuracy rate.

Over-fitting is a major problem in the machine learning algorithms but most of the time, over-fitting won't happen in the random forest algorithm. Due to enough trees in the forest, the classifier won't over-fit the model but sometimes due to many trees, this algorithm may become slow and ineffective for real time predictions. This random forest is fast to train, but it is slow to create predictions. We further concentrate on the deep learning neural network with the TensorFlow for the classifying packets using different features of the IPv4 packet header.

6 DISCLOSURE STATEMENT

NO potential conflict of interest was reported by the authors

7 REFERENCES

Ashari, A., Paryudi, I., and Tjoa, A.M., (2013), Performance Comparison between Nave Bayes, Decision Tree and k-Nearest Neighbor in Searching Alternative Design in an Energy Simulation Tool, *International Journal of Advanced Computer Science and Applications*, 4 (11).

Avudaiammal, R., Swarnalatha, A., and Seethalakshmi, P., (2017). Network Processor Based High Speed Packet Classification for Multimedia Applications, Wireless Personnel Communication, 98, 1219-1236.

Couronn, R., Probst, P., and Boulesteix, A.L. (2018), Random forest versus logistic regression: a large-scale benchmark experiment, *Journal of Bioinformatics*, 19 (270).

Davis, J. and Goadrich, M. (2006), The Relationship Between Precision-Recall and ROC Curves, *CML '06 Proceedings of the 23rd international conference on Machine learning*, 233- 240.

Dixit, M., Kale, A., Narote, M., and Talwalkar, S. (2012), Fast Packet Classification Algorithms, *International Journal of Computer Theory and Engineering*, 4, 1030-1034.

Elmahgiubi, M., Ahmed, O., and Areibi, S. (2016), Efficient Algorithm Selection for Packet Classification using Machine learning.

Erman, J., Mahanti, A., and Arlitt, M. (2006), Internet traffic identification using machine learning, *IEEE International Conference on Global Telecommunications*.

Gupta, P. and McKeown, N. (2001), Algorithm for packet classification, *IEEE Network*, 15, 24-32.

Gupta, P. and McKeown, N. (1999), Packet classification on multiple fields, *ACM SIGCOMM Computer Communication Review*, 29,147-160.

Li, S., Harner, E.J., Adjeroh, D.A., Singh, A., (2011), "Random KNN feature selection - a fast and stable alternative to Random Forests", *Bioinformatics*, 12 (450), 1-11.

Lin, Y., and Jeon, Y. (2006), Random Forests and Adaptive Nearest Neighbors, *Journal of the American Statistical Association*, 101, 578-590.

Liu, Y. (2012), A Survey of Machine Learning Based Packet Classification, *Symposium on computational intelligence for security and Defence Applications*, 1-10.

Mrudul Dixit1, Ankita Sanjay Moholkar, Sagarika Satish Limaye, and Devashree Chandrashekhar Limaye (2017), Statistical based Approach for Packet Classification, *International Journal of computer and Mathematical sciences*, 6, 25-31.

Namdev, N., Agrawal, S., and Silkari, S. (2015), Recent advancement in machine learning based internet traffic classification, *Procedia Computer Science*, 60, 784-791.

Paul, A., Mukherjee, D. P., Gangopadhyay, A., Chintha, A.R., and Kundu, S., Improved Random Forest for Classification, *IEEE Transactions on Image Processing*, 27 (8), 4012-4024.

Perez, K.G., Yang, X., and Scott-Hayward, S. (2014), Optimized Packet Classification for Software-Defined Networking, *IEEE International Conference on Communications*.

Powers, D.M.W. (2011), Evaluation: From precision, recall and F-measure to ROC, Informedness, Markedness and Correlation, *Journal of Machine learning technologies*, 2, 7-63.

Qin D, J Yang, J Wang, and B Zhang (2012), IP Traffic classification based on machine learning, *IEEE International Conference on Communication Technology*.

Singh, A. and Halgamuge, M.N., (2017), Impact of Different Data Types on Classifier Performance of Random Forest, Nave Bayes, and K-Nearest Neighbors Algorithms, *International Journal of Advanced Computer Science and Applications*, 8 (12).

Singh, K., and Agrawa, S. (2011), Comparative Analysis of Five Machine Learning Algorithms for IP Traffic Classification, *IEEE International Conference on Emerging Trends in Networks and Computer Communications (ETNCC)*,33-38

Thanh Noi, P., and Kappas, M. (2017), Comparison of Random Forest, k-Nearest Neighbor, and Support Vector Machine Classifiers for Land Cover Classification Using Sentinel-2 Imagery, *Sensors*, 18 (1).

Tingyao Jiang, Peng Lei, and Qin Qin (2015), An Application of SVM-Based Classification in Landslide Stability, *Intelligent Automation and Soft Computing*, DOI: 10.1080/10798587.2015.1095480.

Wang, P., Lin, S.C., and Luo, M. (2016), A Framework for QoS-ware Traffic Classification Using Semi-supervised Machine Learning in SDNs, *IEEE International conference on Services Computing (SCC)*.

Wong, T.T. (2017), Parametric Methods for Comparing the Performance of Two Classification Algorithms Evaluated by k-fold Cross Validation on Multiple Data Sets, *Pattern Recognition*, 65, 97-107.

You Lu, Baochuan Fu, Xuefeng Xi, Zhancheng Zhang, and Hongjie Wu (2017), An SDN-Based Flow Control Mechanism for Guaranteeing QoS and Maximizing Throughput, Wireless Personal Communication, 97,417-442.

8 NOTES ON CONTRIBUTORS

B. Indira completed M.Tech at Kalasalingam Academy of Research and Education, India from 2009 to 2011.She worked as Assistant Professor in the Department of Computer Science and Engineering for 2 years. Currently, she is pursuing full time Ph.D. under the guidance of Dr. K. Valarmathi in the Department of Information and Communication Engineering in Anna University Chennai. Research areas include networks, data structure and machine learning.

K. Valarmathi obtained Ph.D. degree from the Anna University, Chennai in 2008. She is a member of the IEEE. She is a reviewer for various reputed journals. Her research interest includes process control, system identification, biomedical instru-menttation, image processing, machine learning, wireless network, cloud computing. Currently, she is working as a professor and head of the department of Electronics and Communication Engineering, at P.S.R. Engineering College. She published 23 papers in SCI journals and 58 papers in conferences.
Email: valarmathi@psr.edu.in