AN IMPROVED QOS-AWARE CLUSTERING APPROACH FOR NETWORK TRAFFIC CLASSIFICATION

A

Thesis

Submitted to

For the award of

DOCTOR OF PHILOSOPHY (Ph.D.)

in

Computer Applications

By

Kate Takyi

(11617542)

Supervised By

Dr. Amandeep

LOVELY FACULTY OF TECHNOLOGY AND SCIENCES
LOVELY PROFESSIONAL UNIVERSITY
PUNJAB
September, 2019

Certificate

I certify that Ms. Takyi has prepared her thesis entitled **An Improved QoS-Aware Clustering Approach for Network Traffic Classification** for the award of Ph.D. Degree of Lovely Professional University under my guidance. She has carried out the work at the Department of Computer Applications, Lovely Professional University.

Advisor

Dr. Amandeep
Associate Professor
School of Computer Applications
Lovely Professional University,

Punjab.
Date:

Declaration

I declare that this thesis entitled **An Improved QoS-Aware Clustering Approach for Network Traffic Classification** has been prepared by me under the guidance of Dr. Amandeep, Associate Professor at the School of Computer Applications in Lovely Professional University. No part of this thesis has formed the basis for the award of any degree or fellowship previously.

Kate Takyi

School of Computer Applications,

Lovely Professional University,

Punjab.

Date:

Abstract

The task of network administrators to identify and determine the type of traffic traversing through the network is very critical to the rapid growth of new traffic each day. To ensure that the type of traffic coming in and out of a network is safe, traffic must be identified which requires classification of traffic in real-time. As the requirements of networks change over time, the situation of the network not able to meet some requirements is likely to occur. The same resources that could meet the requirements today may not be enough for tomorrow or in the future. This can negatively affect the quality of service the network offers. Considering wide area networks, with limited resources in terms of low-speed links, quantified number of packets are likely to be lost with varying fragments of packets transmitted at a time which lowers the quality of service. The classification procedure in such scenarios can also be affected due to the limited features extracted from the various fragments of packets that will successfully get to the destination node or server. After a rigorous and extensive review of literature, we realized that not much investigation has been conducted in this area since almost all the proposed work has not considered the case of a limited resource in networks in their proposed works. The renowned works done assumes the network under consideration has all the requirements fulfilled by the network's resources at any point in time. However, such consistent circumstances rarely happen physically. Hence, the need for a solution to help classify traffic flows accurately in such situations is highly required.

The study aspires is to discover the effect of packet loss and fragmentation during the classification procedure. From previous works, the assumption of whether or not the network resources meet its requirements is not considered. This research factors such assumptions. The aim is to discover the magnitude of the effect it has on the statistical features of the flow extracted for classification. Two wide area networks (Wired and Wireless) exposed to extreme packet loss scenario is designed and implemented using OMNET ++ simulation to generate our proposed Fragmentation Packet Loss Induced (FPL) dataset 1 and dataset 2 from the proposed wired and wireless topologies

respectively. The wireless network is proposed to further validate the efficiency of the proposed algorithms in such environments. The speed of links in the networks is such that the number of packets sent at a time exceeds the capacity the links can transport at a time. This ensures the occurrence of fragmentation leading to extreme packet loss.

The second objective of the research is to propose a QoS aware semi-supervised clustering algorithm that will improve upon the performance of traffic classification in networks. The aim of this objective is to provide a solution in situations discovered in the first objective. The semi-supervised algorithm is implemented into a classifier that is able to classify application traffic or packets, utilizing restricted traffic features, few packets and at the same time maintains a low complexity and good classification accuracy. This is achieved by proposing two clusters and label hybrid algorithms, namely KNN+K-Medoids (KNKM) and SVM+K-Medoids (SVKM). The design of both algorithms makes full use of the advantages of the algorithms they constitute and concurrently optimizing other parameters of the primary algorithms. The proposed hybrid algorithms achieved good classification results with respect to the proposed scenario. However, accuracy levels needed to be improved. Also, the time complexity of SVKM resulted to be higher regardless of the improved accuracy rates. R-TAC semi-supervised algorithm is proposed to overcome these limitations.

For the third objective of the study, we implement and validate the two algorithms to serve as a classifier to generate a compact summary (classify) of various traffic types while increasing the performance in presence of Quality of Service Parameters. The proposed model is built and tested in a MATLAB simulation environment. The dataset generated from the OMNET simulation is initially filtered based on the statistical features discovered from the data logs at the end of the simulation. The algorithms are designed and implemented in MATLAB simulation environment with FPL datasets. The algorithms are evaluated using the following classification metrics:

a. Precision
b. Accuracy
c. The area under Receiving Operating Characteristics
d. Processing Time
e. Error Rates

To validate the proposed work, we compare the results achieved with other existing works in literature in terms of the evaluation metrics. Results from the comparison show that our proposed semi-supervised algorithms perform better in such scenarios where networks have limited resources in terms of the parameters selected for the study. Also, the proposed works are able to distinctly classify and separate classes with no conflicting or overlapping clustering or classification. The overall processing time is also fair considering the amount of flows and features incorporated in the classification procedure. Furthermore, two datasets namely Cup KDD 1999 and IDS Trace from Cyber Watch Mid-Atlantic Collegiate Cyber Defense Competition (MACDCC) are employed to validate the accuracy of the proposed models. A compact summary of the traffic traces is derived along with their respective percentages for each traffic class. The results are compared with the existing works in literature with the selected datasets. The results achieved further prove that the proposed works are efficient traffic classifiers, with the overall best classifier being the R-TAC model.

Acknowledgements

My greatest appreciation and thanks are to the Almighty God, through His son my lord and savior Jesus Christ, in Him and through Him all things are made possible – Matthew 12:26, Amen. It is by His gracious mercies, strength, and power that enabled me to complete this research and thesis.

Secondly, my sincere and lovely gratitude would be expressed to my supervisor Dr. Amandeep Bagga for her patience, support, guidance and encouragement from the start to the completion of my Ph.D. studies and thesis writing. May the good Lord bless you abundantly and increase you from grace to grace.

Also, I would like to express a special and warmest appreciation to Dr. Pooja Gupta for her precious time spent, constructive criticisms, encouragement, support, and prayers. She is really God sent and a mentor anyone could wish for.

To all the panel members of SOTA, End Term Seminars and Research Degree cell, I say thank you for providing critical and constructive suggestions during the progress reports and presentations. Without you, my research would not have been kept on track.

I would also like to thank my father, Mr. Mohammed Garibah, a strong pillar in my life for his financial support throughout my studies, my mother Mrs. Susuana Birago and all my siblings for their prayers, support and sacrifices every day. I could not ask for any better family. God bless you and replenish everything you have spent on me.

Finally, to all my friends for the emotional support and being my family far away from home, I say thank you and may the good God grant you success in all your endeavors.

Contents

List of Figures 7

List of Tables 10

Chapter 1: Introduction 11

1.1 Traffic Classification 11

1.1.1 The Port-based Classification 13

1.1.2 The Payload Classification 14

1.1.3 Behavioral Classification 15

1.1.4 Statistical Classification 15

1.2 Machine Learning Techniques 16

1.2.1 Clustering Approach of Classification 18

1.2.2 Standards for Partitioning 20

1.2.3 Distance between Clusters 21

1.2.4 Quality of Clusters and Scalability 21

1.3 Clustering Algorithms 21

1.3.1 Application Areas of Clustering 22

1.4 Motivation of the Investigation 22

1.5 Objectives of the Study 23

1.5 Contributions and Scope of the Study 23

1.7 Research Methodology 24

1.7.1 Description of Suggested Objectives 25

1.7.4 Hardware and Software Requirements 25

1.8 Thesis Structure 26

Chapter 2: Background Study 29

2. 1 Packet Loss Parameter 29

2.1.1 Packet Loss versus Latency 29

2.1.2 Packet Loss versus Throughput 30

2.1.3 Packet Loss versus Fragmentation 30

2.1.4 Causes of Packet Loss in Networks 31

2.2 The Network Study 33

2.2.1 Local Area Networks (LANs) 33

2.2.2 Wide Area Networks (WAN) 34

2.2.3 Metropolitan Area Networks (MAN) 34

2.2.4 Personal Area Network (PAN) 35

2.2.5 Campus Area Networks (CAN) 35

2.2.6 Storage Area Networks (SAN) 35

2.2.7 Enterprise Private Networks (EPN) 35

2.2.8 Virtual Private Networks (VPN) 35

2.3 Network Traffic Classification Techniques 36

2.4 The Unsupervised Strategy 37

2.5 The Semi-supervised Strategy 45

2.7 Classification for Specific Targets 48

2.7.1 Video Traffic 48

2.7.2 Peer-to-Peer Traffic (P2P) 50

2.7.3 Game Traffic 52

2.7.4 Voice over IP Traffic (VoIP) 53

2.7.5 Classification for Analytic Purpose 54

2.7.6 Classification to Enhance Quality of Service 55

2.8 Comparison of Traffic Classification Methods.. 56

2.9 Evaluation and Discussion of Results.. 59

2.10 Challenges in Traffic Classification... 60

2.11 Summary 60

Chapter 3: Design and Implementation of Proposed Scenario Topology......................... 62

3.1 Motivation of the Design.. 62

3.2 OMNET ++ Simulation... 62

3.3 Proposed Scenario Implementation.. 63

3.4 Classification Queues 66

3.4 Results of Topology Implementation... 67

3.5 Proposed Wireless Scenario 70

3.6 FPL Dataset Capture and Statistics 71

3.7 Effects of Packet Loss and Fragmentation... 73

3.8 Summary 76

Chapter 4: Design and Implementation of Proposed Algorithms..................................... 77

4.1 Motivation for Algorithms Design.. 77

4.1.1 Minimal Distance Error 77

4.1.2 Ease of Use and Complexity of Algorithm.. 78

4.2 KNKM Algorithm... 80

4.2.1 The K-Medoids Clustering Strategy 81

4.2.2 K-Nearest Neighbor Algorithm (KNN) 81

4.2.3 The proposed algorithm 81

4.2.4 Complexity of KNKM 84

4.2.5 Results of KNKM on FPL 1 Dataset 84

4.2.6 Results of KNKM on FPL 2 Dataset 89

4.3 SVKM Algorithm.. 90

4.3. 1 Motivation of Algorithm Design 90

4.3.2 Proposed Algorithm Implementation.. 90

4.3.3 Support Vector Machine (SVM)... 90

4.3.4 The proposed Algorithm 91

4.3.5 Complexity of SVKM.. 94

4.3.6 Results of SVKM (FPL 1) 94

4.3.7 Results on SVKM (FPL 2).. 96

.... 97

4.4 Real-Time Application-Based Clustering (R-TAC) 97

4.4.1 Motivation of Algorithm.. 97

4.4.2 The Proposed Algorithm.. 98

4.4.3 Complexity of R-TAC 99

4.4.4 Results of R-TAC 101

4.4.5 Results of R-TAC on FPL 2... 103

4.5 Summary.. 104

Chapter 5: Analysis of Results... 106

5.2 Precision.. 106

5.3 Accuracy.. 106

5.4 Error Rates.. 108

5.5 Area under ROC.. 110

5.6 Processing Time 111

5.7 Analysis of Results from Wired Scenario (FPL 1) in comparison to Wireless Scenario (FPL 2) 112

5.8 Summary 115

Chapter 6: Validation of Proposed Algorithms 116

6.1 Datasets for Traffic Classification 116

6.1.1 Historical and Near-Real-Time UCSD Network Telescope Traffic Dataset 117

6.1.2 The CAIDA Anonymized Internet Traces Dataset 2008 117

6.1.3 Statistical information for the CAIDA Anonymized Internet Traces 117

6.1.4 UCSD Network Telescope Educational Dataset 118

6.1.5 OC48 Peering Point Traces 118

6.1.6 ITA Network Traces 118

6.1.7 Waikato Internet Traffic Storage (WITS) 118

6.1.8 Cup KDD 1999 119

6.1.9 IDS Traces (Mid-Atlantic CDCC) 119

6.2 Validation of Algorithms with CUP KDD’99 Dataset 119

6.2.1 Results of Proposed Algorithms 119

6.2.2 Results on Existing Algorithms 122

6.3 Analysis of Results on CUP KDD’99 Dataset 122

6.4 Validation of Algorithms with MACDCC IDS Trace 126

6.4.1 Results on Proposed Algorithms 126

6.4.2 Results on Existing Algorithms 128

6.5 Analysis of Results on MACDCC IDS Trace 129

6.6 Conclusions of Validated Results 132

Chapter 7: Conclusion and Future Works 134

7. 1 Conclusion 134

7.1.2 Summary of Findings 136

7.2 Future Works 137

Bibliography 138

List of Publications 153

List of Figures

Figure 1: Cluster Representation 19
Figure 2: Flow of Proposed Methodology 27
Figure 3: Structure of the Thesis.. 28
Figure 4: Illustration of K-means Clustering 38
Figure 5: Proposed Network Topology... 64
Figure 6: Internal Activities on Host (Physical Layer) 65
Figure 7: Internal Activities on Router (Network Layer) 65
Figure 8: Classification Queuing Mechanism at the Network Layer 67
Figure 9: Simulation Results Showing Vector Values 68
Figure 10: Simulation Results Showing Scalar Values 69
Figure 11: Simulation Results Showing Histogram Values of Packet Transmission....... 69
Figure 12: Proposed Wireless Topology.. 70
Figure 13: QoS Effects of Packets Loss in Classification Procedure in Proposed Wired Scenario... 74
Figure 14: QoS Effects of Packets Loss in Classification Procedure in Proposed Wireless Scenario... 74
Figure 15: Overall Chart of the Total Number of Packet Flows (Transmitted Flows and Received Flows)... 75
Figure 16: Overall Flow of Proposed KNKM Design and Implementation..................... 83
Figure 17: Identified Clusters after Clustering Phase 85
Figure 18: Clusters Revealed Showing Inliers and Outliers 86
Figure 19: Precision Sine Wave Curve for KNKM Algorithm 86
Figure 20: Confusion Matrix Graph of True Classes against Predicted Classes 88
Figure 21: Area under ROC for KNN+K-Medoids Classifier.. 88
Figure 22: Confusion Matrix Graph Showing the True Class against Predicted Class in Wireless Environment (KNKM)... 89
Figure 23: Overall Flow of SVKM Design and Implementation 93

Figure 24: Precision Sine Wave Curve for SVKM Algorithm 95

Figure 25: Confusion Matrix of Accuracy Showing True Class against Predicted Class 95

Figure 26: Area Under ROC for SVKM Classifier 96

Figure 27: Confusion Matrix Graph Showing the True Class against Predicted Class in Wireless Environment (SVKM) 97

Figure 28: Flow diagram of R-TAC 100

Figure 29: Clusters Revealed after Initial Clustering Phase 101

Figure 30: Sine Wave Graph of Precision against Inter-arrival time for R-TAC 102

Figure 31: Confusion Matrix Graph Showing True class against Predicted Class (R-TAC) 102

Figure 32: Area Under ROC Covered for Classification with R-TAC........................... 103

Figure 33: Confusion Matrix Graph Showing the True Class against Predicted Class in Wireless Environment (R-TAC) 104

Figure 34: Comparative Graph of achieved Precision Range for all Classifiers 107

Figure 35: Comparative Graph of Confusion Matrix Showing Classified and Misclassified classes for Each Classifier 109

Figure 36: Comparative Graph of achieved Accuracy with Error Rates for Each Classifier 110

Figure 37: Comparative Graph of ROC Curves Showing the Area Under ROC for Each Classifier 111

Figure 38: Confusion Matrix of Classification of Classifiers in Wireless Scenario (FPL 2) 114

Figure 39: Confusion Matrix Graph of Proposed KNKM on Cup KDD’99 Dataset 120

Figure 40: Confusion Matrix Graph of Proposed SVKM on Cup KDD’99 Dataset...... 121

Figure 41: Confusion Matrix Graph of Proposed R-TAC on Cup KDD’99 Dataset...... 121

Figure 42: Confusion Matrix Graph of KNN on Cup KDD’99 Dataset......................... 124

Figure 43: Confusion Matrix Graph of KNN+K-Means on Cup KDD’99 Dataset........ 124

Figure 44: Confusion Matrix Graph of IATP on Cup KDD’99 Dataset......................... 125

Figure 45: Confusion Matrix Graph of KNKM on MACDCC IDS Trace 127

Figure 46: Confusion Matrix Graph of SVKM on MACDCC IDS Trace...................... 127

Figure 47: Confusion Matrix Graph of R-TAC on MACDCC IDS Trace 128

Figure 48: Confusion Matrix Graph of KNN on MACDCC IDS Trace......................... 130

Figure 49: Confusion Matrix Graph of KNN+K-Means on MACDCC IDS Trace 131

Figure 50: Confusion Matrix Graph of IATP on MACDCC IDS Trace 131

List of Tables

Table 1: Analysis of Efficient Unsupervised Clustering methods 57

Table 2: Analysis of Efficient Semi-supervised Clustering Methods 58

Table 3: Simulation Parameters for Proposed Wired Topology 66

Table 4: Simulation Parameters for Proposed Wireless Topology 71

Table 5: Flow Statistics of FPL Dataset 72

Table 6: Statistical Breakdown of P2P Flows in FPL Dataset 72

Table 7: Statistical Breakdown of Streaming Application Flows in FPL Dataset 72

Table 8: Statistical Breakdown of Mail Application Flows in FPL Dataset 73

Table 9: Statistics of Transmitted Bytes and Received Bytes after Simulation 75

Table 10: Summary of Machine Learning Classifiers 79

Table 11: KNKM Proposed Hybrid Algorithm 82

Table 12: SVKM Proposed Hybrid Algorithm 92

Table 13: R-TAC Proposed Algorithm 99

Table 14: Comparative Summary of Attained Outcomes for Each Classifier with FPL 1 Dataset 112

Table 15: Comparison of Classification Accuracy (Wired Scenario Vs. Wireless Scenario) 115

Table 16: Classification Analysis of Classifiers with Traffic Flows Classified and Misclassified on Cup KDD’99 Dataset 125

Table 17: Classification Analysis of Classifiers with Traffic Flows Classified and Misclassified on MACDCC IDS Trace 132

Table 18: Summary of Classification Accuracy Results 133

Chapter 1: Introduction

Network and internet communications have become a pivotal aspect in which our work and social life evolve. In order to provide swift health care services, banking services, application of jobs, research work, keeping in touch with family and friends, security services, etc. applications that require internet services have been developed to this aid. With all its contributions and benefits, the complete operation and detailed comprehension of computer networks and internet are limited. This is a result of the rapid evolution of computer network architectures, protocols, traffic, and applications. The research community has therefore been motivated to breakdown this ambiguity and help with the understanding of internet and computer network activities. A branch of this formed a novel area of study known as Traffic Classification. This field of study is relevant and indispensable to unravel the complicated operations of the internet.

1.1 Traffic Classification

Data sharing over networks is one of the optimal methods to transfer and receive information in today's information era. When data is sent from a source network, the traffic can traverse on various hops depending on the path before it gets to the destination network. Traffic classification involves the process of associating examples of traffic generated to the specific source applications that resulted in its generation [1]. The word traffic in the context of this work exemplifies IP traffic, internet traffic and private network traffic, as well as packets or data flows within a network. The traffic can be intercepted or changed at any phase through the traveling process. It is therefore crucial for administrators of all networks to identify the traffic that enters their network. This requires traffic or packets to be classified. Modern years have demonstrated that Internet has proceeded with its sensational development. Use of assorted applications has expanded, particularly advanced applications likened to distributed applications and multimedia has turned out to be generally utilized. These days, managers face gigantic difficulties to distinguish distinctive applications roaming on the Internet. The necessity of classifying internet traffic intensifies for administrators for various reasons such as guarding the system security, traffic engineering and monitoring of usual activities. The task of identifying applications is at the focal point of numerous security functions. An enterprise can have security blueprints to allow specific kind of applications while preventing others from entry, access or use. This could be exclusively determined by security, making it a great concern for Quality of Service (QoS) delivery. Particular kinds of application traffic in relation to QoS may get special treatment above other applications. For instance, in an office domain, email traffic may be prioritized over downloads of video files. Usually, these priorities placed on traffic types and how frequent they are utilized call for such special treatments. With respect to smaller network environments, traffic characterization may be required, for example, home systems, where no expertise personnel are available. Applications used to upload and download content onto and from the internet (such as U-torrent) and application platforms for viewing and streaming video content (YouTube), at a certain phase dynamically changed the criterion of internet architecture. With the influence of such sudden transformations in traffic trends on networks, traffic classification aided administrators to strategize and design architecture protocols to the effect of such new flows.

The concept of Real Time traffic classification using clustering techniques or algorithms involves the process of identifying packet traces or portions of the traffic which are similar, possessing distinct features and grouping them under unique headings. Clustering techniques have several applications in a network system. For instance, they are the precursors to intrusion detection systems, which use anomaly detection [2] [3] [4]. Furthermore, traffic classification is also applied to strengthen security applications and network management to boost the quality of service in smaller and larger networks [5] [6]. In addition, classifying application traffic accurately is required to improve the efficacy of network resource usage [7]. The traditional method of classifying traffic includes the Port based approach and Payload method of classification. Other approaches include Behavioral classification and Statistical classification

1.1.1 The Port-based Classification

During the initial deployment and early existence of the internet, network traffic classification was not a subject of contention by any means. Port-based approach makes use of known ports in the list of registered ports ascribed from the Internet Assigned Numbers Authority (IANA) [8]. This includes determining an application dependent on assessing the packets in headers and coordinating it with its corresponding port number enlisted with the Internet Assigned Numbers Authority. Some recent emerging applications in the same manner as peer to peer traffic [9] similarly adopt unregistered port numbers, non-standard ports, or can select an irregular port. This increases the results of misclassification. In the worst scenarios, inappropriate applications conceal themselves behind certain known ports in order not to be detected. When a session connection is established, the target port is sought using the application and the traffic is classified based on this information. In a few circumstances, there is a difficulty in realizing genuine port numbers in the situations of obscurity packet headers in the process of IP layer encryption. In any case, because of the infringement of port number assignments by an ever increasing number of recently rising applications, the technique has turned out to be progressively off base [9] [10] [11] [12] [13] [14]. The approach, though successful, experienced some drawbacks as newer trends in traffic emerged. Recorded improvements have uncovered the errors and shortcomings of these conventional strategies [15]. The limitations stem from various causes. Applications which invoke the use of HTTP server may use other standard port numbers instead of the originally assigned port. Also, the use of dynamic ports and encryption of the IP layer which also emerged made it difficult to find the genuine port number [16]. The approach could not classify these encrypted types of packets in the traffic flow. The port-based approach is explained into details in Schneider [17] and can be referred for in-depth understanding.

1.1.2 The Payload Classification

To solve the inadequacy and dependence on the port-based method of classifying traffic, numerous research works suggested not only examining packet headers but including other features, in a method referred to as Payload Classification. In the Payload classification, packets found in flow payload are examined carefully to find known signatures [18]. Classification is done based on the similarities of these signatures with the knowledge-based signatures, which has already been trained with the classifier [19]. Payload scrutiny has been broadly utilized in open source systems like the Linux kernel firewall implementation process [20]. Additionally, intrusion detection systems (IDS) utilize payload-based classifiers frequently for distinguishing malevolent actions in networks [21]. The unwavering quality of this technique has been explored generally. Sen et al. [22] research work showed the effect of utilizing payload approach to detect P2P (Peer- to- Peer) traffic. They uncovered that it could limit the bogus misclassification rate by only 5% in most examined cases. There also exists different research dependent on payload classification [18] [23] [24] [25]. In spite of the fact that payload-based technique is viewed as a dependable procedure, it has some critical impediments and shortcomings. To begin with, the technique is challenged by the difficulty in updating the database of application signature, to enable effective comparisons. They can either coordinate the identified signatures in payloads packets or approve application-layer convention message groups. Also, the Payload method could not also classify encrypted data and handling of proprietary protocols became a challenge [16]. The capacity of the classifier reduces when encrypted traffic is classified or when protocols are encapsulated; analyzing of packets which are encrypted with this technique is unfeasible, which implies tons of system traffic stays unclassified. These traditional methodologies have high computational overheads and impose a lot of load on the device used for classification. As a result, they face trouble in adapting to the huge number of flows and the fast rate at which the network traffic propagates. With the advancements in technology, trends of how data is sent over networks changed which made masquerading of data another issue related to the traditional approach.

1.1.3 Behavioral Classification

Behavioral classification strategy deals with examining the entire traffic patterns from the network generated by the hosts with the aim of identifying application types of data from selected or particular hosts. For instance, the quantity of conveyed hosts is tallied, considering the total ports and transport layer protocol. The authors in [26] [27] for example, used heuristic information like unique ports communicated with and exploited transport layer protocols to examine network traffic patterns and identify the kind of application that runs on the host. Different works [28] [29] demonstrated that network traffic can be classified by utilizing tons of information. They graphically investigated the connections between hosts, and they demonstrate that patterns in the created connections and client-server application graphs are not very quiet the same as those of peer to peer. Other researchers [30] [31] used the capabilities of ML algorithms together with some metrics for classification of individual network applications. Despite the behavioral classification yielding encouraging outcomes with lower computational cost, the greater part of these proposed works considered just the movement of the endpoints or hosts [32] [27].

1.1.4 Statistical Classification

Because of the confinements of the previously mentioned methodologies, a huge number of late investigations have been concentrating on statistical-based approaches. The motivation stems from the notion that traffic produced by various applications shows discernable attributes. Statistical classification involves making use of statistical features or factual attributes of flows in network traffic to distinguish applications. It makes use of various flow-level quantifications [33] [10] [9], like inter-arrival time of packets, idle time of flows, size of packets and length of packets. These estimations are distinctive for peculiar applications. Subsequently, this enables the classifier to separate distinctive applications from one another. Initially, network traffic statistical features were examined in some few works. Paxon [34] looked at the correlation existing between the class of traffic and characteristic feature of the flow, for example, the number of bytes and flow length. He recommended setups of experiential association attributes for a substantial number of applications. Likewise, Dewes et al. [35] used the network flow statistics, for example, the time duration of flows, inter-arrival time of packets, packet size to examine chat systems across the internet. Works from research conducted later, for example, [36], [37] and [38] looked at the distinct features of network traffic for various web applications. The results from the examinations have enlivened analysts to delve into modern dimensions of classification procedures dependent basically on statistical attributes and features.

In order to implement statistical classification, classifiers need to utilize data mining methods (to be precise algorithms in Machine Learning (ML)) since they have to manage diverse patterns of traffic from extensive datasets. The algorithms pertinent to ML are exceptionally lightweight and cost less in processing and resource consumption in comparison to the payload-based method. They are not based on packet inspection, but rather make use of the data extracted from flow analysis conducted. The adequacy of classifiers based on statistical approaches relies on the extricated features of flows, which stand in need for broader learning because of their multifaceted nature. Nevertheless, these procedures perform better compared to payload classification since they try not to include content from packets, hence encrypted traffic can be examined and classified without problems.

1.2 Machine Learning Techniques

The newer approach for classifying traffic is the use of clustering techniques which forms part of Machine Learning (ML). Machine Learning approaches are advanced in terms of accuracy, performance, and complexity compared to traditional approaches. ML enables a system to train itself with an existing database, from which the system later infers appropriate decisions regarding the traffic classification. That is, it enables a system to train itself with information fed to the system and later take the appropriate decision when required based on the prior knowledge or information it has [39]. The information normally represented as a dataset. The way of learning falls specifically under

Supervised, Unsupervised and Semi-supervised [40]. The supervised approach involves using an algorithm to study and make predictions from a labelled dataset. The dataset is also referred to as the training dataset or examples. A preexisting knowledge of the nature of the anticipated output is known. The algorithm learns from the training dataset which serves as a guide to infer the output of new examples that will be fed to the system. Thus, subjecting the process to supervision [41]. Unsupervised Learning as the name suggests has no supervision. The input dataset is unlabelled, hence prior knowledge of the output is not known. Algorithms are used to divulge and reveal the inherent structure of the data. The technique of unsupervised learning associates data elements possessing comparative qualities together from the unlabelled dataset. Even so, it has lower accuracy in classification and exhibits strenuous processes for training in contrast with the supervised methods. Be that as it may, there are two primary difficulties for classifying traffic utilizing these ML techniques. Right off the bat, labelled examples are rare and hard to get. With limited labelled tests, the indigenous supervised strategies frequently produce classifiers that don’t sum up well to already concealed examples. Besides, not a wide range of application examples is known and new ones may show up after some time. Conventional supervised techniques compel an alignment of each example into one known class, eliminating the capacity of recognizing newer patterns of samples. This fostered the initiation of Semi-supervised learning. Semi-supervised descends as part of both supervised and Unsupervised Learning. The method makes use of some amount of labelled data infused, coupled or mixed with unlabelled data. The unlabelled data forms the majority of the dataset fed as input to the system. A mixture of labelled and unlabelled brings some form of supervision in the classification of the data [42]. Clustering techniques fall under unsupervised and semi-supervised learning and mostly has to deal with the association of some characteristic features [39].

There are different methods of clustering, namely classic K-means clustering, hierarchical, Density-based clustering, Grid-based clustering, and Probabilistic-based clustering. [43] [44] [45] [46]. Hybrid approaches to these methods are also in existence. The classic K-means method divides the dataset into a disjointed set of clusters and exemplifies each cluster with its centroid whereas the Hierarchical clustering methods are more settled generating a clustering hierarchy [47]. Probabilistic model-based clustering presumes data is formed by an assortment of the inherent probability distributions among various populations intended to be described by its characteristics. The advantage is, the clusters formed are effortless to interpret [47] [46]. On the other hand, with Density Based, clusters are defined to comprise of unpredictable shapes. This characteristic provides protection against outliers and noise [48]. Grid-based partitions given data spaces into a multi-resolution structure of grids with a finite number of cells [49] [50]. The method's collection of grid data contributes to the independent attribute of the Grid-based clustering approach of data ordering [46].

The clustering approach has been used by many researchers to promote adequate ways to classify traffic from various networks. Most of the approaches extract and examine the statistical features of packets which encompass packet size and distribution, arrival times between packets and packet length [51]. Others also make use of the patterns in the traffic. Their approaches have yielded higher results proving clustering techniques as an efficient method. With further research into this, better efficient methods than the existing ones can still be developed for different application areas. The selected works in literature to estimable cognition has led to most proficient results in the literature as at the time of conducting this research. Hence, unsupervised and Semi-supervised methods using clustering analysis form the scope of the literature. Where supervised methods are incorporated in semi-supervised suggested works, we discuss the former for better understanding. The results obtained from the works are also discussed in terms of its accuracy in classification, performance, complexity in computations and run time where applicable.

1.2.1 Clustering Approach of Classification

The method of clustering bears on the process of classifying given amount of objects such that similar ones fall into a common class also called a cluster [46]. Thus, a cluster contains objects with similar characteristics which are distinct from objects in other clusters as represented in Figure 1. Each colour (red, blue, green) represents a cluster. Objects and points with familiar features are represented in the duplicate colours, constituting a cluster.

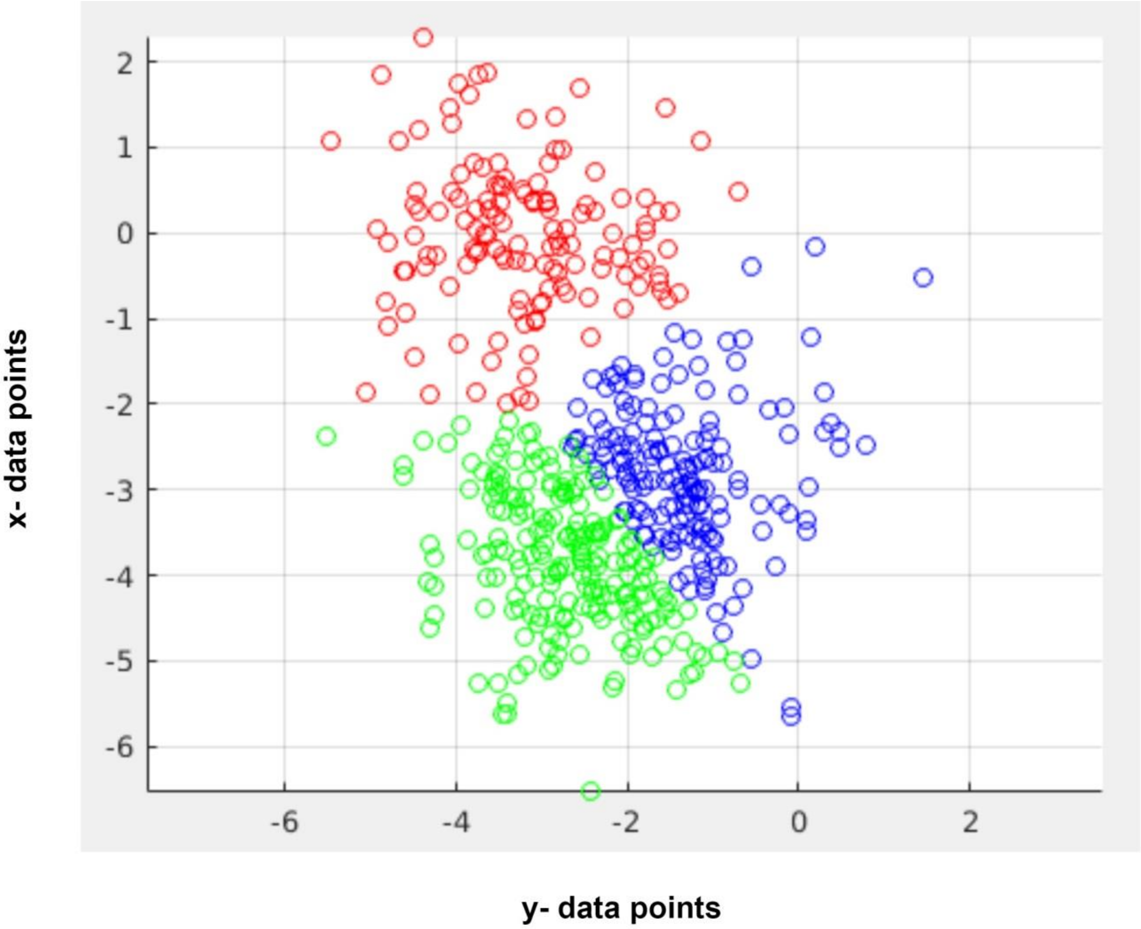


Figure 1: Cluster Representation

The magnitude of how objects are similar or dissimilar to each other determines the particular cluster which they will belong. The perception of a cluster cannot be described strictly, but seen in diversity as a limited distance in scope within other cluster members, dump orbits of data space, or possessing intervals and exceptional distributions of statistics [52]. Cluster parameter settings include distance function and the total number of clusters to expect, also termed as the threshold of density. The point of concentration in a cluster for cluster analysis is mainly the middle section of the cluster known as the centroid, which is usually denoted and located by a vector.

Clustering approach is unsupervised since no labels are set or defined beforehand. It is therefore difficult to adjudicate whether clustering is right or wrong. However, careful consideration of certain factors, including the standards for partitioning, the distance between clusters, the quality of clusters and scalability supports the method of proficient and effective clustering.

1.2.2 Standards for Partitioning

The standard for dividing data traffic into clusters, whether hierarchical or non-hierarchical, contributes to the effectiveness of a cluster. Hierarchical partitioning creates a form of classification in which closely related objects form smaller clusters and become sub-clusters of a larger cluster [53]. This form of partitioning can further be categorized into Agglomerative and Divisive. Agglomerative starts from the bottom and proceeds upwards into building a hierarchy. It begins with small single clusters and fuses two clusters in a continuous manner to form a larger group of clusters. The divisive approach, in contrast, conforms as a building from the top and proceeding downwards where a single cluster is divided in a continuous manner to form sub-clusters [54]. Non-hierarchical partitions refer to clustering a given dataset into non-occurrence groups, with a tree-like structure. Categories of non-hierarchical partitioning are single pass, relocation and nearest neighbor approaches. Single-pass generates clusters that rely on the arrangement of the dataset, whereas relocation categorizes data into a known quantity of clusters and re-assigns them into finer clusters. The nearest neighbor approach puts objects of the given data into clusters with respect to the similarities of their nearest neighbour [52].

1.2.3 Distance between Clusters

A clustering algorithm must consider the separation of the clusters. The decision of a data object or flow’s exclusiveness to only a single cluster or multiple clusters must be considered. The similarity measure of distance must also be factored, for example, to choose a Euclidean distance or a connectivity-based method [54].

1.2.4 Quality of Clusters and Scalability

The quality of a cluster deals with its capacity to handle noisy data, the formation of clusters with discretional or absolute shape, and dealing with different data types. A proficient cluster should be able to cluster all the data objects instead of a representative sample and also cater for high dimensional data [55]. Data must be scalable to prevent the wrong representation of results in the clustering process.

1.3 Clustering Algorithms

Clustering classification makes use of a set of instructions or codes to determine how the overall classification process is done. These sets of codes define a clustering algorithm. Clustering algorithms are usually categorized into Linear and Non-Linear algorithms. Linear algorithms are suitable for datasets with low variance. Less or no change or input is required. Examples of Linear algorithms for clustering include the hierarchical algorithm, K-means algorithm, Quality threshold algorithm, Gaussian (Expectation Maximization) algorithm and Fuzzy C-means algorithm [55]. Contrarily, Non-Linear suits datasets with very high variance requiring the number of features to be reduced or dataset to be manipulated. Examples include Density-based algorithm, MST (Minimum spanning tree) algorithm and Kernel K-means algorithm. An efficient or good clustering algorithm must possess but not limited to the characteristic of:

- Being able to cluster datasets of high dimensions
- Dealing with a variety of attributes
- Identifying outliers and noise

- Generating results that are easy to interpret to provide more insight on input parameters
- Identifying arbitrary shaped clusters

1.3.1 Application Areas of Clustering

Clustering techniques can be incorporated in Data summarization, data compression and reduction procedures of data mining and storage [56]. This makes clustering applicable to multimedia data (image, audio, and video), sequential data [57], uncertain data such as noise and big data. Furthermore, clustering can be used to observe extreme outliers in sets of data. Outliers consist of data that fall outside the parameters of any cluster. This makes clustering useful in detecting patterns and dynamic trends in traffic flow. Other application areas of clustering include determining genetics and DNA in Humans, providing analysis of crime cases, categorizing results of search engines, analysis in climate science, sequence analysis in genome, analysis in social networks and web applications [58][59][60].

1.4 Motivation of the Investigation

The effects of quality of service parameters on homogenous and heterogeneous networks on entire network execution have resulted in a major concern for most network administrators. The scope of this research concentrates on two QoS parameters and that is Packet loss and Fragmentation. These two parameters go hand in hand such that, the packet size and length is as a result of fragmentation. Depending on the measure of resources the network possess, (bandwidth, link capacity, and speed) determines what amount of packet loss the network experiences. The objective is to find out the effects or how these QoS parameters will behave in the clustering procedure in machine learning in relation to their performance. Not much investigation has been done on this to utmost knowledge hence the aim to delve more into this notion and propose an algorithm that is able to withstand such conditions.

1.5 Objectives of the Study

1. To investigate the effects of Packet Loss and fragmentation in Network traffic classification
2. To develop QoS aware semi-supervised clustering algorithm that will improve upon the performance of traffic classification in networks efficiently in the presence of Packet loss and Fragmentation.
3. To implement and validate the developed algorithm to serve as a classifier to generate a compact summary of various traffic types while increasing the performance in presence of QoS Parameters.

1.5 Contributions and Scope of the Study

The primary aim of the work under study is to provide a novel classification approach for wired networks which at a point in time due to increasing requirements of the network leads to insufficient resources. The classification occurs at the flow-level. For this, a network in a case of low-speed links is implemented in simulation. The QoS effects of it are captured in real traffic data logs to formulate a dataset for an experimental framework analysis. The dataset is utilized as ground truth in the training and assessment of proposed and existing classifiers. Furthermore, based on the requirements of the proposed classification strategy, algorithms are proposed that will be fully equipped to classify the traffic flows efficiently. Before the formulation of the proposed classification strategy, an extensive literature survey on existing works are carried out to study how they were formulated, parameters considered, achievements and flaws in them. Some of the models are selected based on the above mentioned areas and considered for an improved approach for the problem identified for the study.

1. The relevance of the undertaken objectives and the proposed work offers a good technique for traffic classification which is of significance in seamless data transmission. The objectives ensure a quantified amount of literature is analyzed

to identify the gap in knowledge with respect to the problem statement. A summary of prominent works with most efficient results is presented to serve as a base groundwork to save cost and time for other research works [61].

2. It provides an alternative approach for networks seeking to achieve good classification of traffic with the limited resources provided by the network. With the absence of datasets that considers the effects of QoS parameters when a network has low-speed links, real-time traffic data logs are captured and formulated into a dataset for evaluation and comparison with other existing works.
3. The research further enhances the network's quality of service provided by network administrators and service providers. The suggested algorithms implemented into classifiers are efficient in classification when networks have limited resources [62] and also validated to be efficient even when requirements are met using existing datasets that do not consider such parameters. Therefore network providers QoS are never undermined with respect to traffic classification at any phase in the life of the network.

1.7 Research Methodology

To improve the Quality of service in Networks such as Wide area Networks (WANs) and Campus Area Networks while classifying the traffic coming into the networks, we first delve into Machine Learning and Clustering Techniques to identify persistent challenges that inhibit better quality of service and poor classification through systematic, well-structured and authentic literature review. The research gap is identified to formulate the research problem and objectives. The overall goal is to classify application traffic flows using restricted traffic features, quantified number of packets to maintain a subsidized complexity in time and good classification accuracy.

1.7.1 Description of Suggested Objectives

With respect to Objective 1 **(To investigate the effects of Packet Loss and fragmentation in Network traffic classification),** a network scenario is designed in OMNET ++ simulation and subjected to parameters that contribute to low quality of service which is accelerated loss of packets and inconsistent fragmentation. The method of simulation is preferred because it provides the resources to investigate into worst scenarios of network situations, and also afford adequate room to test various parameters during experimentation at a low cost. The logs generated from the implementation of the network scenario are analyzed to find the effects of packet loss and fragmentation in the classification procedure. The logs are filtered and transformed into a data set for analysis.

To achieve this objective **(To develop QOS aware semi-supervised clustering algorithm that will improve upon the performance of traffic classification in network efficiently in presence of Packet loss and Fragmentation)**, we propose a semi-supervised algorithm by exploiting the advantages of renowned works in literature to create a hybrid algorithm (classifier) that is able to withstand the conditions that deter the quality of service in the classification procedure.

With respect to objective 3 **(To implement and validate the developed algorithm to serve as a classifier to generate a compact summary of various traffic types while increasing the performance in presence of Quality of Service Parameters.)**, we implement the proposed algorithm in MATLAB on the generated dataset. The metrics used for the evaluation are confusion matrix, accuracy, precision, error rates, and time complexity. We validate the proposed work by comparing with other classifiers that proved to be efficient in literature. A comprehensive flow of methodology for the groundwork is displayed in Figure 2.

1.7.4 Hardware and Software Requirements

The system and simulation requirements of the study include:

- RAM – 10GB

- Processor type – Intel Core i3
- Hard disk size – 500GB
- Selected operating System for Simulation – Ubuntu 16.04
- Windows 7 Professional for Editorial purposes
- OMNET ++ - Version 5.3
- INET – Version 3.6.5
- MATLAB – Version R2017b

1.8 Thesis Structure

The write-up is organized systematically in chapters. Introduction of the study, motivation, suggested objectives and adopted methodology for accomplishing the targeted objectives are covered in the first chapter. The second chapter presents the history and background knowledge of all aspects of the study. This includes the scope of networks, parameters employed, and a comprehensive review of precedent works conducted. The work conducted with experiments is subsequently discussed in the next three chapters. The third chapter elaborates the topology design and execution. The architecture and exertion of the prospective algorithms are deliberated in the fourth chapter. Analysis of the outcomes achieved and verification constitute the fifth chapter of the study. Validation and analogy of the suggested works against other works and already in existence is performed in the sixth chapter to deduce conclusions. The seventh chapter summarizes the study with its findings and prospective additions that can be made in the future. The taxonomy of the study is demonstrated in Figure 3.

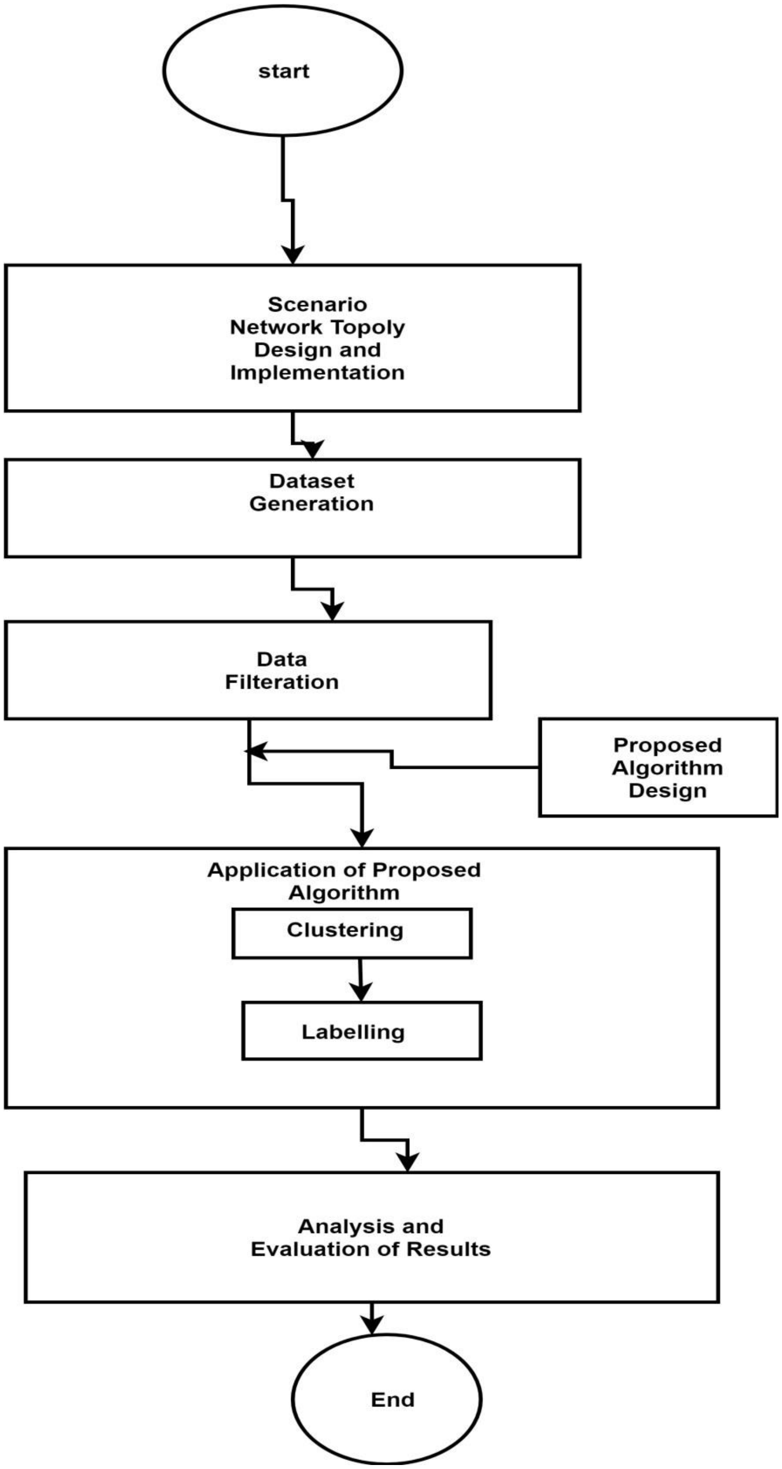


Figure 2: Flow of Proposed Methodology

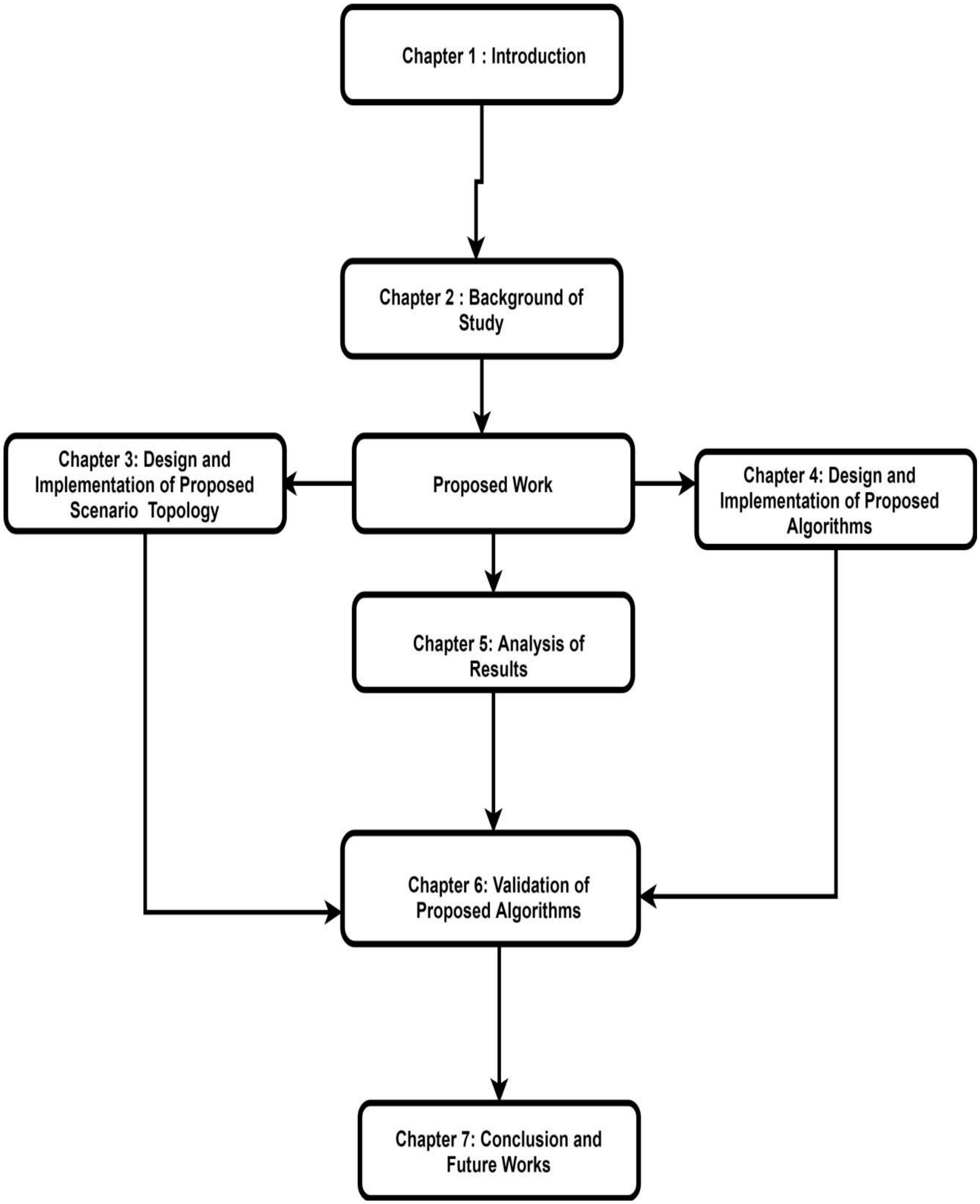

Chapter 1 : Introduction
Chapter 2 : Background of Study
Proposed Work
Chapter 3: Design and Implementation of Proposed Scenario Topology
Chapter 4: Design and Implementation of Proposed Algorithms
Chapter 5: Analysis of Results
Chapter 6: Validation of Proposed Algorithms
Chapter 7: Conclusion and Future Works

Figure 3: Structure of the Thesis

Chapter 2: Background Study

2. 1 Packet Loss Parameter

Triad parameters that affect network performance, in general, are latency, loss and jitter metrics [63]. Latency can be explained as the amount of time (in milliseconds) a data packet takes to journey to its destination and back again. This amount of time is known as the round trip time (RTT). Even though latency can be measured in a one-way trip, it fetches more costs and requires advanced instrumentation compared to the round trip measurement. Pertaining to the performance measure of QoS, latency falls under the Quality of Experience (QoE) metric. The variation of delay or latency from one point to another in a network depicts the jitter, which is the significant difference between the latency from one packet to another on a data path. Depending on the jitter buffer of networking equipment constitutes the amount of jitter bearable by the network. Packet Loss is the measure of packets misplaced during the journey of packets from a source point to a destination point. The network experiences packet loss when some of the packets are not received at the intended destination. This is very common in the real world of networks. The behaviour of networks to riffle or alternate occurs from time to time.

2.1.1 Packet Loss versus Latency

In the scenario of phone calls where a caller places a call to the receiver, latency will be the time it takes for the caller and receiver to have effective communication. The concept of queuing theory shows that when a link is more busy or congested, more packets will have to wait in the queue. When there is a high latency in transmission, packet loss also increases contributing to some packets being discarded to reduce the congestion, especially on low-speed links. Also, there will be no or limited space in the TCP buffer. This will cause transfers of packets to halt until the lost frames have been retransmitted and received at the destination. Some UDP- based applications can show its response to packet loss by resending lost frames, corrupting the data or even terminating the active connection. Much of these effects are experienced in VoIP applications where one can hear feedback or echo as well as distorted audio. Other elements of the network and buffers try to fix the losses of packets which in turn cause delays which are easily seen in conversations. This depicts that when the latency is high, packet loss increases. There can be a solution around it but that will also cause TCP activities to slow down rapidly.

2.1.2 Packet Loss versus Throughput

Some packets make it through to its destination even though some are lost on the way. The amount of data that travels or traverse through the network successfully is referred to as the throughput. The effects and impact of latency in the network are not eradicated by bandwidth increment. The time it takes to transmit a packet between two nodes A and B, across a network and to receive a response back impacts the network as a whole. Since the speed at which the packet travels cannot be greater than that of light, the amount of data that can travel successfully (throughput) greatly depends on the distance between the nodes in the network.

The operation of TCP is such that when a packet is lost usually in wired networks as a result of congestion, the degree of packets sent is slimmed down to 50% of the prevailing size of the congestion window (cwnd). Assuming *cwnd* is 4, it will be lowered to 2 for the process of slow start in TCP to begin again. In order to find the appropriate and optimal throughput, the congestion window will be increased by 1 segment gradually. In situations of high packet loss, this will result in overall lower levels of throughput.

2.1.3 Packet Loss versus Fragmentation

The maximum size of packets or frames varies as a result of different networks that are connected via the internet. The maximum size otherwise the maximum transfer unit (MTU) of the data payload of 802.11 standard is normally 2312 bytes. Other forms of the 802.15.4 standard have 104bytes as the MTU [64]. Fragmentation occurs when packets get onto a link with lesser maximum size than the packets arriving. Fragments of packets are then transmitted in packets independent of each other. At the end of the transmission, with all fragments received, they are reassembled at the end host.

2.1.3.1 Effects of Fragmentation on Packet Loss

MTU size puts a limit the maximum size of packets that traverses on a link. Fragmentation, however, helps to overcome that limitation but that can also affect the performance of the network negatively by incurring extra costs and complexity due to the reassembling of protocols and the fragmentation process. However, there is a high probability for packet loss to increase since more packets for each of the primary packets must be transmitted. In networks with limited resources, the repercussions of packets loss may be very critical.

2.1.4 Causes of Packet Loss in Networks

The network layer constitutes the building blocks for several applications and data that traverse on the network. Hence, problems arising from this layer will affect the other layers that operate on top of this layer negatively. Packet loss is one of the adverse effects networks suffer. The reasons why packets get dropped in a network includes link congestion, network device performance, software errors and issues on network devices and faults in cables and hardware.

2.1.4.1 Link Congestion

During a trip across a network, data traverse through several devices and links. When a link has utilized its capacity fully, incoming data arriving on that link must wait in a queue for its turn to be given access to travel on the link. Depending on the length of the queue, when the buffer space of the network device is full, the incoming data will have to be dropped and discarded. However, several applications are able to handle it in such a way that it will be unnoticeable by the end user. The transfer speed of packets is slowed down in any stage a lost packet is encountered, which initiates a process to re-transmit the missing packets again. If packet loss does not occur continuously or consistently, legitimate time applications like email retrievals and downloading of files normally experience minimal effects of it. On the other hand, applications which include phone and audio calls, video messaging and chats have its users noticing the effects of the lost packets such as missing parts of audio and video distortion.

Increasing the bandwidth of congested links can help reduce congestion to a certain level. Also, configuring QoS to give much priority to some of the real-time applications like audio and video will not eliminate but can go a long way to reduce the congestion on links.

2.1.4.2 Network Device Performance

Network Devices includes switches, routers, and firewalls. Each device has its working performance as to how much traffic it can keep up with. Bandwidth increase can help to reduce packet loss to a certain level in that when the network device is not able to process the traffic that comes to it, packet loss can still occur. This can result because the network device has reached the maximum throughput that it can allow for. The device's memory has exhausted its capacity or CPU processing capacity, therefore dropping the packets when the traffic reaches the device.

It is required to replace the device with a new one or add another device to an existing one to keep up with the incoming traffic and handle the throughput to its maximum.

2.1.4.3 Software Errors on Network Devices

Bugs in software can cause device malfunction and make the device not behave the usual way it is supposed to. Due to the complexity of network devices, it takes time for one to detect these bugs. Furthermore, some features do not work well or might even not work entirely. Since it takes some time to detect it might cause performance issues. Most of these performance issues are mostly found in packet captures and system logs.

Frequent upgrade and updates of the device software are required for the devices that have been affected.

2.1.4.4 Cabling and Hardware Faults

Another common cause of packet loss is faults with the physical components such as cables (e.g. fibre optic and copper cables), pinched cables, corrosions, poor crimping and faulty connectors. Hardware malfunction can generate errors in system logs and the device console.

The best remediation is to replace faulty hardware or repairing faulty links if detected.

2.2 The Network Study

When computer components, systems, and devices connect in order to share or transfer resources through media or communication channels, it can be termed as a network [65]. A simple example of an existing network can be a couple of computers in connection to each other by a wired media, with the two being able to send or access information (files and documents) between each other. Networks can also be more robust and complicated such as several networks linked together to formulate a larger network. The type of network can be defined by the geographical area the network covers. With respect to that, network types can be grouped into three major groups namely Wide Area Networks (WANs), Local Area Networks (LANs), and Metropolitan Area Networks (MAN) [65]. Other varieties include Personal Area Network (PAN) and Campus Area Networks (CANs). The type of media used in the network can also define network types such as Passive Optical Local Area Network (POLAN).

Depending on the type of devices connected to the network for communication, a network can be also grouped into Homogeneous or Heterogeneous.

2.2.1 Local Area Networks (LANs)

A LAN usually connects devices, workstations, and computers in a setting or buildings closer to each other or covering a small geographical area. Switches, Hubs, and Ethernet cables are some of the hardware devices used in LANs. Routers are used to connect

LANs to bigger networks and other LANs together to share resources, data, and information. Twisted pair and coaxial cables are examples of transmission media used in LANs. When devices are linked by wireless technology in a LAN, it can be referred to as a Wireless LAN (WLAN). This is made possible by using wireless access point devices to serve as a bridge between the computer devices and networks.

2.2.2 Wide Area Networks (WAN)

Networks covering larger geographical areas and spans constitute a large area network. A group of LANs connected to each other through radio waves, fibre optic cables, microwaves, satellites and can be restricted or made accessible to the public domain. Routers are used to route traffic from one network to another when the destination is not within the same network. The internet is an example of a WAN. An organization or enterprise with several branches can also set up a Wide Area Network to share its data and information among the branches and headquarters. WANs are more prone to errors due to the distance the transmit covers compared to LANs.

2.2.3 Metropolitan Area Networks (MAN)

MANs cover a larger geographical area compared to LANs but also smaller compared to WANs. It is formed by integrating components from both networks. In this network, computers or systems at different locations within different or same towns are connected to share resources or information. A MAN can be set up for a town, city or even a campus. When several LANs are interconnected with a backbone line, a MAN can be formed and sometimes referred to as a Campus Network. Congestions are more prone and the network possesses low fault tolerance. An organization can be responsible for managing and maintaining the network since it is can be very difficult to design and maintain by an individual.

2.2.4 Personal Area Network (PAN)

An individual can set up a network personally for himself or herself with computers, phones, wireless modem, cables, printers, etc. within an office, home or a small room. This type of network is called a Personal Area Network or PAN.

2.2.5 Campus Area Networks (CAN)

This network is designed and normally used for universities, larger or bigger schools, a business or districts. The geographical coverage is usually smaller than MANs but a smaller version of a MAN can be a CAN.

2.2.6 Storage Area Networks (SAN)

The technology of a storage device connected or attached to a server is similar to that of storage area networks. Pools of storage devices are connected to a number of servers. These storage devices and resources are placed at a different location away from the network to establish another performance network with high speeds. SANs do not necessarily depend on a Local or Wide Area Network.

2.2.7 Enterprise Private Networks (EPN)

Most organizations, businesses, and enterprises prefer to build their own network to enhance communication, resource sharing and security (data protection and privacy). The network links the other locations or branches of the enterprise together.

2.2.8 Virtual Private Networks (VPN)

In scenarios where private and enterprise networks have to extend their network securely over a public or internet, VPNs come into use. It allows users of a private network to send, receive or access data remotely when they are not physically present or at a different location where they cannot connect to the private network. It gives the virtual connection to the users but seems as if they are connected physically to the private network. A virtual point-to-point connection is established to make access to the private network possible.

The scope of the study covers wide area networks where a lot of traffic is generated and congestion is likely to occur. WAN is selected not only because of its geographical coverage but also for its ability to permit more complexity in its design. WANs are also widely used with the internet being the largest of all; therefore the research can be implemented and applied in many areas and sections of today's global world.

2.3 Network Traffic Classification Techniques

In literature, classification of traffic is generally grouped into signature-based, flow-based methods and statistical methods. With signature-based, the strategy recommends employing signatures composed of behaviour adopted by application conventions and protocols. Networks of Palo Alto that employs the unification of thread mitigation [66] utilize decoding the encoded traffic along with their respective signatures to recognize different applications. Taking inspirations from the drawbacks of deep packet analysis in signature-based methods such as being inclined to errors and impeded, Tongaonkar et al. [67] suggested a mechanized mark extraction approach which is fit to find new applications. Identified signatures depend on payload content of packets and are created with proportional sections of payloads in various streams. Furthermore, encoding of signatures into text content can be added. Statistical methods utilize statistics and insights determined out of packets [13] being arrival time between packets, average size of packets and application distribution which serves as features to distinguish between applications. Branch [68] employed two features that is inter-arrival time and length of packet in training his proposed algorithm to reveal the class of traffic. The decision of arrival time in between packets is an appropriate parameter since applications executed in actual time keep strict time fulfilment. For identical reasons, packet length likewise is additionally a helpful feature in identifying such applications. In flow-based methods, attributes of flows are extracted namely flow span, average size and packet quantity, etc. [69], [70] to detect applications. In [71], the technique of flow characteristics in a multilevel manner is adopted to distinguish flows associated with various classes. An adopted hybrid technique for an algorithm is suggested [72] to develop a classifier, utilized to group flows with attributes totalling to 17. In a comparative methodology [73], algorithms capitalized on machine learning are adopted to classify flows, extracting important features from packet headers. The use of a public dataset with the hypothesis that port numbers resident in the packet headers can classify application flows correctly which is not accurate in some cases comes as a limitation to this methodology.

Dorfinger et al. [74] utilized entropy of flows initial packets to determine whether or not the flow is encrypted. The initial packets refer to the payload excluding the packets inherent in the headers. The intended work was not targeted to detect the applications but to reveal if there is an existence of application flows encrypted. Alshammari and Zincir-Heywood [15] additionally utilized flow statistics conjoined with selected algorithms in literature to distinguish traffic. Their work yielded accuracy above 84%. Correlation information from flows is utilized by Zhang et al. [73] for their proposed work to categorize traffic into distinct applications. They defined the identified correlation as a bag consisting of flows (BoF). The flows are ranked with IP address, port, and protocol of similar destinations. The correlation information gained is utilized as a feature for the classifier in a probabilistic model. Evaluation results after the experimentation of traces from real networks resulted in accuracy from 60% to 85% for various applications.

2.4 The Unsupervised Strategy

Over the span of almost three decades, a quantified amount of works in literature have been suggested. A selection of clustering algorithms and methods that classifies network traffic effectively to our utmost knowledge is discussed. As far back as 1967, McQueen [75] proposed a non-hierarchical method of partitioning captioned as the K-means algorithm. Lloyd [76] adopted this method to partition datasets into clusters based on a predefined number of initially selected centroids (k). By this method, the centroid of the K number of clusters (C_k) is iteratively computed using the Euclidean distance, until a convergence measure is reached. The aim of this algorithm is to utilize the Euclidean

Distance to diminish the errors that occur in computing the mean squares from the objective function as given in equation (1) by:

$$q = \sum_{x=1}^{k} \sum_{x_i \in C_k} h^2 \qquad (1)$$

where h is the Euclidean distance and x represents objects in the data. The distance is h defined as the separation existing between two points a and b as shown in equation (2);

$$h = \sqrt{\sum_{i=1}^{m} (a_i - b_i)^2} \qquad (2)$$

where a_i and b_i are points within the Euclidean m space. The algorithm proved to be efficient with $o(jkn)$ computational complexity in that k represents the sum of clusters, j equals the total number of iterations, n equals to the total number of objects. Figure 4 illustrates how clusters are made from the k-means method. The first diagram, denoted as S1 represents objects in a given space of data to be grouped into two clusters. The value of K then becomes 2.

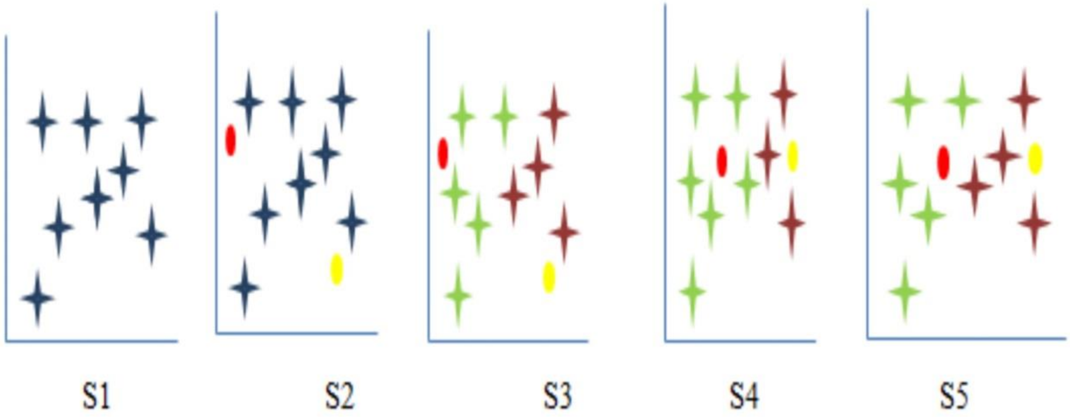


Figure 4: Illustration of K-means Clustering

Two centroids C_k are chosen at random represented by the red and yellow balls in S2. The Euclidean distance from all the objects to the centroids are computed and assigned to the nearest centroid in S3. The green objects and brown objects are assigned to the red ball and yellow ball respectively, showing objects assigned to its nearest centroid. The mean distance from the objects in their respective centroid in each cluster is computed to find the new position of the centroid as represented in S4. New assignment of objects to its nearest centroid is computed again as shown in S5. Repetition of the process is done until the centroid placements remain the same and do not shift to a different position upon further iterations. Quantity of Clusters and iterations demonstrated to be highly lesser than the number of objects. Moreover, the algorithm which terminates at local optimal produced closely related clusters, and is also computationally faster, as compared to the hierarchical method, which is characterized by high complexity of $O(n^2)$. However, the method possessed a high sensitivity to noise and outliers limiting its quality criterion. The efficacy of this method also relies on the strength of centroids initialization. A weak or strong initialization produces poor and good clusters respectively. Using the Silhouette tool can help to predict good initialization. K-means algorithm has served and still serves as the basis for different and several algorithms and its analysis [77] [78] [79] [80].

Hirvonen and Laulajainen [77] proposed a two-phase classification of traffic for Better Quality of Service (QoS) management incorporating the K-means clustering. The aim of their work is to provide an efficient classifier that is able to make out target applications and detect unknown flows (noise) in the network which could not be trained during the process. Classification is based on flow behavior and the process comes in phases, namely assignment phase and labelling phase with all using K-Means. The assignment phase basically assigns the flows to a cluster. The product of density measure and the phase threshold value determines the coverage of a particular cluster. The labelling phase uses the proposed algorithm to assign the appropriate label to the flows. A decision is then made based on the outcome of the above two phases serving as input to the classifier with some additional inputs. The proposed work resulted in classifying 97.8 % of target applications correctly after evaluation. Detection of untrained flows after evaluation resulted in 97.9% MSN flows and 100% Telnet flows. However, the calculation and determination of threshold values are not explained in details by the authors. Furthermore, the evaluation only compared its efficiency to pure port-based classification which has already been proved in literature as less efficient. Although the authors mentioned their method as a lightweight, the issue of the computational heaviness of the proposed work is not discussed.

Zhang et al. [81] came up with the BIRCH method which incorporated scalability into the clustering model. They used clustering feature tree (CF tree) with an in-memory structure and multilevel clustering to process large datasets in two main steps. Each step has an additional optional phase. Foremost, large datasets are compressed into a compact in-memory CF tree with the underlying clustering structure intact. By digesting into more suitable ranges, an optional smaller CF tree can be built. In the second step, they used an agglomerative algorithm with other flexible clustering methods to produce initial clusters, which were then refined based on their centroids (optional). Results from experimentation show that BIRCH aligns and scales linearly well when points in every cluster increase in number. In comparison to the K-Means, BIRCH performs better with respect to time on variant base workload. Incorporation of the agglomerative algorithm led to achieving $O(n^2)$ complexity. However, the BIRCH algorithm is found to be sensitive to the order in which data points are introduced. In addition, the generated clusters' natural appearance is very slim because of the static nature of the leaf nodes making the algorithm generate spherical clusters. For the purpose of clustering large datasets, many authors have formulated and developed other algorithms which serve as an improvement on the drawbacks of BIRCH or for comparative analysis [82] [83].

Guha et al. [84] used a hierarchical clustering algorithm, termed as Clustering using Representatives (CURE), to cluster larger dataset. They hypothesized that CURE can withstand distortions caused by outliers and that this approach was best suited for arbitrary-shaped and non-spherical shaped clusters with wide variances. CURE awards an enormous complexity in costs as $O(n^2 \log n)$ with respect to a higher dimensional space of input size and generates a complexity of $O(n^2)$ with a lower dimensional space of input size. Using this computation, the CURE approach marks scattered cluster data points as representative points of each cluster and introduces a shrinking factor to shrink these points towards the centroid. Comparatively, CURE produces higher quality clusters than BIRCH. When the number of points increases CURE results in better (lower) execution and running times (above 50%). CURE, however, is associated with higher data costs when it samples from a larger dataset.

Ester et al. [48] aimed at bringing out clusters shaped arbitrarily from their work called DBSCAN. The Density-based approach factored the quality of clusters that will be produced by considering the algorithms capacity to identify noise. With the origination of a density-based opinion of a cluster, parameters *Eps and MinPts* were defined. *Eps* reflects the density reachability possessed by clusters, and it characterizes the highest radius value of a point (P) neighbourhood. *MinPts*, on the other hand, refers to the density connectivity, which is the lowest value of points in number with an *Eps* neighbourhood. Commencing from an arbitrary point a, the clusters are formed by finding if, for any a the distance to the P is lower or equivalent to *Eps.* It includes every point in that neighbourhood to a cluster if the condition is true and becomes part of the neighbourhood of P. The process is performed iteratively to include new points. DBSCAN possess a sensitive characteristic to its parameter assignments which are not easy to compute or set. Performance evaluation in comparison to CLARANS [85] in terms of the accuracy of selected synthetic databases shows that DBSCAN is able to pinpoint all clusters and also detect the points depicted as noise. Complexity of $O(nlogn)$ with time is achieved, which is fair enough. However, CLARANS only divides large clusters or assign the noise data points to its closest cluster. In terms of run time, experimental results show that with increasing database size, DBSCAN performs better than CLARANS by a factor range of 250 to 1900, depicting much better runtime of DBSCAN.

Ankerst et al. [86] attempted using OPTICS to overcome inherent DBSCAN drawbacks. With DBSCAN, it is observed that the clusters produced from higher density require a smaller radius. It becomes a subcluster of those produced from lower density requiring a higher radius under the circumstance of the same *MinPts*. The proposed algorithm emerged as less sensitive to the parameter settings. In accordance with the structure of density-based methods, OPTICS generates a clustering order that stores information equating to a wide array of the parameter setting. Figure 6 shows how the algorithm scales well with varying values of Eps (ε) within a range of 10,000 to 100,000. This gives OPTICS the advantage of being linear and running very fast with respect to variable quantities of data points. The achieved complexity $O(n \log n)$ is fair enough, with *n* being equivalent to the totality of points.

Subramani et al. [87] adopted a hybrid of OPTICS and DBSCAN to tackle the issue of selecting an appropriate density threshold in social network community detection. Selection of a suitable density threshold contributes to the production of substantive clusters which was a limitation in SCAN [88] and DENGRAPH [89]. With density defined by a distance function, OPTICS usage enabled the authors to select a good *Eps* parameter distance value in DBSCAN and also to realize the outcomes of using alternating density threshold values. The study revealed that a community definition is liable to lead to sudden change and relies on the application assumptions used. The issue of whether a true definition for a community in social networks is feasible is an open-ended query that is derived from the analysis of the authors.

Hajikarami et al. [90] proposed a high-speed link two-layered lightweight system. Their work is an improvement of [91]. Their architecture suggested using k classifiers in the first phase of classification to reduce cost, resources, and memory compared to using only one classifier. Furthermore, the two-layered system made way for network administrators to make necessary adjustments based on their network requirements. In that when new flows whose signatures have not been captured in the existing knowledge base are introduced, there are labelled as unknown. The network administrator is then alerted to examine them and make the necessary changes to the classifier. After evaluation, the system's capacity to classify flows from applications correctly produced an accuracy of 99.5% and time of 41.28 seconds. The proposed architecture gives higher results but the time constraint for the network administrator to make adjustments to network requirements whenever there is a new trace which is not captured limits its efficiency. Moreover, the results showed that 60% of these new flows from applications were classified as unknown instead of being misclassified. This suggests the at least 40% of the flows can be misclassified before the network administrator makes the necessary changes which can really cost the network involved.

Research into IP and Traffic classification using unique flow characteristics also proved to be very efficient. McGregor et al. [33] went for the Expectation Maximization (EM) calculation to separate packet follows into groups of traffic, where each group has exceptional traffic qualities.

Zander et al. [92] suggested a novel computerized method of classification, an unsupervised method based on the statistical flow characteristics of NetMate [93]. Using the Expectation Maximization algorithm [33] and AutoClass Algorithm [94], the flow is first partitioned into bi-directional flows for the computation of flow characteristics. Intra-class homogeneity metric is then applied to maximize the overall homogeneity of the class. By this method, better separation of the different applications in a flow trace is achieved. The authors' approach resulted in an average accuracy of 86.5 percent in classification with some individual applications achieving a median close to 95% as depicted in Figure 8. The issue of computational complexity, as well as performance on larger datasets, has not been addressed by the authors as done by previously proposed methods.

In [95], Bernaille et al. came up with a stream arrangement approach dependent on the magnitude payload of the initial of packets. Spectral clustering was adopted in the process. A value of 80% precision was attained for varying applications under study. Similarly, in [96], the authors targeted performance of ML algorithms by utilizing the initial couple of packets in a flow. Despite the fact that this method is viewed as quicker and less tedious compared to the use of full flows, the situation of losing the underlying packets causes the capacity of this classifier to be progressively worse.

Crotti et al. [13] came up with a unique fingerprint protocol strategy. Based on normalized thresholds of the algorithm, network traffic flows are classified. The suggested works depended on collective attributes of gathered packets which are the order of packets arriving, gap interval in time between arriving packets and packet dimension. Utilizing initial packets as [97], their investigation results accomplished immense accuracy for distinguishing a variety of applications types. In any case, adequacy of the strategy worsens if the orientation of the client and server is not known by the classifier, the start of the flow is omitted, loss of the principal or initial packet, or if reordering of a packet is excluded.

Undeniably, a great number of researchers probed the viability pertaining to ML algorithms via the training and testing of classifiers with complete flows [9] [33] [10] [91] [96] [70]. For that reason, the ability to single out the sort of application before the end of flow contributes to the avoidance of losses in enterprises in any occurrence of security incidents. A couple of research considers (for example [12] [98]) assessed ML strategies utilizing a sub-flows. The researchers in [12] proposed a technique of classification dependent on sub-flows as opposed to depending on classification using the extraction of flow features. They used the Naive Bayes algorithm utilizing a little quantity of the most current packets that were obtained from full flows for the training of the classifier. They demonstrated that they had the capacity to limit the amount of space required for buffering in the process of classification. Besides, this strategy stayed away from the classifier's pursue to obtain the beginning of flows as done in the previous studies ([95] and [24]), which has the propensity to be missed and thus influence effectiveness of the classifier; thereby showing poor execution of classification based on complete and full flows in certain situations. Further investigations were carried out in [97], utilizing the datasets the former researchers used for their work.

The creators in [99] consolidated two algorithms and exploit their resources in their methodology. The advantages of classifying flows based on previous experience and detection of new applications without supervision were adopted from supervised and unsupervised techniques respectively. The proposed strategy outperforms classification that utilizes supervised methods only. This procedure lessens the time expected to prepare the classifier by utilizing labelled flows in limited measures. It can deal with debasement of supervised methods in their dealings with labelled datasets by applying clustering techniques to improve the performance of the classifier.

2.5 The Semi-supervised Strategy

The semi-supervised strategy can retain great outcomes with limited labelled examples and various unlabelled examples. Semi-Supervised Techniques have also led to a new dimension of research in clustering approaches. So as to address issues pertinent issues in supervised and unsupervised learning, few semi-supervised strategies have been considered over the previous years, which includes graph-based methods, cluster and label, and co-training [100]. The Co-training technique as recommended by Blum and Mitchell [101] serves as a common semi-supervised art. The instructions for the algorithm presume two features which are free and sufficiently able to prepare a proficient classifier. Initially, the labelled data set is trained with two classifiers using the two features separately. At that point, every classifier characterizes the unlabelled information and instructs the other classifier with the small number of unlabelled examples they are generally certain of. Every classifier is trained again with the extra examples of the training data given by the other classifier, and the procedure is performed again. The least difficult procedure for semi-supervised strategy is self-training. When there exist significant quantity reduction of features to one or a single accustomed classifier, the process of co-training is changed into a self-training operation. With respect to self-training, the inherent classifier is foremost trained to utilize a little measure of labelled examples available. Thereafter, the classifier categorizes the data points not assigned to any label. The classifier is trained for another time and the technique rehashed. The prevalent issue concerning self-training stems from the fact that classification procedure frequently experiences miscalculation aggregation and the algorithm used is not sturdy to anomalies or outliers [100].

Erman et al. [102] proposed one of the earliest semi-supervised works in clustering using supervised and unsupervised methods. Packet Milestones were used as a design consideration. The authors researched into classifying traffic using flow-based characteristics in applications and proposes a semi-supervised method for classifying traffic from known and unknown applications. The classifier is trained with flows comprising of few labelled and many unlabelled flows. The problem of class imbalance (high accuracy in flow and low accuracy in byte) [103] is addressed with the detection and good representation of both the mice and elephant flows as well as the effect of sampling methodologies on the selection. Higher accuracy of flow and bytes is achieved from the evaluation results with over 90% accuracy using the proposed semi-supervised procedure with real-time traces collected in a span of 24weeks.

To revamp the accuracy of the clustering method of classification, Wang et al. [104] suggested a semi-supervised strategy called set based constrained K means. The statistical features of flows are extracted along with some background information of the TCP/IP flows. Using Gaussian mixture density, the observed data and derived constraints are modeled. The authors establish that the introduction of feature discretization in flow clustering can increase the level of clustering accuracy. Based on only how the flow features are similar or dissimilar, they are grouped according to the tuple labels. Flows which bear similarities from different applications are likely to be grouped into a particular cluster. The background information incorporated is that any traffic flow having their two destination tuples and protocol in common is said to belong to the same application. The authors used a Gaussian mixture model to formulate their constrained clustering problem. To generate the maximum likelihood of the model parameter of the given dataset, a log likelihood of the formula must be taken to make the computations easier. But using the log likelihood directly is expensive or hard. Therefore an

Expectation maximization approach must be incorporated. The authors proposed what they call an Approximate Method. The distinguishing element of this accession in comparison to the traditional K-means is exhibited in its scheme of assignment. K-means processes, each data point independently. On the other hand, SBCK considers equivalent samples as one whole based on the extra set based constraints. Comparing SBCK to K-means and EM without feature discretization, SBCK provides better accuracy after clustering with a percentage range of 85 to 91 percent and with feature discretization provides accuracy of 94 to 97 percent accuracy. Also, run time for SBCK is better than K-means when clustering large datasets, but K-means performs better than SBCK when clustering small datasets.

To overcome the limitation of labelled examples and wearisome manual work associated with supervised learning, Shukla et al. [105] proposed a semi-supervised clustering method classifying traffic using flow statistics and K-Medoids algorithm. The three-phase procedure constitutes data preprocessing and transformation, clustering with K-Medoids and assignment of class labels. The probabilistic assignment technique after evaluation again proves that accuracy values accelerate with increasing number of clusters. An accuracy of 92% is achieved with a cluster initialization of 30 which in the real world is delusive and increases the cost and overall time of the classifier.

To address the poor traffic conducts with meager labelled examples, Zhang et al. [106] integrated flow correlation methodologies in all phases of classification. For the fundamental training, flow correlation boosts labelled data by first using the pre-labelled data set to label the unlabelled ones using their correlation to the former. Subsequently, the traffic classifier possesses prominent execution by virtue of the all-encompassing size stature and nature of administered informational indexes. The correlated flows during the testing stage are recognized and arranged together by consolidating their individual expectations, that it may additionally upgrade the precision and exact certainty of classification. The exactness of the proposed technique is higher than KNN [107] and Erman [102] strategies over 20% on account of 10 labelled test sets for each class. The outcomes demonstrate that using correlation stream can successfully address the minimized sample issue.

Yan et al. [108] adopted a co-training method to classify online traffic in order to overcome the disadvantage of rare labelled samples in supervised learning. The recommended strategy is a semi-supervised technique, which utilizes minimal marked examples and a lot of unlabelled examples to improve the execution of the supervised learning strategy. The algorithm for co-training demands two distinct features which are enough for a proficient classifier. The authors choose the dimensions of the packet and the gap in packet arrival of the initial flows. Since the time of arrival is reliant on the network working conditions and influenced by jitter, the authors implement a jitter flexible feature called Netipt and merge this feature with the co-training algorithm. Evaluation results after experimentation reveal that the proposed method performs better with an overall accuracy of 97.36% compared to J48 algorithm with 85.83% and self-training with 95.82%. All algorithms were integrated with the Netipt feature which further shows how the introduced feature helps to boost and more suitable for classification compared to other features.

2.7 Classification for Specific Targets

The technique of traffic classification has also been used to classify and identify specific kinds of traffic such as Internet video traffic, P2P traffic, Voice over IP traffic (VoIP), gaming traffic, analytical purposes and also to enhance the quality of service. These approaches are very relevant to users or organizations requiring very specific types of traffic usage, and anomaly detection on their networks.

2.7.1 Video Traffic

To give a complete Quality of Service (QoS) assurance for services pertaining to video streamed on the Internet, verified international standard organizations [109] have established distinct granularity for certain QoS service classes. Nevertheless, the granularity of classes may not be appropriate for the classification of every kind of video traffic with the surfacing of contemporary ones. Based on the resolution of the video or the amount of bandwidth required for its transmission, for instance, video traffic can be classified. With the end goal of increasingly proficient resource usage by networks, better granularity is required when video traffic on the Internet is to be classified. In the interim, the rapid growth of new applications make the conditions in the network progressively intricate and causes a progression of issues to emerge, for example, managing of resources in the network and guaranteeing QoS for multimedia systems. It is trusted that classifying network traffic precisely is a compelling and efficient way for ISPs and administrators of networks to handle these issues [110]. This will enable enterprises offering internet services to ensure exceptional guarantees for QoS, video traffic and services which possess constrained QoS resources in the entire network.

Concerns of analyzing video traffic that traverse the internet effectively with guaranteed Quality of Service is addressed by Zai-jian et al [111]. With one target application (Video Traffic) in focus, the authors use the QoS based Flow Aggregation framework to present a modified K-Singular Value Decomposition (KSVD) framework. Five QoS features are defined based on the upstream and downstream rates. These rates can be applicable when selecting features for QoS based classification because they depend on the bandwidth resources. Video traffic is grouped into five classes namely, Web video, trade style video, barter style video, and interactive video. Their QoS requirements are also grouped on a scale of 1 to 5, with 5 representing higher requirements of QoS. A bag-QoS-word dictionary formation rests on the model. Modified K-SVD is thereafter used to train the Bag-of-Words and the video traffic classified using a linear Support Vector Machine (SVM) classier. The aftermath of the experiments exposes that the suggested mechanism achieves 98.98% accuracy which is better compared to Naïve Bayes of 88.79% and Hidden Markov’s Model of 89.87%.

2.7.2 Peer-to-Peer Traffic (P2P)

Several applications reliant on P2P conventions have turned out to be immensely prominent, presently representing a noteworthy offer of the absolute system traffic. To evade limitations forced by system overseers for different reasons, the P2P conventions have turned out to be increasingly complex and utilize different strategies to stay away from identification and acknowledgment with standard estimation devices. The principal strategies to identify P2P traffic were port-based [112] [113] [114] [115]. These systems adopt port numbers at the fourth layer, data within headers, with a rundown of familiar ports to recognize corresponding P2P packets. With the subject of port-based discovery, it is highly compelling for P2P applications with constant port assignment. Presently the vast majority of the P2P applications adapts to ports changing. Classification of P2P stream is significant for managing networks, quality of service, traffic analysis, etc. since P2P applications embody a greater portion of the internet's contemporary traffic. ML strategies have pulled in wide consideration due to its high precision values in classification, and the ability to characterize obscure P2P traffic. Existing ML methods, for the most part, utilize the time characters of the domain where the flows are derived for P2P traffic classification.

Bin and Hao [116] concentrated on detecting particular application traffic that is P2P traffic for their work. P2P metrics coupled with preferred semi-supervised technique of clustering referred to as Particle Swarm Optimization (PSO). To resolve the puzzle of obtaining labelled samples which are scarce and also the problem of new samples whose labels are not known beforehand, the authors proposed using a two-step semi-supervised method. The first step PSO is used to prepare the dataset by partitioning the scarce labelled dataset mixed with an enormous amount of unlabelled dataset. The second step constitutes using the labelled examples to acquire a direct linkage from clusters to their prospective classes respectively. PSO algorithm augments the inter-cluster distance while decreasing intra-cluster distance by discovering appropriate centroids. The algorithm outputs a set of clusters with centroids. A probabilistic mapping function is used to assign other applications to their respective clusters for the final classification. Clustering evaluation depicts that the proposed method achieved more than 85% accuracy with labelled samples as many as 100 or more. The proposed work also achieved as high as 95% accuracy of classification results with higher numbers of cluster initialization. The authors, however, assert the fact that a cluster initialization of 5000 is not realistic and inflates classification costs.

With a similar target of also identifying P2P traffic with clustering technology, Tseng et al. [117] based on the traffic flow aggregates similar unfamiliar flows conjoined with labelled cases into related clusters. Categories of unknown traffic are classified based on the cluster they belong to the labelled ones. From [5], when the batch of packets within flows exceeds 64, it signifies there are enough features to classify the flow. The authors sift all flows beneath 64 threshold number and obtain the features of packet size which includes minimum and maximum packet in a flow, the average and standard deviation. They further use aggregation clustering technology, where the number of clusters is generated automatically as per the network flow. Based on the concept of correlation flow in [73], if a set of flows has the same 3 tuples, the flow is said to be from the same application. The correlation of traffic flows is evaluated to see if the traffic is coming from the same application or not. A semi-supervised classification is proposed where the labelled and unknown traffic is mixed. Based on similarities of the flows, the unknown traffic is grouped under the same cluster of the labelled traffic. It groups the unfamiliar traffic with labelled exemplars after aggregation. Similar flows are merged into the same cluster. Results after assessment show that the suggested mechanism attained the maximal accuracy of 90% compared to Single-Linkage of approximately 65% and K-means of 79% giving the same initial number of clusters for all three methods. Again the complexity of this method is not discussed, though results show it to be more efficient compared to the other methods.

Using flow characteristics for classification is subject to change in different networking environments causing these features to be unstable. To overcome this drawback and to increase machine learning classification stability, Du and Ou [118] proposed to use time and wavelet transform based frequency domain framework to identify and classify P2P traffic. The control packet information exchanged between peers concerning a data block before and after transmission is very stable. It does not change or differ irrespective of the network environment, P2P protocol or implementation used. Another characteristic of P2P is its unique periodic transmission, which differs from other applications. Discrete Wavelet transform (DWT) can break a time series in several portions with each portion containing significant information about the pilot time series. The authors utilize the above mentioned characteristics together with information gained from using DWT to represent a P2P flow and classify the packets. Experimental results compare the accuracy of classifiers trained by time domain characters only and classifiers trained by a mixture of time and frequency domain characters. Classifiers trained with only time domain characteristics results show varying values for flow and byte accuracy as being indirectly proportional. Where the flow accuracy is high (88% - 99%), byte accuracy is very low (28.13%). On the other hand, classifiers trained with mixed features showed stable flow accuracy values (99.59%, 99.34%, and 99.04%) and byte accuracy values (99.62%, 97.18%, and 98.10%).

2.7.3 Game Traffic

To analyze and identify game traffic from different game applications, Han and Park [119] employed its statistical features and characteristics to ensure a productive classification. Based on the simple decision tree method of classification, the authors suggested a new method by name Alternative Decision Tree which incorporates these statistical features of game applications. Game traffic has few features which are not enough to effectively classify them. The authors investigated other statistical characteristics at the transport layer. They uncovered that, without using intricate methods, packets sizes which are frequently used can be used to classify some of the applications. In addition, packet size distributions amid the peculiar applications were common. Utilizing the above mentioned features discovered, the authors proposed the

ADT method of classification. ADT contains two distinct phases. The prime phase consists of an examination of statistical data and pre-classification of the flows. IP addresses for both clients and servers are also gathered. In the second phase, grouping of the flows rests on the pair of port numbers together with IP addresses. Relevant groups obtained are then classified again. The proposed method is relieved of forensic issues since user data is not involved, and limited features are used, therefore reducing the complexity. After experimentation, the results were evaluated in terms of precision and recall and compared with results from using port-based classification only and classification using correlation from packet size distribution only. The combination of port-based and correlation (ADT) produced better results compared to using each method separately.

2.7.4 Voice over IP Traffic (VoIP)

Do and Branch [120] also adopted the use of machine learning classification to identify VoIP, Skype in the presence of other traffic on the internet (gamming, etc.). Instead of using the whole flow of traffic, a short sliding window is selected. Cisco IP phones are used to generate the VoIP traffic. Also, Skype traffic which transcends under the proprietaries for VoIP calls over the internet is captured, when a client makes a call from one computer or phone to another over the Skype application. The authors aimed to prove that the significant features for classification are packet dimensions and the gap evident in their arrival. C4.8 decision tree mechanism is adopted to select the important features dominant in the dataset and then partitioned into segments of 1 to 10 seconds of sliding windows while increasing the sliding window by a second each time. The datasets are then split into two folds for the training and testing. Using the J48 decision tree classifier the testing data is classified using a classifier trained with statistics in relation to packet measurements only, lag in arrival time of trailing packets only, and a mixture of both features. Experimental results show that with packet length statistics only, one second window has values 96%, 100%, and 92% while 10 second window has 98%, 100 and 97% for VoIP, Skype, and Other traffic respectively. For the classifier trained with inter-arrival statistics only, 1-second window has values 89%, 93%, and 92.5% while 10-second window has 97%, 99 and 99.1% for VoIP, Skype, and Other traffic respectively. The classifier trained with both features showed that for 1-second window, values 95%, 99%, and 93% are achieved while 10 second window has 99%, 100 and 99.3% for VoIP, Skype, and other traffic respectively. Results for the 10-second widow indicate that using both features to train the classifier is very effective.

2.7.5 Classification for Analytic Purpose

Relying on an OMNETT++ prototype classifier, Achunala et al. [121] introduced an effortless packet classification. The authors aimed at using a clustering technique to build an effective, efficient and accurate classifier. Their research establishes a novel method to classify packets, excluding payload data or information. An Inter-Arrival Precision (IATP) clustering algorithm is proposed. The proposed model consists of applying clustering methodologies to generate independent clusters with training data and then classifying the obtained clusters further into smaller cluster subsets which are then labelled. The clusters are automatically labelled or grouped under five distinguish class ID's (Class ID 0 – Class ID 4). Simulation results show 100% accuracy of results. The authors assert the fact that the results obtained do not represent real-time classification which falls within an accuracy of 85 to 95%. They agree on the fact that much study and heuristics are needed for real-time traffic quantifications.

Identification of Traffic proof methodologies that depend on heuristics got from examining the patterns in host communication have likewise been proposed [26] [71] [122]. For instance, Karagiannis et al. [71] built up a technique that uses social, practical, and hosts habitudes in applications to recognize classes of traffic. Simultaneously, Xu et al. [27] built up an approach, in light of data theoretic methods and data mining, to find practical and application standards of hosts' habits and resources utilized by them. They accordingly employ these cases to construct a generic profiling for traffic.

2.7.6 Classification to Enhance Quality of Service

Wang et al [122], classify clusters based on Quality of Service requirements other than applications of the traffic in a software defined networks (SDN) along with the implementation of Deep Packet Inspection. Incoming flows possessing long lives are detected with an SDN switch. With values of Hurst packet, port and average packet inter-arrival time as inputs into a mapping function, traffic flows are classified into their respective QoS classes. Every network has a specific purpose; therefore the traffic generated from every network will not be the same. Because new applications are developed each day or even existing ones are getting updated, their statistical properties may also change. So there must be a way to retrain the QoS classifier to update these changes in the existing database already gathered after some duration. The QoS classifier therefore employs a semi-supervised ML algorithm called Laplacian Support Vector Machine (LapSVM) inside the centralized SDN controller to achieve a coarse-grained classification. The authors apply a clustering assumption that statistical properties are similar in applications having identical QoS requirements. Evaluation of the classifier with respect to classifying accurately indicates that the proposed classifier is more efficient than the existing K-means based classifier proposed in [101]. With an accuracy exceeding 90% is an indication of a proficient classifier. However, the effects of packet loss on the classifier if any are not addressed by the authors.

For the purpose of Quality of Service using a generative model (Hidden Markov’s model, HMM) for semi-supervised sequence learning, Dianotti et al. [123] recommended a unique packet-level manner of traffic classification. Usage of this HMM sequence qualifies this approach to be in line with semi-supervision. Using the characteristics of packet dimensions and packet arrival timings, the authors performed classification applying these characteristics in an aggregated fashion using real network traffic and estimation, making it usable on encrypted traffic as well. From experimental results, their model classifies more than 90% applications correctly, which signifies a higher efficacy rate of being a multi classifier system. Performance measure under the increasing rate of traffic data streams, as well as its capability to handle outliers, is not included by the authors in their experimental results. Moreover, the authors admit that the success rate of their model is considered under the assumption of traffic flow in one direction of the application and not in both directions. They perceive the adaptation of the latter into the model to produce better accuracy.

Packet loss induced by attenuation, shadowing, etc. in networks differs from packet loss as a result of congestion in networks. Periodic packet loss by virtue of network congestion can result in extreme abasement of network performance. To enhance QoS, increase throughput and reduce the congestion, packet loss classification has been suggested in literature. Hsiao et al. [124] used the detection trends in relative one-way time (ROTT) in situations where the packet classification is not directly straight forward and descends in ambivalent zones. The author’s proposition is that, since delay is a summation of delays in electromagnetic wave propagation from one end to another, queuing delays, delays in router processing, information from packet delay can be inferred from this. Hence, the occurrence of a packet loss in an estimated time is regarded as congestion delay so far as the delay propagates in escalating order, else it should be considered as a wireless loss. Within the interval of the ROTT gray zone, the classification of packet loss is such that when the received ROTT is greater than the upper bound gray zone it is referred to as congestion loss, other than that, it is categorized as wireless loss. Evaluation results of the classification algorithm depict high throughput values compared with Spike-train and ZigZag for distinguishing packet loss. For the purpose of network security, a multi classifier is more desirable by many corporations and organizations.

2.8 Comparison of Traffic Classification Methods

It is perceptible that the vast majority of the exploration so far has not assessed the efficacy of the classifier as far as fragmentation, delay, and packet loss are concerned. Besides, albeit unsupervised systems can distinguish the presence of another application in networks, most of the works have not examined and considered this issue, despite the fact that it is referenced in [99]. Out of the works discussed above, the most productive ones with best performance results are analyzed and compared in Table 1 for unsupervised and Table 2 for semi-supervised.

Table 1: Analysis of Efficient Unsupervised Clustering methods

Author	**Objectives**	**Clustering Method**	**Clustering Parameters**	**Limitations**	**Results**
Lloyd [76] K -Means	To diminish the errors that occur in computing the mean squares in cluster formation	Classic K-means	Distance function as a parameter setting	* Sensitive to noisy data * poor clustering resulting from poor initialization of centroids	Produces closely related clusters compared to the traditional hierarchical methods
Zhang *et al*. [81] BIRCH	To use a limited amount of resources to process large datasets	Hierarchical (agglomerative algorithm)	* Clustering feature tree (CF tree) * Multilevel approach of clustering	Sensitivity to insertion of data points	* Handle outliers (noise), * Higher workload base performance * **Time** : clusters large datasets in less than 15 seconds (within 10-14 seconds) to K means (minimum within 12 – 44 seconds range), CLARANS (Minimum of 816 seconds)
Ester *et al*. [48] DBSCAN	To better the quality of clusters using the algorithm's capability to identify noise	Density Based Clustering	Density reachability *(Eps)*, Maximum radius of neighborhood *(MinPts)*	* Sensitivity to parameter settings *(Eps* and *MinPts).* * Difficulty in computing parameters	* **Accuracy:** Able to identify and detect noise points while CLARANS assigns to nearest cluster * **Run time:** with increasing database size, DBSCAN performs better than CLARANS by a factor range of 250 to 1900. * Complexity of time which is fair enough
Guha *et al*. [16] CURE	To Identify non-spherical shaped clusters, arbitrary shaped clusters and withstand outliers in large datasets	Hierarchical	* Representative points for clusters * Shrinking factor	High computational complexity (cost) with higher dimensional space of input size (from large datasets)	* Produces high quality clusters. * **Time:** 50% lower execution time compared to BIRCH with increasing number of points.
Ankerst *et al*. [86]OPTICS	To overcome the limitations of DBSCAN's sensitivity to its parameters	Density Based	* Density reachability *(Eps)*, Maximum radius of Neighbourhood (*MinPts)* * Augmented Clustering ordering / structure	Challenge of managing the clustering order with increasing updates of the database taking place	* Reachability plot is insensitive to input parameters when compared to DBSCAN and other clustering methods * **Run Time:** Fairly same as DBSCAN with its parameter setting, but lower other parameter settings such as tree based special index or using grid objects
Subramani *et al*. [87]	To select an appropriate density threshold in social network community detection	Hybrid Approach (OPTICS & DBSCAN)	Density threshold parameter	Computational complexity of the hybrid approach not discussed	* Community definition is liable to lead to sudden change and relies on the application assumptions used. * Hybrid approach gives clear understanding into clustering structure * Ease of density threshold selection using the proposed method.

Zander *et al.* [92]	* To improve the overall intra class homogeneity * To overcome traditional methods of classification limitations.	Probabilistic Clustering Approach (Expectation Maximization and mixture models (AutoClass)	* Statistical flow characteristics * Intra class Homogeneity as a metric.	Performance on increasing datasets and runtime complexity not considered	Achieves an average 85% accuracy of clustering the flows with some applications achieving as high as close to 95%
Hirvonen and Laulajainen [77]	To provide an efficient classifier that is able to identify target applications and classify network flows in applications that are untrained as unknown.	Classic K – Means	* Flow behaviours * density measure * phase threshold value	* Calculation and determination of threshold values not discussed. * The evaluation compared its efficiency to only pure port based classification and not to other renowned existing works * Computational heaviness of the proposed work is not discussed	* Classifies 97.8 % of target applications * **precision**: detection of untrained flows from applications

Table 2: Analysis of Efficient Semi-supervised Clustering Methods

Author	**Objectives**	**Clustering Method**	**Clustering Parameters**	**Limitations**	**Results**
Erman et al. [102]	To build a fast and accurate classifier that accustoms to known and unfamiliar applications.	Classic K - means	Distance function, Flow characteristics, packet Milestones	Do not compare results and performance with other works or classifiers	A high flow and byte accuracy is achieved with over 90% accuracy.
Wang et al. [104] SBCK	To improve upon the accuracy of clustering method of classification	**Hybrid**: Probabilistic Hierarchical (K - means with Gaussian Mixture Model)	Flow Statistical Features, Feature Discretization, Log Likelihood,	K means outperforms SBCK for small datasets in terms of run time SBCK – 0.4 seconds, K-means – 0.2 seconds.	* **Accuracy**: SBCK – 94 to 97 percent, K-means – 73 to 81 percent, EM – 90 to 93 percent (at higher levels of K =500) * **Feature Discretization:** SBCK – 96 to 99 percent accuracy * **Run time**: SBCK – 5 seconds, K-Means – 13 seconds for large datasets
Dianotti et al. [123]	To develop a multiclassifier for higher accuracy to achieve a better Quality of Service	Hybrid (Hidden Markov's Model with Packet features)	Packet size, inter packet time,	Do not compare its performance with other classifiers	classifies more than 90% applications correctly
Wang et al. [122]	To realize an accurate traffic classification for Improved Quality of service	Hybrid (Machine learning & Deep Packet Inspection)	QoS requirements, average packet inter arrival time, Hurst parameter, length of packet	Issue of packet loss not addressed.	Test accuracy exceeds 90% which performs better than the existing K-means method in [102]

2.9 Evaluation and Discussion of Results

From Table I, it can be deduced that all proposed works achieved some level of accuracy ranging from 80% to above 90% indicating that clustering techniques are better for network traffic classification. Also, supervised and semi-supervised methods that incorporated the K-means, either as an aggregation or adopting its advantages, achieved higher percentages of accuracy compared to the others. From Table 1, Hirvonen and Laulajainen [77] used the classic K-means in an unsupervised technique and resulted in classifying 97.8% of target applications. Similarly, from Table 2, Erman et al. [102] using the Classic K-means in a semi-supervised technique achieved an over 90% accuracy in classifying flows. In addition, Wang et al. [122] semi-supervised SBCK, which has the Classic K-means and Gaussian mixture model (hybrid approach), resulted in 96% to 99% accuracy with feature discretization. They also obtained an accuracy of 94% to 97% without feature discretization. SBCK also had better run time of 5 seconds compared to K-means of 13 seconds. The above methods with Classic K-means yielded better results compared to Zander et al.'s [92] probabilistic clustering approach, which obtained accuracy between 85% to 95%. Although Lloyd's [76] approach is one of the earliest to produce closely related clusters, its high sensitivity to noise remains a challenge. The Hierarchical agglomerative method used by Zhang et al. [81] overcame this drawback. The Hierarchical and density-based methods adopted by some authors considered the run time of the proposed algorithms. In terms of run time, Ester et al.'s [48] DBCAN performs better than an existing density-based algorithm CLARANS by a factor range of 250 to 1900. In spite of the similarity of Ankrest et al.'s [83] OPTICS to DBSCAN in run time, it could achieve a lower complexity of *O(n)* using grid objects. The hybrid approach of the above methods in Subramani et al. [87], from Table 1, is able to define and give a clearer understanding of the clustering structure. However, its runtime complexity is not discussed by the authors. The most interesting derivation is that methods that aimed to improve the quality of service also achieved better results with accuracy above 90%. In Table 2, that the approaches used by Dianotti et al. [123] and Wang et al. [122] achieved accuracy greater than 90%, which makes their methods more effective than the K-means approach adopted by Erman et al [102]. Thus, incorporating quality of service features into the K-means method is more likely to produce higher percentages of accuracy.

2.10 Challenges in Traffic Classification

Even though the clustering technique of network traffic classification has yielded higher results in terms of accuracy and performance, some challenges still persist. The method of clustering itself has a challenge of how to produce good and non-overlapping clusters. The definition of a good cluster depends on the purpose for which the clustering is to be used or what it seeks to achieve. Another challenge is how to reduce the error rate. Roughan et al. [9] investigated the origin of this problem using statistical signatures of the flows, utilizing algorithms from machine learning and Nearest Neighbours for the purpose of Quality of Service. Their evaluation resulted that flows consisting of different applications are more prone to errors. As the number of mixed applications increases, the error rate also increases. The challenge of a better clustering technique with low computational complexity is another challenge in Network traffic classification. To the best of our knowledge, there is no proposed work that has achieved a lower computational complexity than K- means and overcoming the drawbacks of K-means at the same time. This poses the problem of one technique being able to factor all challenges and overcome all drawbacks. However, depending on the requirements of a particular network, the importance of what a technique can be prioritized.

2.11 Summary

Existing algorithms and methods which have had the greatest impact on clustering traffic flows have been discussed in the paper. These algorithms and most of the existing work is focused on features like packet size, inter-arrival time, including some QoS features as well [125]. However, their over-concentration on particular applications that are traversing through the network limits their capacity to classify service classes efficaciously for better QoS. In [6], the authors aimed at overcoming this limitation, but only focused on internet video traffic without adequate attention to other types of traffic that can traverse through the same network. We therefore recommend further research into Quality of Service approaches to clustering. QoS levels provided by networks form an important aspect to many networks and service providers, therefore developing a more effective algorithm that uses some QoS parameters like throughput, packet loss, packet fragmentation, and delay will be of great value. Researchers have a keen interest in developing more accurate methods of classifying and identifying real-time traffic patterns in network security and other network solutions. A lot of models have been formulated based on the existing unsupervised and semi-supervised methods of clustering. These models comprise techniques, which demonstrate the algorithm's capability to handle noise and its performance and ability to classify a large dataset of real-time network traffic. Although classic K-means approach has served as a relevant model for the development of several semi-supervised clustering approaches, related computational complexity impedes its ability to work with limited computational resources. However, to our utmost knowledge, there is limited research on how the algorithms will perform under certain QoS parameters are incorporated, which is our aim to investigate in the study.

Chapter 3: Design and Implementation of Proposed Scenario Topology

3.1 Motivation of the Design

The proposed topology's motivation discerns from real-time wide area networks with limited resources for data transmission such as low-speed links. Most datasets used for testing classification algorithms are collected in network environments having adequate network resources to transmit data from source to destination networks.

However, the performance of these algorithms on datasets generated from deprived networks has not been investigated. A topology to represent such scenarios is designed and implemented in OMNET ++ (Object Modular Network Testbed in C++) simulation to generate a dataset for the study.

3.2 OMNET ++ Simulation

OMNET ++ simulation software is a component-based object-oriented program with a modular open-architecture. It works as discrete-event network simulator for wired networks and distributed systems which include computer networks. It is gratis for studious research, unremunerated purposes and compatible with both Windows and Linux platforms, making it flexible to use. Graphical user interface (GUI) and parallel execution are supported with various add-on libraries which are imbedded and easily accessible. The main components include:

1. The language topology (.ned files) defining the structure of elements and parameters
2. message definitions which is used to set several messages and supply the data sections required to be transferred to finely developed classes of C++

3. Sources from which simple modules are derived, including the files coded with C++ and identified with .h or .cc extensions. The sources for the constructed modules constitute all instructions and codes for the implementation of the modules to be used.

Existing modules can be imported and edited to meet requirements or new modules can be created. OMNET ++ is installed on Ubuntu Linux 16.04 and integrated with INET framework 3.6.5 for the implementation of the network topology. The INET Framework is a module library for simulation environment of OMNET. It is open-source and provides various protocols, agents, internet stack and other models for research purposes in the field of network communications.

3.3 Proposed Scenario Implementation

We assume a wide area network scenario where there is a lot of congestion leading to packet loss subjected to a lot of fragmented packets. Network topology is implemented in OMNET ++ simulation software as illustrated in Figure 5. The network topology consists of two autonomous networks connected by routers. Different types of application messages or packets are generated at random and exchanged between the two networks. The application messages include various audio and sound formats, video, https (websites or internet), FTP and VoIP. The parameters subjected to the topology are shown in Table 3. The simulation is allowed to run for a time period of 150 hours. Data is exchanged between the clients and the data generated is sent over the border routers using the Border Gateway Protocol (BGP). For the actual transmission of messages, Transport Control Protocol (TCP) or User Datagram Protocol (UDP) is preferred and collected on the server. To begin with, when application data is generated, a session is established from the Transport layer using either TCP or UDP from the clients to the first hop (router). The Internet Protocol (IP) address of the router is fetched and the routing table is updated as shown in Figure 6. On the router, the best matching route for the packet transmission is sought. Internet Control Message Protocol (ICMP) is adopted to handle errors which may occur at this layer, and Address Resolution Protocol (ARP) aligns the virtual address to the physical client address as displayed in Figure 7. Data logs throughout the entire transmission are collected on the server are saved in .vec (vector) file format by OMNET. The log file contains records of the data values in relation to time. It includes the inter-arrival time of packets, packet types, the time covered in hops for a packet from a node to another node, hop counts, time to reach destination nodes, mean, standard deviation and all other records of time events with respect to the requirements of the network. The dataset is saved in .csv file format and exported to MATLAB for clustering and classification.

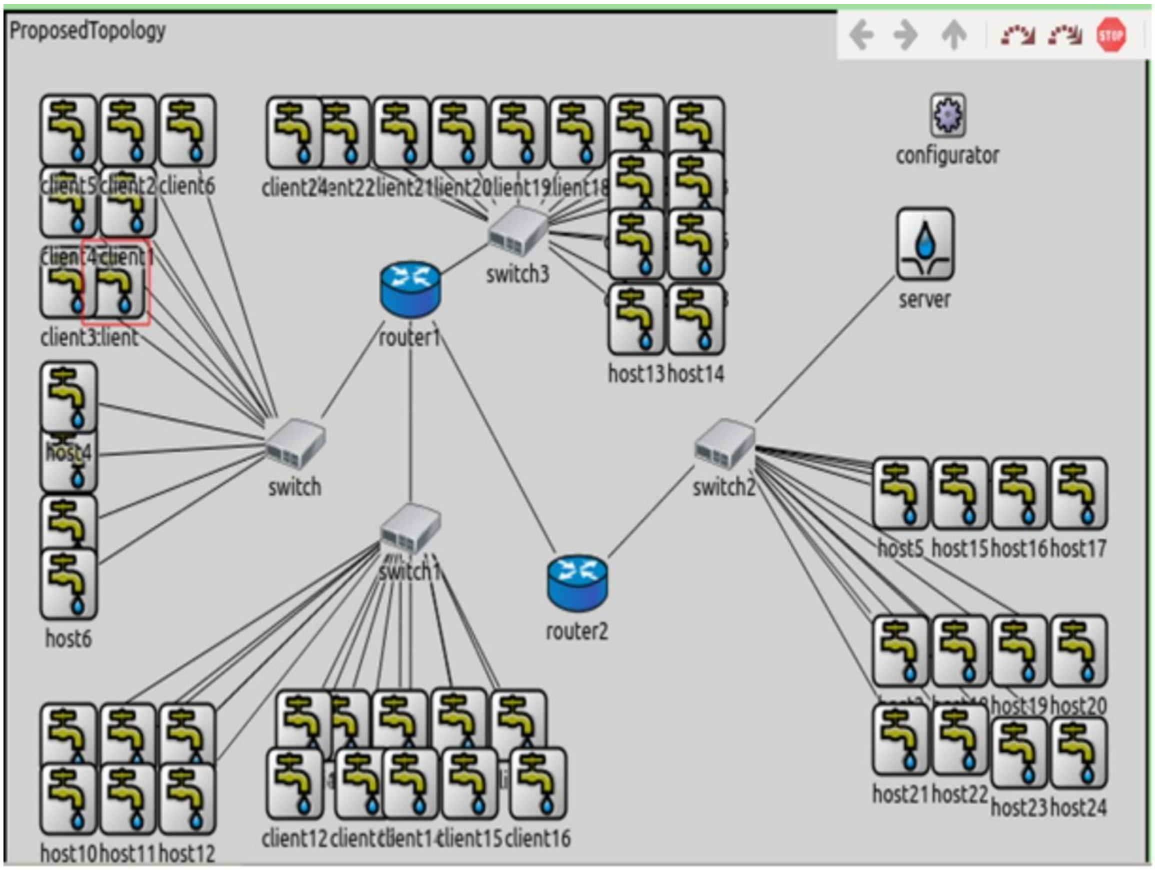


Figure 5: Proposed Network Topology

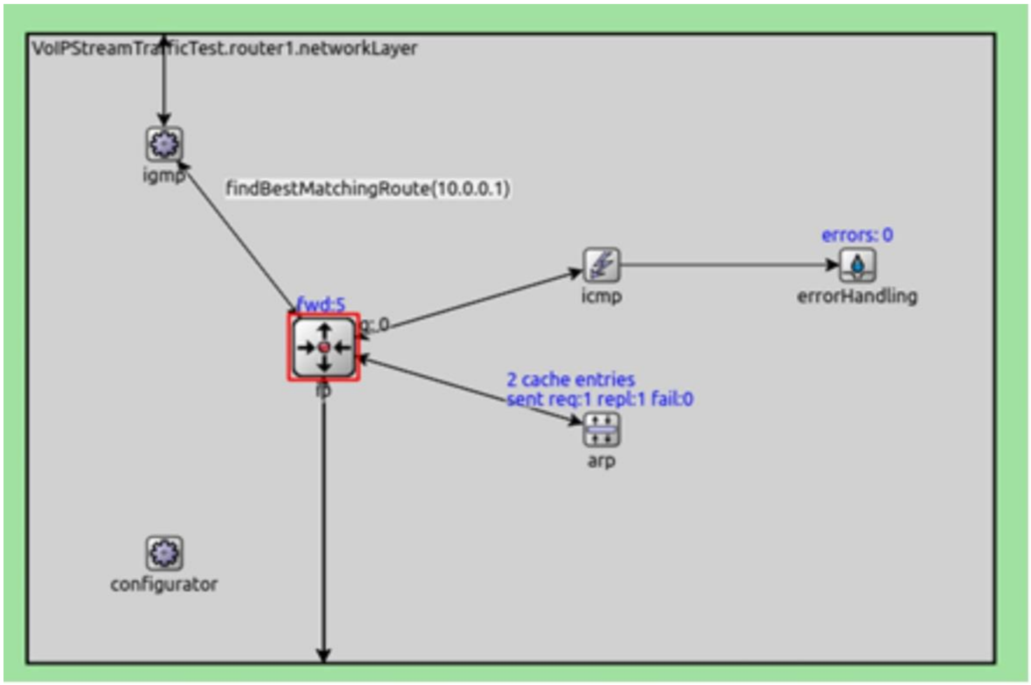


Figure 6: Internal Activities on Host (Physical Layer)

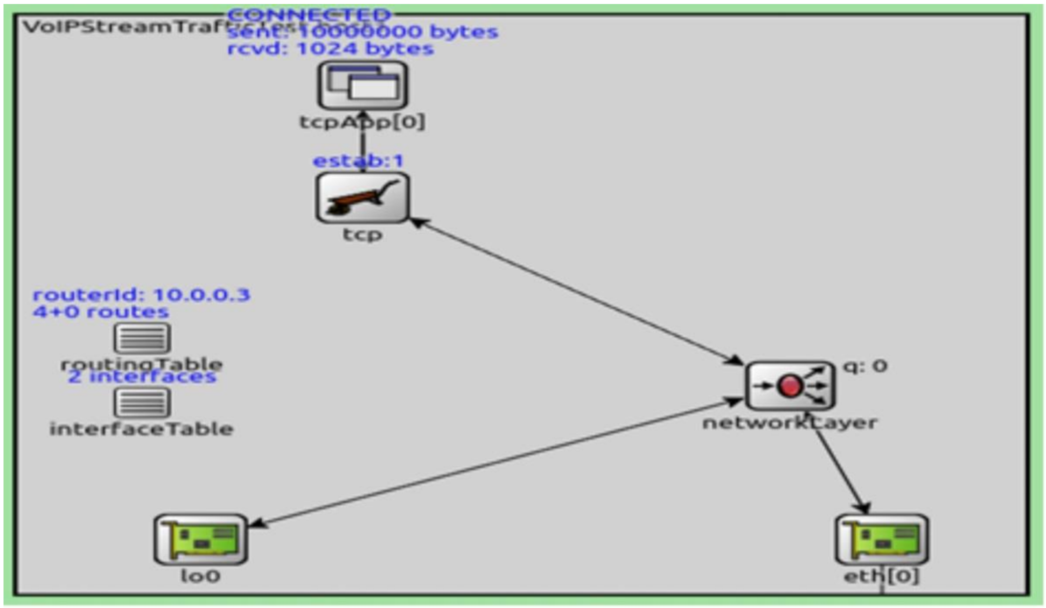


Figure 7: Internal Activities on Router (Network Layer)

Table 3: Simulation Parameters for Proposed Wired Topology

Simulation Parameter	Value / Type
Simulation Time	150 hours
Channel Type	Wired
Channel Delay	10us
Link Speed	Client – Server = 10mbps Client /host – Switch = 10mbps Router – Router = 100mbps
Packet Length (in time)	45ms
Packet Size	5420kbps
Interval	100ms
Sampling Rate	7000Hz
Send Bytes	1000000000 bytes
Protocols for transmission	TCP UDP
Queue Type	Drop Tail Queue
Switch Relay Unit Type	Mac Relay Unit
Routing Protocols	BGP (Border Gateway Protocol)

3.4 Classification Queues

Classification is implemented at the network layer in the routers. Drop tail queuing is the adopted model for queuing the queuing process. Drop tail queue is an easy to implement mechanism that determines the manner and when packets are dropped. All packets are assigned the same priority level. When the queue is filled to its maximum capacity, new packets are dropped from the tail of the queue until enough space to accommodate packets is created. Figure 8 demonstrates the queuing process in the study. The packets are aligned in a queue at the router before being encapsulated and sent to the next layer as depicted in 8(a). In 8(b), the internal processes are visualized. There exist a classifier, pause queue, data queue and a scheduler. After classification, the packets go to the data queue to be scheduled for the next hop or layer. When the data queue reaches its maximum capacity, incoming packets

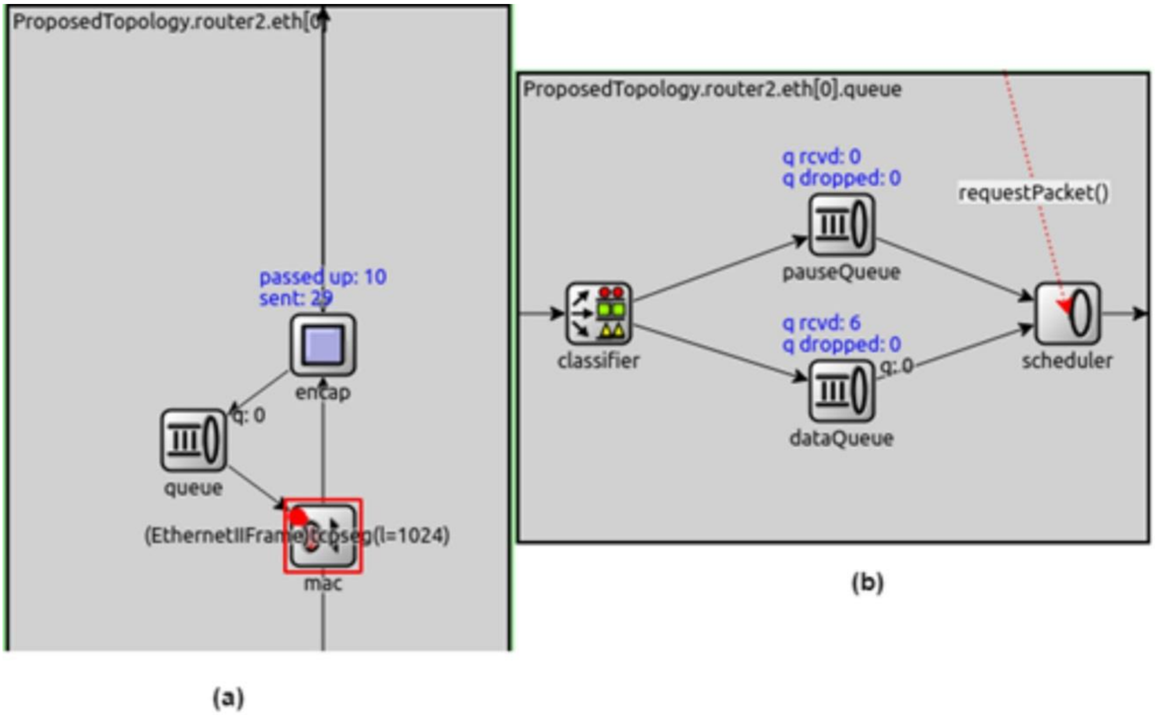
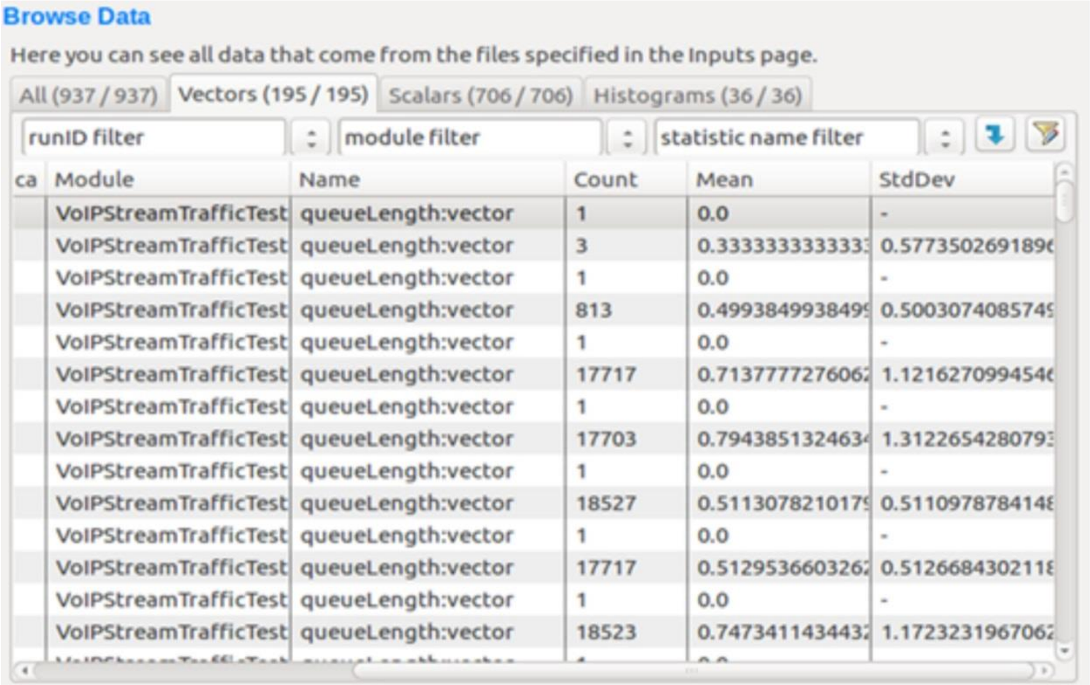


Figure 8: Classification Queuing Mechanism at the Network Layer

are sent to the pause queue. The pause queue puts the scheduler on hold until packets in the data queue are fully processed. However, packets are dropped when all queues are full.

3.4 Results of Topology Implementation

After completion of the simulation, the results are presented in three tabs by OMNET. These are the Vectors, Scalars and Histogram tabs. Vector section gives the record of the data values as a function of time. It includes the inter-arrival time of packets, the time covered in hops, hop counts, time to reach destination nodes and all other records of time events with respect to the requirements of the network. It also gives the mean values and standard deviation values as well. Figure 9 shows the Vector results as per the simulation run. The scalar tab shows records of the aggregate values at the completion of the simulation. This includes the output attained with the parameter values incorporated in



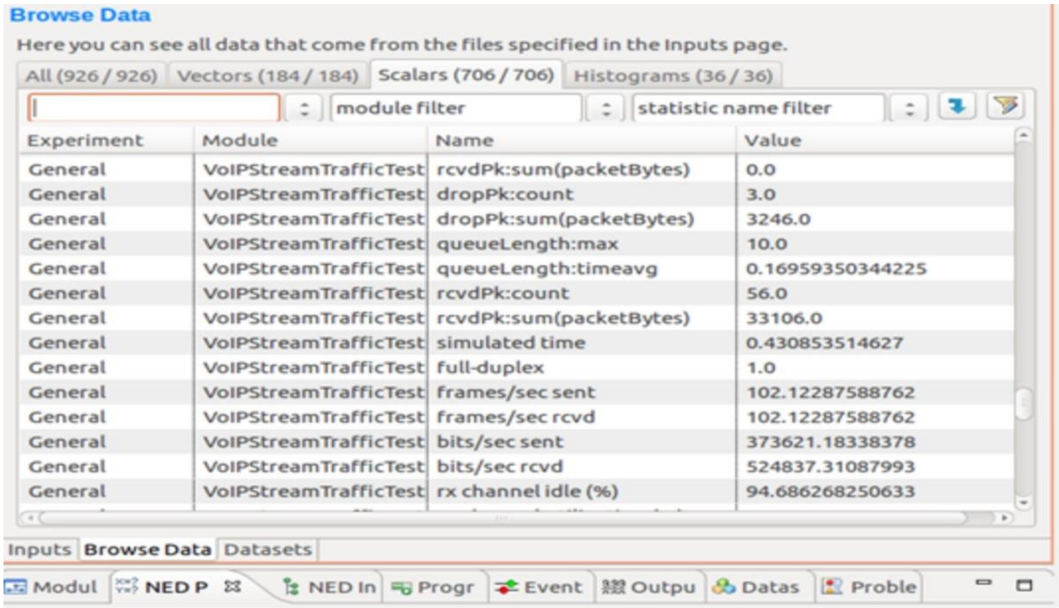
Browse Data

Here you can see all data that come from the files specified in the Inputs page.

ca	Module	Name	Count	Mean	StdDev
	VoIPStreamTrafficTest	queueLength:vector	1	0.0	-
	VoIPStreamTrafficTest	queueLength:vector	3	0.333333333333	0.577350269189
	VoIPStreamTrafficTest	queueLength:vector	1	0.0	-
	VoIPStreamTrafficTest	queueLength:vector	813	0.49938499384	0.50030740857
	VoIPStreamTrafficTest	queueLength:vector	1	0.0	-
	VoIPStreamTrafficTest	queueLength:vector	17717	0.71377772760	1.12162709945
	VoIPStreamTrafficTest	queueLength:vector	1	0.0	-
	VoIPStreamTrafficTest	queueLength:vector	17703	0.79438513246	1.31226542807
	VoIPStreamTrafficTest	queueLength:vector	1	0.0	-
	VoIPStreamTrafficTest	queueLength:vector	18527	0.51130782101	0.51109787841
	VoIPStreamTrafficTest	queueLength:vector	1	0.0	-
	VoIPStreamTrafficTest	queueLength:vector	17717	0.51295366032	0.51266843021
	VoIPStreamTrafficTest	queueLength:vector	1	0.0	-
	VoIPStreamTrafficTest	queueLength:vector	18523	0.74734114344	1.17232196706

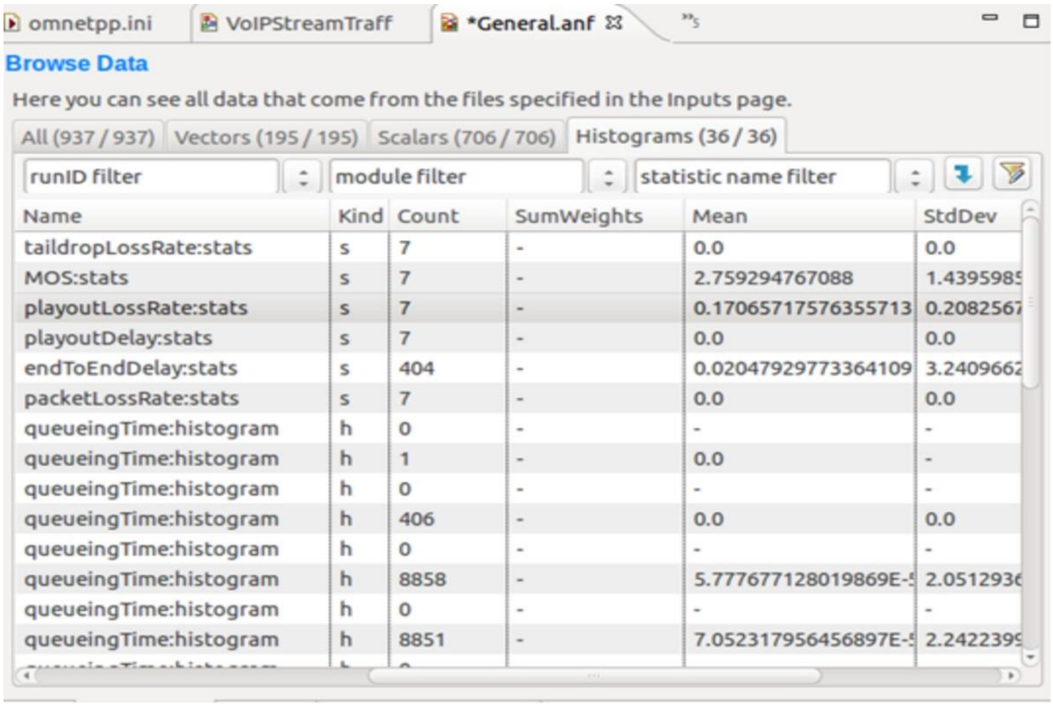
Figure 9: Simulation Results Showing Vector Values

the simulation including drop count, delay in transmission, throughput, total sum of received packets with frames sent as shown in Figure 10. Histogram tab gives the statistics of the simulation results that can also be plotted into a chart. It presents the statistics of the entire simulation results such as the tail drop packet rate, delay in transmission, the rate at which packets are lost and queuing time as illustrated in Figure 11. The results are then saved into a .csv file and exported into MATLAB for further clustering and further analysis.



Experiment	Module	Name	Value
General	VoIPStreamTrafficTest	rcvdPk:sum(packetBytes)	0.0
General	VoIPStreamTrafficTest	dropPk:count	3.0
General	VoIPStreamTrafficTest	dropPk:sum(packetBytes)	3246.0
General	VoIPStreamTrafficTest	queueLength:max	10.0
General	VoIPStreamTrafficTest	queueLength:timeavg	0.16959350344225
General	VoIPStreamTrafficTest	rcvdPk:count	56.0
General	VoIPStreamTrafficTest	rcvdPk:sum(packetBytes)	33106.0
General	VoIPStreamTrafficTest	simulated time	0.430853514627
General	VoIPStreamTrafficTest	full-duplex	1.0
General	VoIPStreamTrafficTest	frames/sec sent	102.12287588762
General	VoIPStreamTrafficTest	frames/sec rcvd	102.12287588762
General	VoIPStreamTrafficTest	bits/sec sent	373621.18338378
General	VoIPStreamTrafficTest	bits/sec rcvd	524837.31087993
General	VoIPStreamTrafficTest	rx channel idle (%)	94.686268250633

Figure 10: Simulation Results Showing Scalar Values



Name	Kind	Count	SumWeights	Mean	StdDev
taildropLossRate:stats	s	7	-	0.0	0.0
MOS:stats	s	7	-	2.759294767088	1.4395985
playoutLossRate:stats	s	7	-	0.17065717576355713	0.2082567
playoutDelay:stats	s	7	-	0.0	0.0
endToEndDelay:stats	s	404	-	0.02047929773364109	3.2409662
packetLossRate:stats	s	7	-	0.0	0.0
queueingTime:histogram	h	0	-	-	-
queueingTime:histogram	h	1	-	0.0	-
queueingTime:histogram	h	0	-	-	-
queueingTime:histogram	h	406	-	0.0	0.0
queueingTime:histogram	h	0	-	-	-
queueingTime:histogram	h	8858	-	5.777677128019869E-5	2.0512936
queueingTime:histogram	h	0	-	-	-
queueingTime:histogram	h	8851	-	7.052317956456897E-5	2.2422399

Figure 11: Simulation Results Showing Histogram Values of Packet Transmission

3.5 Proposed Wireless Scenario

Though the scope of the research work focuses on wired topology, most of today's wide area networks incorporate wireless architecture. For this we propose a Wireless WAN setting as displayed in Figure 12. The network consists of three autonomous connected networks. The mobile hosts are disconnected and connected to a particular network depending on the range covered by the access points connected to each individual network. The wireless topology is subjected to the same situation of extreme packet loss and fragmentation with the parameters in Table 4. The data logs and transmission are captured and converted into a dataset which will be included for the validation of the proposed solutions.

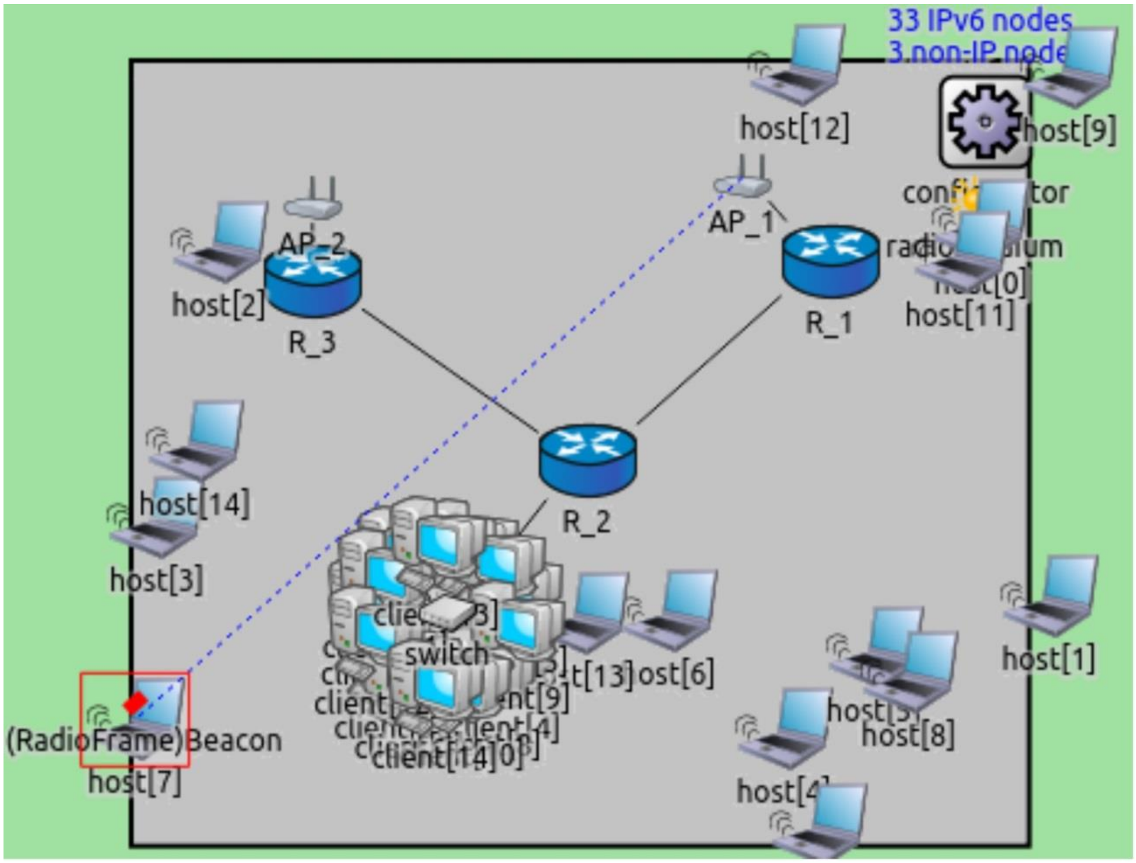


Figure 12: Proposed Wireless Topology

Table 4: Simulation Parameters for Proposed Wireless Topology

Simulation Parameter	Value / Type
Simulation Time	150 hours
Channel Type	Wireless with wired backbone
Channel Delay	10us
Link Speed	Client /host – Switch = 10mbps Router – Router = 100mbps
Wireless LAN bit rate	2Mbps
Packet Length (in time)	45ms
Packet Size	128bytes
Interval	100ms
Sampling Rate	7000Hz
Send Bytes	1000000000 bytes
Protocols for transmission	TCP UDP
Queue Type	Drop Tail Queue
Data queue frame capacity	10
Maximum Queue Size	12
Mobility speed	1mps
Mobility update Interval	0.1s

Mobility constraint Area	MinX = 150m
	MinY = 120m
	MaxX = 430m
	MaxY = 110m

Switch Relay Unit Type	Mac Relay Unit
Routing Protocols	BGP (Border Gateway Protocol)

3.6 FPL Dataset Capture and Statistics

There exist several datasets for traffic classification collected at links and Border routers in CIADA [126]. However, as at the time this study is being conducted, none of these datasets has considered capturing the data when the network is deprived of resources or not able to meet the requirements of the network. These parameters have been considered by this research to create such a dataset, a first of its kind to the best of our knowledge. Traces of packet transmission of data from source to destination are captured onto the server. We name the dataset derived from the proposed scenario as Fragmentation-Packet Loss induced (FPL) Traces. The dataset from the wired topology and wireless topology are referred as FPL 1 and FPL 2 respectively. The dataset consists of flows of IP packets generated from several applications. A total of approximately 5.1 billion IP packets with 81 million flows constituting to 920.04 GB data was collected. Out of these, various application flows are identified. A breakdown of the flows and its percentages are given in Table 4. P2P, Email, and Streaming applications have a further breakdown into specific application flows as depicted in Table 5, Table 6 and Table 7 respectively.

Table 5: Flow Statistics of FPL Dataset

Application Type	**FPL 1 No. of Flows**	**Flows (%)**	**FPL 2 No. of Flows**	**Flows (%)**
INSTANT MESSAGING	11,491	0.0%	403,436	1.0%
EMAIL	2,465,201	5.0%	5,367,912	16.0%
HTTP	14,530,023	30.0%	3,456,221	10.0%
FTP	389,673	1.0%	7,34,205	2.0%
P2P	9,582,023	19.0%	1,561,422	4.0%
STREAMING APPS	10,723,297	22.0%	12,003,129	35.0%
DIRECT LINKS (DOWNLOAD)	7,432	0.0%	18,457	0.0%
MPEG	3,445,170	7.0%	4,345,472	13.0%
WINDOWS MEDIA	54,257	0.0%	3,454,448	10.0%
ICMP	134,556	0.0%	567,390	1.0%
DATABASE	7,241,283	15.0%	457,889	1.0%
UNKNOWN	1,236,103	2.0%	2,389,017	7.0%
Total	**48,590,509**	**100%**	**32,464,932**	**100%**

Table 6: Statistical Breakdown of P2P Flows in FPL Dataset

Type	**FPL 1 No. of Flows**	**Flows (%)**	**FPL 2 No. of Flows**	**Flows (%)**
BIT-TORRENT	1, 461, 593	15%	128,512	8.3%
SKYPE	3,562, 323	37%	712,452	45.7%
EDONKEY	478, 590	5%	6,868	0.4%
GNUTELLA	1,936,100	20%	4,990	0.3%
GOSSIP	2,143,417	23%	708,602	45.3%
Total	**9,582,023**	**100%**	**1,561,422**	**100%**

Table 7: Statistical Breakdown of Streaming Application Flows in FPL Dataset

Type	**FPL 1 No. of Flows**	**Flows (%)**	**FPL 2 No. of Flows**	**Flows (%)**
YouTube	4,553,871	42%	4,101,455	34.2%
Netflix	6,169,426	58%	7901455	65.8
Total	**10,723,297**	**100%**	12,003,129	**100%**

Table 8: Statistical Breakdown of Mail Application Flows in FPL Dataset

Type	**No. of Flows**	**Flows (%)**	**FPL 2** **No. of Flows**	**Flows (%)**
MAIL_POP	989,342	40%	3,470,733	64.6%
MAIL_SMTP	643,994	26%	862,120	16.1%
MAIL_IMAP	831,865	34%	1,035,059	19.3%
Total	**2,465,201**	**100%**	5,367,912	**100%**

3.7 Effects of Packet Loss and Fragmentation

The incorporation of the parameter values in Table 3 and 4 will result in a high packet drop count with the amount of sent bytes exceeding the capacity link can handle at a time. A graph of packet drop count against inter-arrival time is plotted in both cases of the proposed topologies. From the wired topology in Figure 13, as packet drop count increases, the pace at which packets arrive also increases. This depicts that latency to reach the destination is increased with rapid packet loss in the network. Results from the graph in Figure 14 depict the case of the wireless scenario. It can be observed that throughput also declines with increased latency. However, the rate at which throughput declines is dynamic. This is due to the mobility of the clients as they move from one network to another. The network they are connected to changes depending on the maximum coverage radius of the access points When a network has too many hosts connected at a point in time throughput declines rapidly with increasing inter-arrival time. When the number of hosts connected at a point in time is less, the rate of declination reduces. This fulfills the first objective’s investigation. Table 9 shows the ratio of packets transmitted and received, which is further revealed in percentages also depicted in Figure 15, at the end of the simulation. It can be seen that the ratio of sent packets to the received is approximately 2:1, representing almost 50% of the packet transmitted were dropped. Overall throughput decreases with increasing inter-arrival time in both cases of the proposed network environments. Hence, only a limited or small amount of flows can be classified at a time and few packets features can be extracted from the limited flows for the purpose of classification.

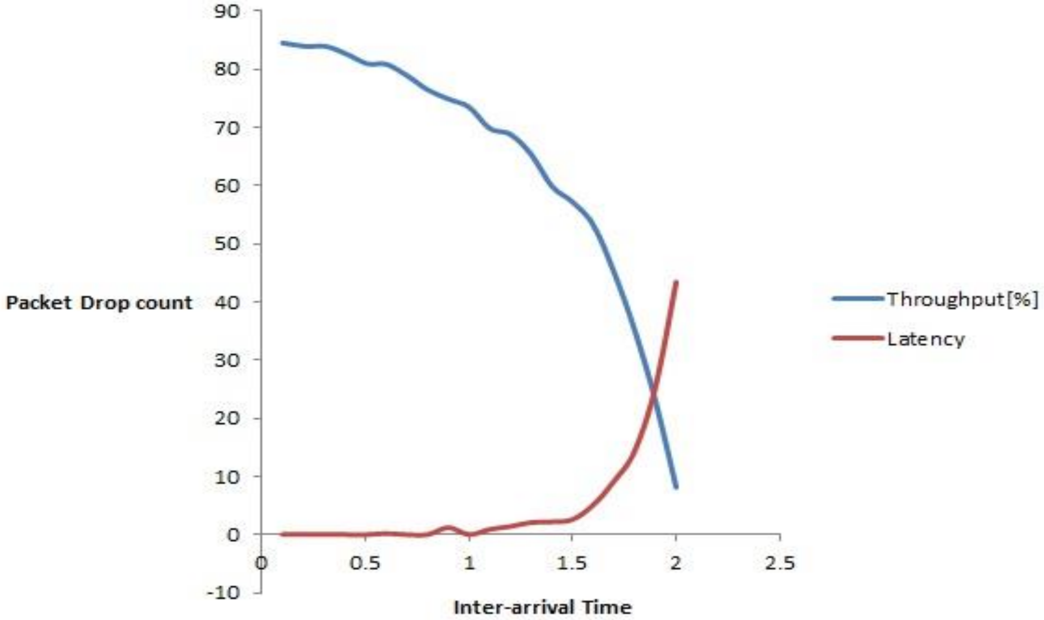


Figure 13: QoS Effects of Packets Loss in Classification Procedure in Proposed Wired Scenario

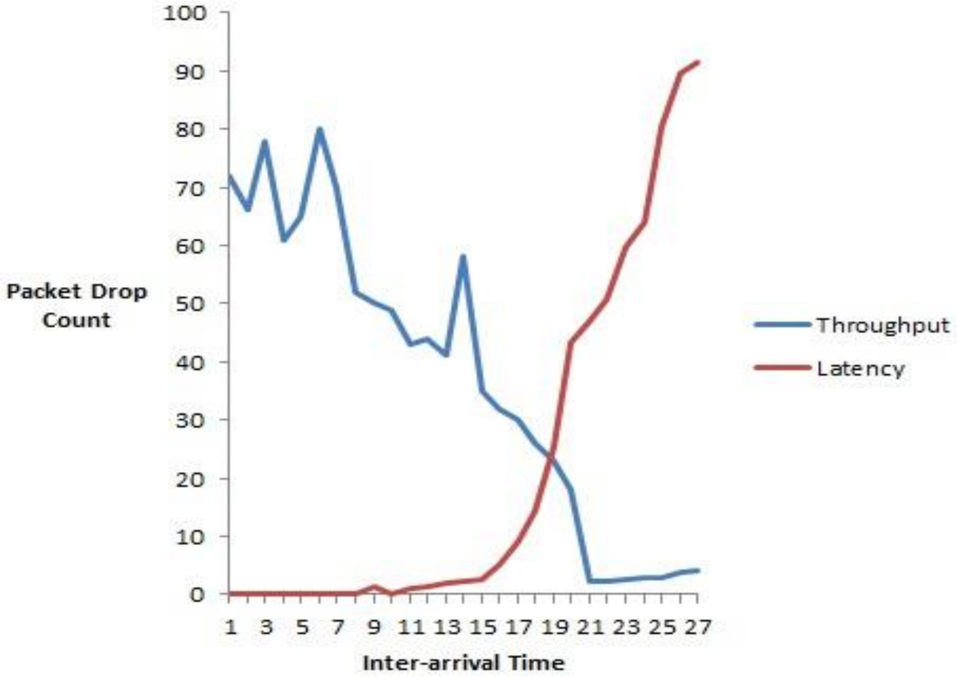


Figure 14: QoS Effects of Packets Loss in Classification Procedure in Proposed Wireless Scenario

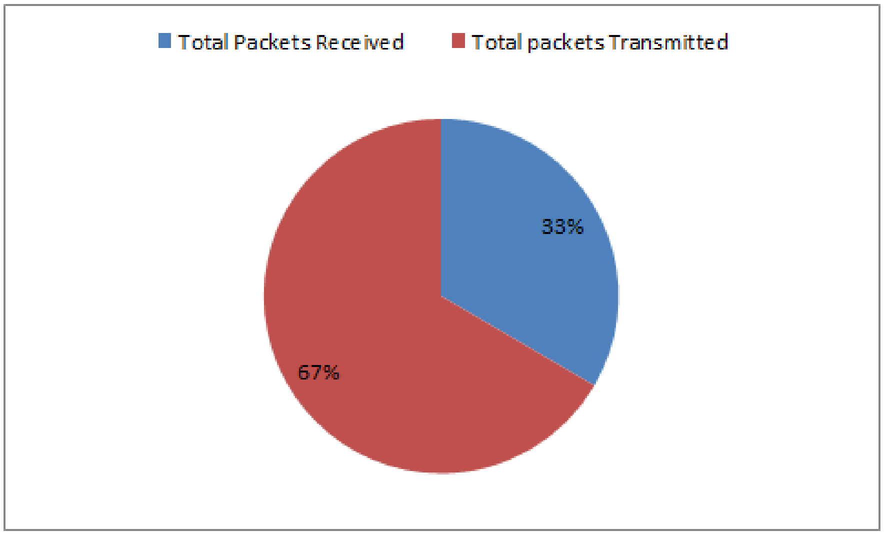


Figure 15: Overall Chart of the Total Number of Packet Flows (Transmitted Flows and Received Flows)

Table 9: Statistics of Transmitted Bytes and Received Bytes after Simulation

Parameters	Sum of bytes transmitted	Sum of bytes received
Number of IP Packet (bytes)	10,327,610,944.31770000	5,179,977,888.79730000
Percentage (%)	66.59714192	33.40285808
Ratio	1.99375579703788 : 1	

To validate the proposed framework as a stand-alone classifier for traffic classification, the outcomes and results are compared with two existing renowned works namely, K-Nearest Neighbour (KNN) classifier [127], KNN+K-Means hybrid classifier [72] and Inter-arrival Time Precision classifier [121]. All classifiers are applied to the generated dataset with cross-validation in five folds. The results are assessed with the metrics as discussed previously.

3.8 Summary

In this chapter, we set to design and implement the proposed topology and environment where the problem statement is likely to occur. The wired and wireless network topologies are implemented in a simulation environment in OMNET ++ . The parameters of the simulation are such that it ensures the network is prone to rapid packet loss. Due to the low speed and capacity of links, the higher arrival rate of packets on the low capacity of links this ensures a lot of congestion leading to fragmentation. The data logs collected at the server side contains entire statistical values of packet flows which include packet drop count, inter-arrival time, hop count, delay in transmissions, number of packets sent and received in each successful transmission. Gathered data logs and captured traces of transmission throughout the entire simulation period are converted to .csv format to be exported to MATLAB to serve as the dataset for the study.

Chapter 4: Design and Implementation of Proposed Algorithms

4.1 Motivation for Algorithms Design

With respect to the second objective, the study aims to propose an algorithm with a higher prediction and accuracy rates during phases in a network where the quality of service is less. In order to achieve this, real-time transmission implemented in simulation is captured. We identified in the previous chapter that few flows resulting in few features can be extracted and classified at a time. Hence, few flows will be used for the training procedure. To develop an algorithm to this effect, factors such as distance error, computational complexity must be highly considered. The algorithm in order to achieve high accuracy and high prediction rate must almost not tolerate any form of error in the distance calculation between the data points. Since resources are already limited in our case study a fast and easy to implement algorithm is required. Hence, the consideration of computational complexity is included.

4.1.1 Minimal Distance Error

An indebt survey of literature is conducted in chapter 2 from which we derived that the classical k-means approach of clustering (including hybrid approaches) yielded the most efficient results. This is due to the nature of partitional algorithms which tend to break the dataset into groups and reduce the distance between data points. This kind of process helps to identify flows with similar features or characteristics Due to the parameters incorporated in the topology few flows from the dataset will be used in the training process. It is therefore a necessity to achieve the most minimal distance error between data points. Since our dataset is likely to contain multiple fragments of data packets, a partitional algorithm like K-Medoids [128] and K-Means will be able to generate finer groups of packets which possess similar features. An optimized K-Medoids algorithm is adopted for the clustering phase in the proposed algorithms.

4.1.2 Ease of Use and Complexity of Algorithm

After the clustering process, the algorithm must assign the output of the clustering (which now becomes the input for the next phase) into peculiar groups. This is achieved with classifiers. There are several machine learning classifiers including Perceptron, Naïve Bayes, Decision Tree, Logistic Regression, K-Nearest Neighbour (KNN) and Support Vector Machine (SVM) [129] [130] [131][132].

Perceptron classifiers are built on the concept of Artificial Neural Networks. It is a linear classifier that employs a binary function to decide if the input is associated with a class or not. The concept of random weights generation is employed in all iterations of simulation. Naïve Bayes classifier makes use of a probabilistic function that makes the assumption of total independence of each feature value with respect to other features in a class. Decision Tree adopts the concept of a tree with branches (observations made) and leaf nodes (conclusions drawn). It derives information from observations of an item and uses that to draw conclusions about an item in question or target. Logistic Regression classifiers are normally used to predict problems with two class values (could be extended) in the field of statistics. An output curve in an 'S' shape is generated which accepts real numbers and maps them to values ranging from 0 to 1. KNN classifier also termed as the Lazy Classifier. It omits the whole process of training data points. A data point is assigned to a class based on the vote popularity of its closest neighbours. With SVM classifiers, the goal is to establish a line between classes in order to maximize the distance between the classes. The larger the distance between the classes, the closer the data points in a class and better predicted classes will be produced. Table 9 describes the strengths and weaknesses of these machines learning classifiers.

Table 10: Summary of Machine Learning Classifiers

Classifier	**Type**	**Strengths**	**Weaknesses**
Perceptron	Linear Model	Ability to infer unseen or unknown data Input variables have no restrictions	Hard limit function characteristic allows it to take only 1 value (0 or 1) Only vectors which are linearly separable can be classified
Naïve Bayes	Probabilistic Model	Easy to implement Fast prediction of test data For parameter estimation, training requires few data samples	There is a chance that accuracy can be lost Due to the assumption of Independent variables, dependency nature of variables cannot be modified or improved
Decision Tree	Linear Model (with respect to Decision rules)	Process is very transparent Decision making is less ambiguous Each decision outcome can be analyzed comprehensively Flexible to allow both categorical and real value features	Very unstable in nature Not very accurate relatively Biased towards categorical data possessing distinct number levels Complex calculations evolve with uncertain and linked output values
Logistic Regression	Statistical Model	Requires less computational resources Easy Interpretation No requirement of scaling input features Results in well computed probability predictions	Cannot be implemented on non-linear problems Only performs better when the required independent variables have been identified Predicts only categorical values as output
KNN	Non-Parametric Model	Possess the ability to classify outliers (noise) Very fast and easy to implement Training stage can be omitted which reduces the time complexity Gives the flexibility to	The value of the K parameter is a requirement to be determined If training samples are many, high costs can be incurred

		opt for other distance metrics Evolves easily with new data or changes	
SVM		Works well with both linearly and non-linearly separable problems High accuracy prediction rates with the use of its kernel tricks Problem of solving for a local optima is avoided	User must experiment with several kernels to find the right parameters for the problem

With time complexity in consideration, a classifier such as K-Nearest Neighbour [127] [107] is incorporated. KNN is easy to implement and omits the training aspects of a dataset contributing to lower time complexity of the suggested algorithm to be implemented. Furthermore, SVM is also opted for due to its strengths and ability to work with non-linear datasets. These classifiers are optimized further into algorithms that are easy to use with good computational complexity.

4.2 KNKM Algorithm

A hybrid semi-supervised algorithm is proposed, implemented and evaluated. The hybrid approach consists of combining the advantages of K-medoids algorithm and KNN classifier. The semi-supervision of the hybrid algorithms falls under the cluster and label mechanism. Implementation of the suggested algorithm is performed in MATLAB. The recommended algorithm consists of three stages namely, data filtering, clustering, and labelling. Raw data in .csv format imported into MATLAB is filtered to convert all the content of numeric strings into numbers. Where contents of fields are formatted as strings, NaN (Not a Number) is put in its place. The clustering process is initiated and performed to establish the features within flows and extract them. The total number of features extracted and clusters revealed will serve as input and a form of supervision for the labelling process. A Brief discussion of these selected algorithms constituting the proposed algorithm is described in the next sections.

4.2.1 The K-Medoids Clustering Strategy

The K-medoids is a long established partitioning clustering as an improved K-means. The best representation of a cluster in K-means is to select the average distance of the data points resident in the cluster. The mean does not match or necessarily correspond to a point in the original dataset. The Euclidean distance is adopted to calculate the mean. Contrarily, K-medoids selects a single data point among the rest to exemplify the cluster. These data points are also termed as exemplars. The advantage of exemplars is to help reduce the total sum of the data objects' dissimilarities. The middlemost point within a cluster is described as the medoids. Other distances other than the Euclidean distance such as Manhattan distance can be used in K-medoids. K-medoids is sturdier to noise and outliers when compared to K-means, motivating us to adopt K-medoids for the proposed algorithm.

4.2.2 K-Nearest Neighbor Algorithm (KNN)

KNN stands as an elementary supervised classification algorithm method, unlike the unsupervised K-means. The amount of data points that must be trained lies in proximity to the data points which is used to predict the particular class it belongs to, is the *k*. The advantages of this algorithm for classification include its predictive power, ease of output interpretation, and low computation cost. KNN just like K-medoids has the flexibility of utilizing several distance measures. Furthermore, it is a non-parametric algorithm; therefore no assumptions must be met before implementation.

4.2.3 The proposed algorithm

The KNKM algorithm is as shown in Table 10. The clustering process is performed with K-Medoids with Silhouette. The features for each cluster are extracted and the number of clusters revealed as *K* serves as an additional input to the KNN classifier for classification. For example, if *n* clusters are identified when the clustering procedure is concluded, the initial assignment of k for the labelling procedure using KNN algorithm will be *n*. With the extra information from the clustering process, Weighted KNN is employed to label the dataset given by Equation (3).

$$f(x_q) = \arg Max_{v \in V} \sum_{i=1}^{k} w_i \ \sigma(V, f(x)) \qquad - \qquad (3)$$

Where x_q represents the query point, *i* represent each data point, w_i is the weight of a point to a k neighbour and V represents a vector point. The contribution from every individual k neighbour is weighed with their respective distances to query point $x_{q.}$ As a result greater weight w_i is allotted to closer neighbours. The weight w_i is given by:

$$w_i = \frac{1}{d\,(x_{q,}x_i)^2} \qquad - \qquad (4)$$

Table 11: KNKM Proposed Hybrid Algorithm

Step 1: Load dataset	Data Filtration
Step 2: For each column in the dataset: • Convert contents of string fields with numeric values to numbers • Replace strings without numeric values with NaN	
Step 3: Select *Km* as the Medoids for *n* data points at random.	Clustering
Step 4: Compute the distance between all data points *n* and the selected Medoids *k* for the closest Medoids.	
Step 5: For each duo of data object *j* not chosen and elected data object *p*, calculate total cost to swap, TC*pj*. • If TC*pj* < 0, *p* is substituted by *j*	
Step 6: Re-run steps 4-5 until convergence is reached if and there is no change in the assignment process.	
Step 7: Apply Silhouette to obtain fine clusters with its outliers. Quantity of clusters *k* and feature labels obtained serves as an input for the labelling.	
Step 8: Load the input data into KNN classifier	Labelling
Step 9: Initialize the *k* value	
Step 10: For every query point $x_{q,}$ For every data point, *i = 1 to n* • Compute the weighted distance to all data points • Save the all W_i computed and sort the list	
Step 11: Select the first *k* points from the list with its corresponding *k*-distances to determine k-th Minimum distance	
Step 12: For k ≥ 0, Let ki and kj represent number of points in an i^{th} and j^{th} class among k points, • If $k_i > k_j$ ∀ i ≠ j, assign $x_{q,}$ to class *i*.	

Hence, more weight is allotted to data points closer to the query point and less weight to points which are far from the query point. The overall classification is performed with 5-fold cross-validation to identify classes the flows belong. The design of the suggested mechanism and implementation is depicted in Figure 16.

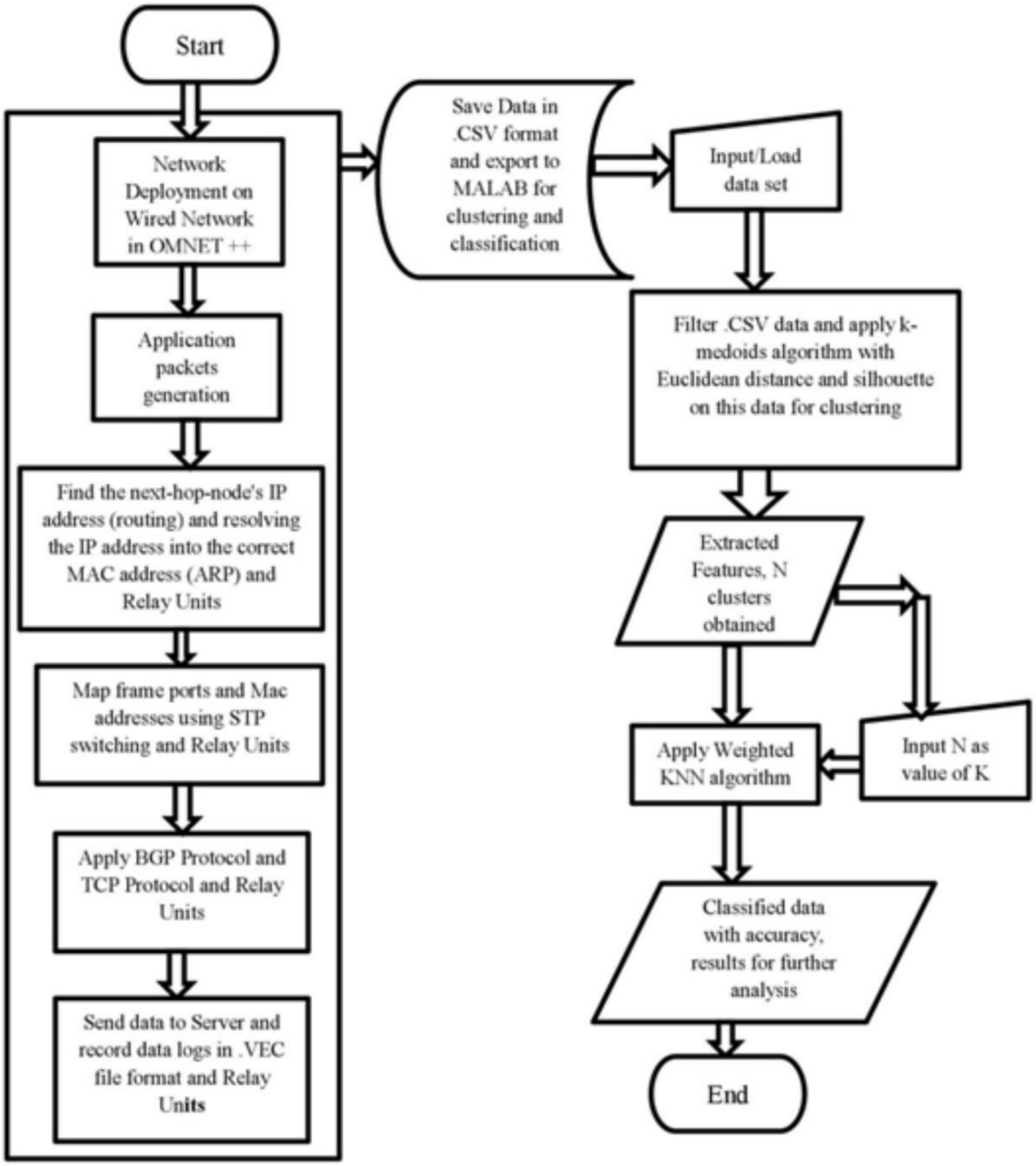

Start
Network Deployment on Wired Network in OMNET ++
Application packets generation
Find the next-hop-node's IP address (routing) and resolving the IP address into the correct MAC address (ARP) and Relay Units
Map frame ports and Mac addresses using STP switching and Relay Units
Apply BGP Protocol and TCP Protocol and Relay Units
Send data to Server and record data logs in .VEC file format and Relay Units
Save Data in .CSV format and export to MALAB for clustering and classification
Input/Load data set
Filter .CSV data and apply k-medoids algorithm with Euclidean distance and silhouette on this data for clustering
Extracted Features, N clusters obtained
Input N as value of K
Apply Weighted KNN algorithm
Classified data with accuracy, results for further analysis
End

Figure 16: Overall Flow of Proposed KNKM Design and Implementation

4.2.4 Complexity of KNKM

KNKM algorithm executes with linear time, in that the time it takes for the algorithm to run into completion depends on the number or size of its input data. In notation form, the complexity of KNKM can be represented asymptotically as $O(k(n-k)^2)$ where k represents the number of clusters n is the size of input data or number of instances.

4.2.5 Results of KNKM on FPL 1 Dataset

The results after implementing the algorithm on the dataset show 5 clusters were revealed after clustering with K-Medoids and Silhouette as shown in Figure 17. After Silhouette tool is applied, the flows of packets are represented in a bar chart in Figure 18. The longer bars represent the inliers or how high the flows are similar features. The shorter bars represent the outliers of a particular cluster. Application of Weighted K-NN with 5-fold cross-validation is discussed under the following metrics: Precision, Accuracy, Processing time, Area under Receiving Operating Curve (ROC) and Error Rates. Precision is explained as a prediction of an accuracy measure. Estimation of precision values is expressed in Equation (5):

$$Precision = \frac{Classess\ Predited\ Correctly(TP)}{Classess\ Predited\ Correctly(TP) + Classes\ Predicted\ Incorrectly(FP)} \quad - \qquad (5)$$

TP symbolizes True Positives while TF characterizes True Negatives. A graph of precision versus inter-arrival time is plotted. Increasing rates of inter-arrival time are experienced with the increment in packet drop count from the scenario in the topology. Hence, in the worst scenario of a higher or increasing inter-arrival time, precision values ranging from an average of 82% to 94% can be obtained with the hybrid method as exhibited in Figure 19. The achieved range presents a good prediction that accuracy value after classification with the proposed is likely to be better. Accuracy represents the amount of classes the proposed classified correctly out of the total number of classes as given in Equation (6):

$$Accuracy = \frac{Number\ of\ Correctly\ Predicted\ Classes(TP+TN)}{Total\ Number\ of\ Predicted\ Classes(TP+TN+FP+FN)} \quad - \quad (6)$$

where TN denotes True Negatives, FN means False Negatives and FP exemplifies False Positives. For accuracy of classification, derived confusion matrix graph is employed to determine the effective rate of the classifier. From Figure 20, KNKM predicted 20 classes after classification. In the case of classifying P2P traffic, traffic belonging to Skype class is predicted wrongly as Edonkey. A similar misclassification occurs with streaming applications where YouTube class is predicted as Netflix. In addition, all mail generated traffic is classified as mail_smtp, irrespective of whether it is mail_pop, mail_imap or mail_smtp, identified in the true class.

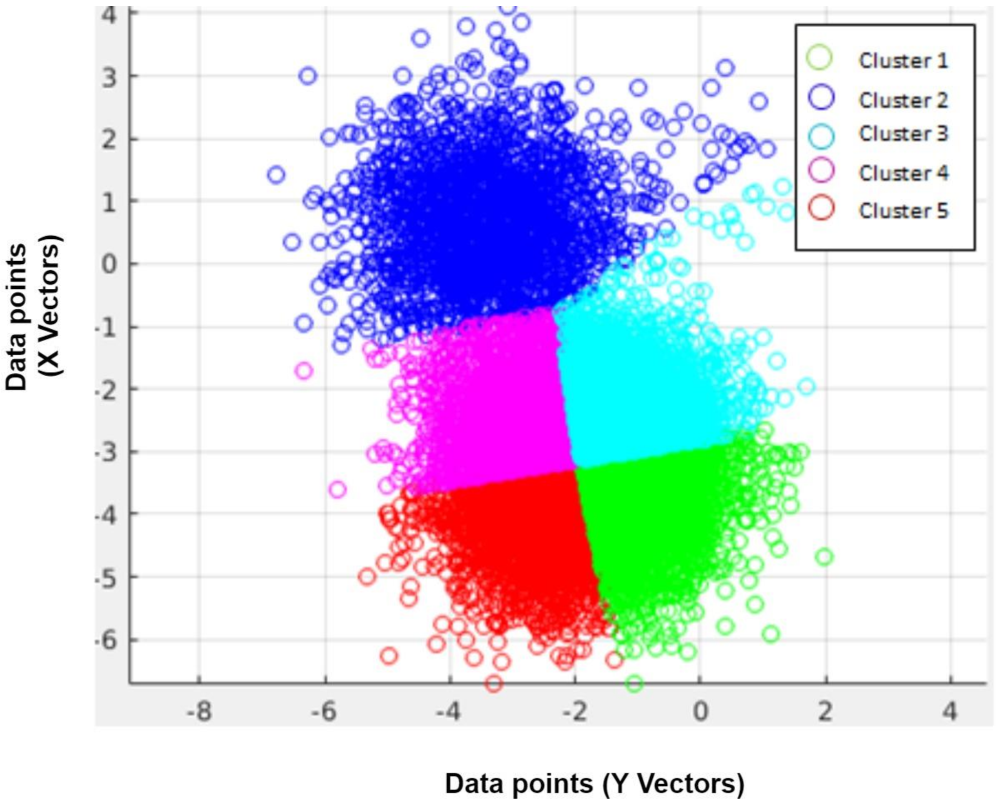


Figure 17: Identified Clusters after Clustering Phase

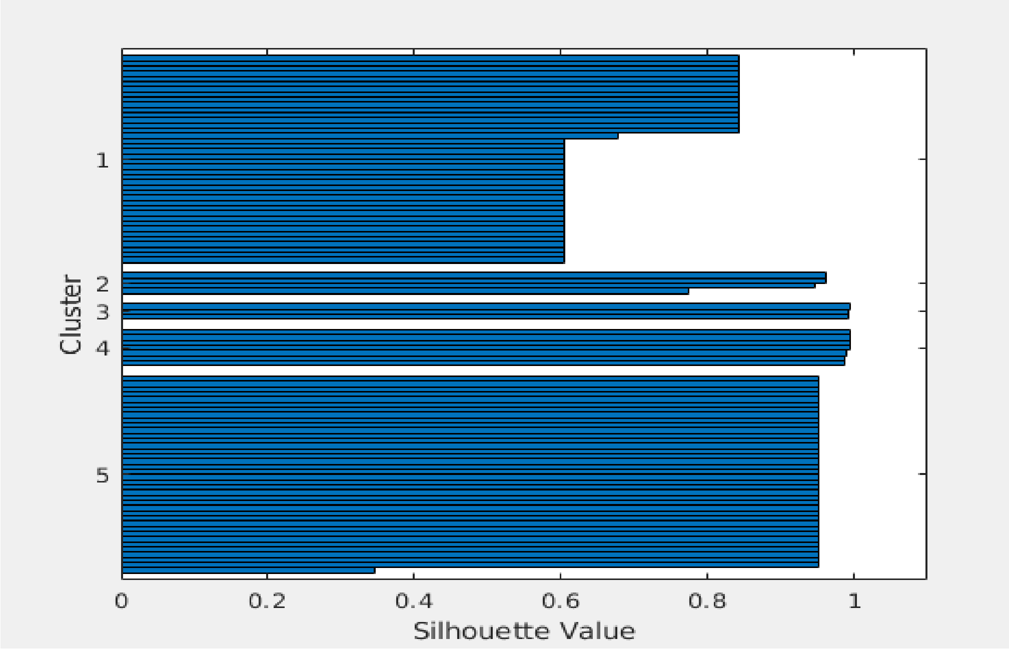


Figure 18: Clusters Revealed Showing Inliers and Outliers

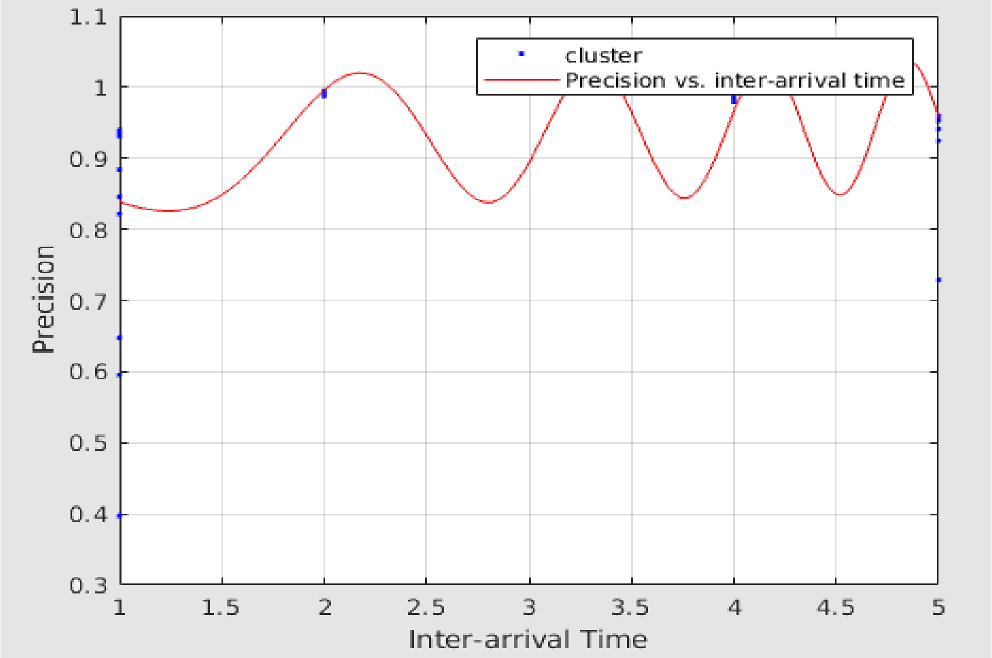


Figure 19: Precision Sine Wave Curve for KNKM Algorithm

Mpeg is also misclassified into the windows media class. Windows media framework has a protocol known as Media Transfer Protocol (MTP) which enables the transfer of media files between portable devices. Remote Access Protocol (RDA) is used by database applications to access data from remote locations. The foreign or unknown traffic injected were identified, but also misclassified as belonging to the database class or unknown class. This could be that the unknown traffic bears some feature similarities with the database generated traffic. From the misclassification, it can be observed that the classifier is able to identify the type of traffic (either P2P, Mail, streaming, etc.) but not distinctively classify it. Out of the 20 classes, 12 classes were classified correctly while 8 classes were misclassified giving it an accuracy of 91.3%.

Receiver Operating Characteristics (ROC) curve plots TP rates versus 100-Specificity for varying parameter checkpoints. The area covered by the ROC curve (AUC) depicts a measure of parameter’s performance to differentiate the various classes. The closeness of the curve to the topmost corner on the left denotes how high the long-term accuracy of a model stands on a scale of 0 to 1, where 1 is highest and 0 is lowest. From Figure 21, the AUC graph shows an area of 1.0 which concludes that the proposed classifier gives a high overall accuracy prediction. KNKM achieved an error rate of 8.7% with 4.5482 seconds of processing time.

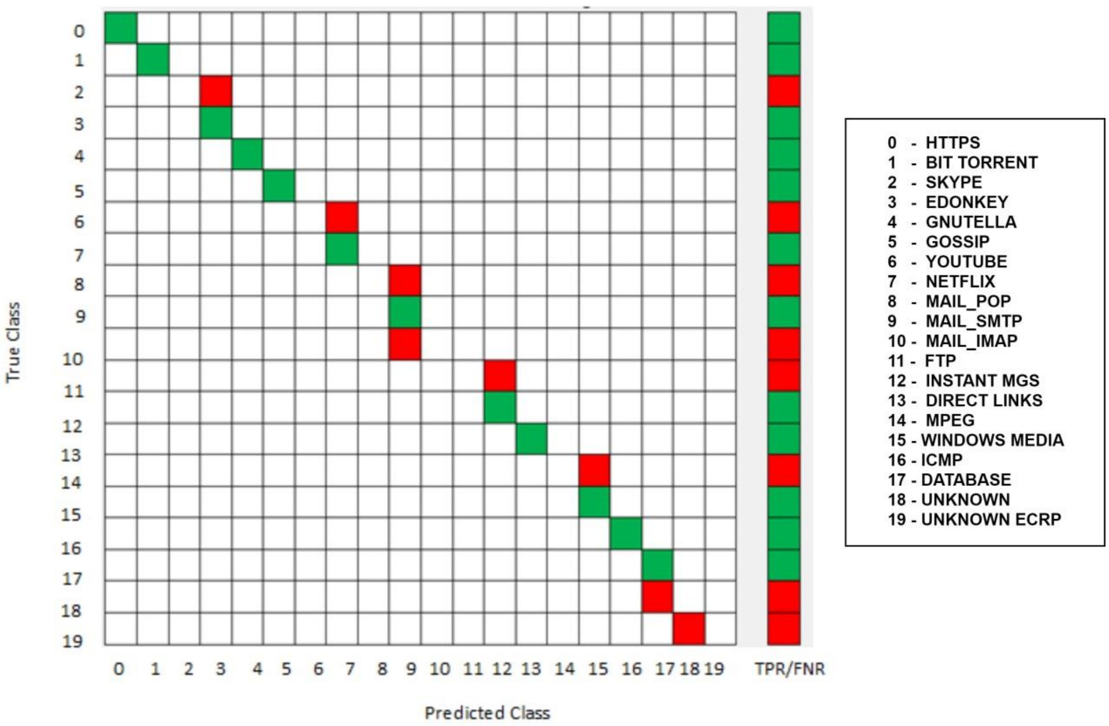


Figure 20: Confusion Matrix Graph of True Classes against Predicted Classes

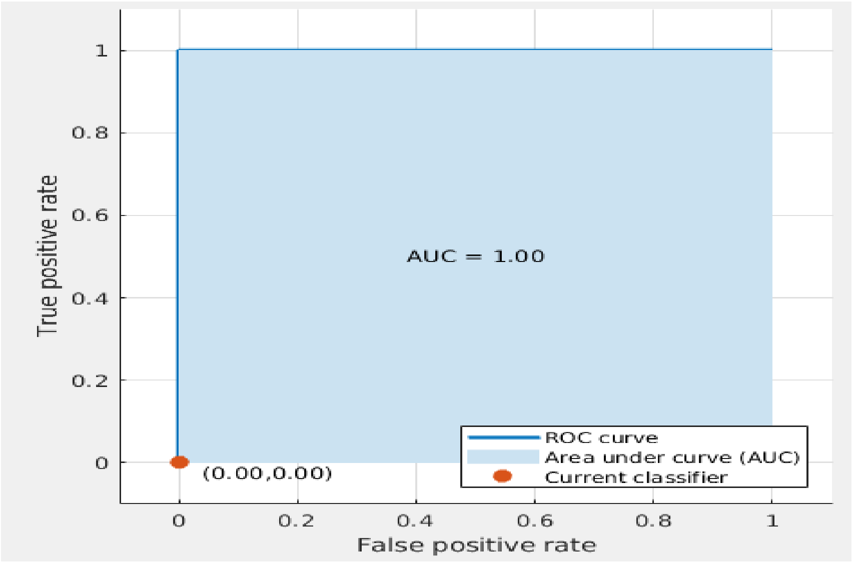


Figure 21: Area under ROC for KNN+K-Medoids Classifier

4.2.6 Results of KNKM on FPL 2 Dataset

The algorithm is tested on the dataset generated from the wireless scenario and evaluated in terms of accuracy of classification. From Figure 22, KNKM predicted Skype traffic as Edonkey and YouTube as Netflix. There were misclassifications as well with respect ot unknown and unknown encrypted traffic as well mails generated with the IMAP protocol and FTP traffic. KNKM predicted 11 classes precisely out of the 20 classes with an overall classification accuracy of 90.86% and error percentage rate of 9.14%. The accuracy in comparison to FPL 1 decreases at a rate of 0.44%.

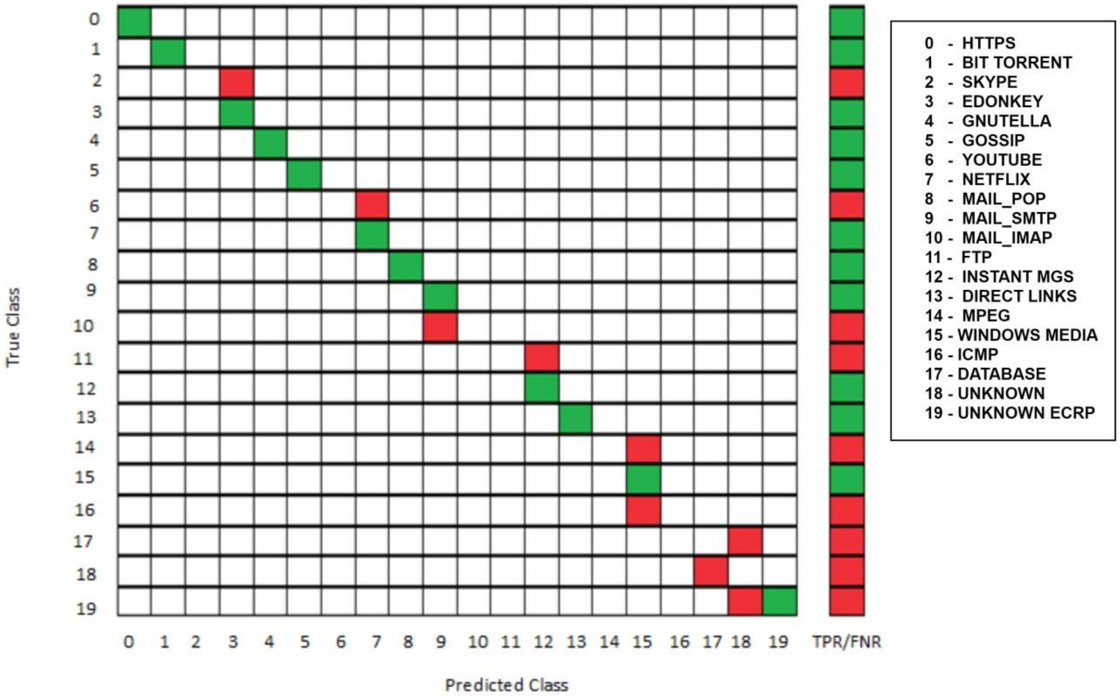


Figure 22: Confusion Matrix Graph Showing the True Class against Predicted Class in Wireless Environment (KNKM)

4.3 SVKM Algorithm

4.3. 1 Motivation of Algorithm Design

To improve recuperate accuracy in classification and scale down error rates as suggested according to the formulated objectives of the study, we optimize the parameters of the labelling process by introducing kernel functions of Support Vector Machines which has the capacity to work with non-linear datasets effectively. The hybrid algorithm is an improvement of the KNKM algorithm, which employs the advantages of K-medoids and Support Vector Machine algorithms.

4.3.2 Proposed Algorithm Implementation

A cluster and label algorithm is again proposed, implemented and evaluated. The hybrid model combines the advantages of K-medoids algorithm and SVM classifier. MATLAB simulation is employed in the algorithm implementation. The proposed algorithm is demonstrated in Table 12. Figure 23 displays the overall operations of algorithm implementation. The algorithm has three phases which consist of data filtering, clustering, and labelling. Similarly to the former algorithm, the raw data is filtered and clustered with the K-Medoids algorithm. The features residing in flows are extracted in addition to the number of clusters revealed. The number of features extracted and clusters revealed serves as guidance for the labelling process. The distinct difference between SVKM and KNKM lies in the labelling phase. While the former incorporates the SVM algorithm for its labelling the later utilizes KNN algorithm. A brief discussion of Support Vector Machine is described from the next section.

4.3.3 Support Vector Machine (SVM)

The exertion of Support Vector Machines [133] relies on supervised models possessing related algorithms for learning tasks, for analyzing large amount data for classification, regression and pattern identification purposes. With SVM, almost the entire attributes are utilized to create parallel partitions giving rise to two parallel lines using hyperplanes with margins in high-dimensional space. The given data is separated into classes using the hyperplane margins. Greater margins are directly proportional to lower error rates of the classifier. SVM has the advantage of being flexible and robust which generally gives its exact precision predictions. It is however sensitive to the kernel parameters selected for its implementation leading to a possible high computational complexity. SVM can be categorized as linear or non-linear according to the nature of datasets to be classified. Linear SVM accepts that the examples to be trained in space are parted by a visible gap. A straight hyperplane separating two classes is predicted. The essential concentration while drawing the hyperplane is on expanding the separation it bears to the adjoining data point of every other class. A real-time dataset is, for the most part, scattered up to some degree. To take care of this issue, data division into various classes based on a straight and linear hyperplane cannot be viewed as a preferable decision. To address this, Vapnik et al. [134] recommended making non-linear classifiers by incorporating kernel functions to maximize the margins of hyperplanes.

4.3.4 The proposed Algorithm

The process for the data set acquisition and generation remains the same as in KNN+K-Medoids procedure. Thus, the clustering portion is the same while the labeling and classification procedure differs from the former. Since the dataset is real-time and non-linear, there is a tendency of overlapping classes. A kernel trick for SVM must be utilized to reconstruct the data into an elevated dimensional data space to ensure accurate classification. A hybrid or multiclass kernel is incorporated by the application of both Cubic Polynomial and Gaussian kernels. The training data set is first transformed into a higher dimensional data space with the cubic polynomial function given in Equation (6).

$$T \rightarrow \; K(x,y) = \left(\sum_{i=1}^{j} x_i \, y_i + C\right)^n \qquad - \qquad (6)$$

Where *x* and *y* denote vectors in input space, *C less than zero* designates a complimentary parameter that trades off the impact single training example reaches, j denotes the total data points, and *n* symbolizes the dimension of the training data. The features that are extracted in addition to clusters obtained after clustering with K-Medoids provides a form of supervision for obtaining hyperplanes and the number of classes for the classification procedure.

Table 12: SVKM Proposed Hybrid Algorithm

Step 1: Load dataset
Step 2: For each column in the dataset: • Convert contents of string fields with numeric values to numbers • Replace strings without numeric values with NaN
Step 3: Select *Km* as the Medoids for n data points at random.
Step 4: Compute the distance between all data points *n* and the selected Medoids *k* for the closest Medoids.
Step 5: For each duo of data object *j* not chosen and elected data object *p*, calculate total cost to swap, TC*pj*. • If TC*pj* < 0, *p* is substituted by *j*
Step 6: Re-run steps 4-5 until a convergence is reached if and there is no change in the assignment process.
Step 7: Apply Silhouette to obtain fine clusters with its outliers. Quantity of clusters *k* and feature labels obtained serves as an input for the labelling.
Step 8: Transform training dataset *T* into higher dimensional space, ∀ *(x, y) ∈T* $T \rightarrow K(x,y) = \left(\sum_{i=1}^{j} x_i \, y_i + C\right)^{n}$
Step 9: Discover the appropriate hyperplanes *h,* and points closest to hyperplanes *sv* (support vectors), from all classes, C
Step 10: Compute d (*h*, *sv*), ∀ *(x, y) ∈ T* where d is the distance using the Gaussian kernel function
Step 11: Select optimal hyperplane for which the margin is maximized to classify and assign data points to appropriate class, C.

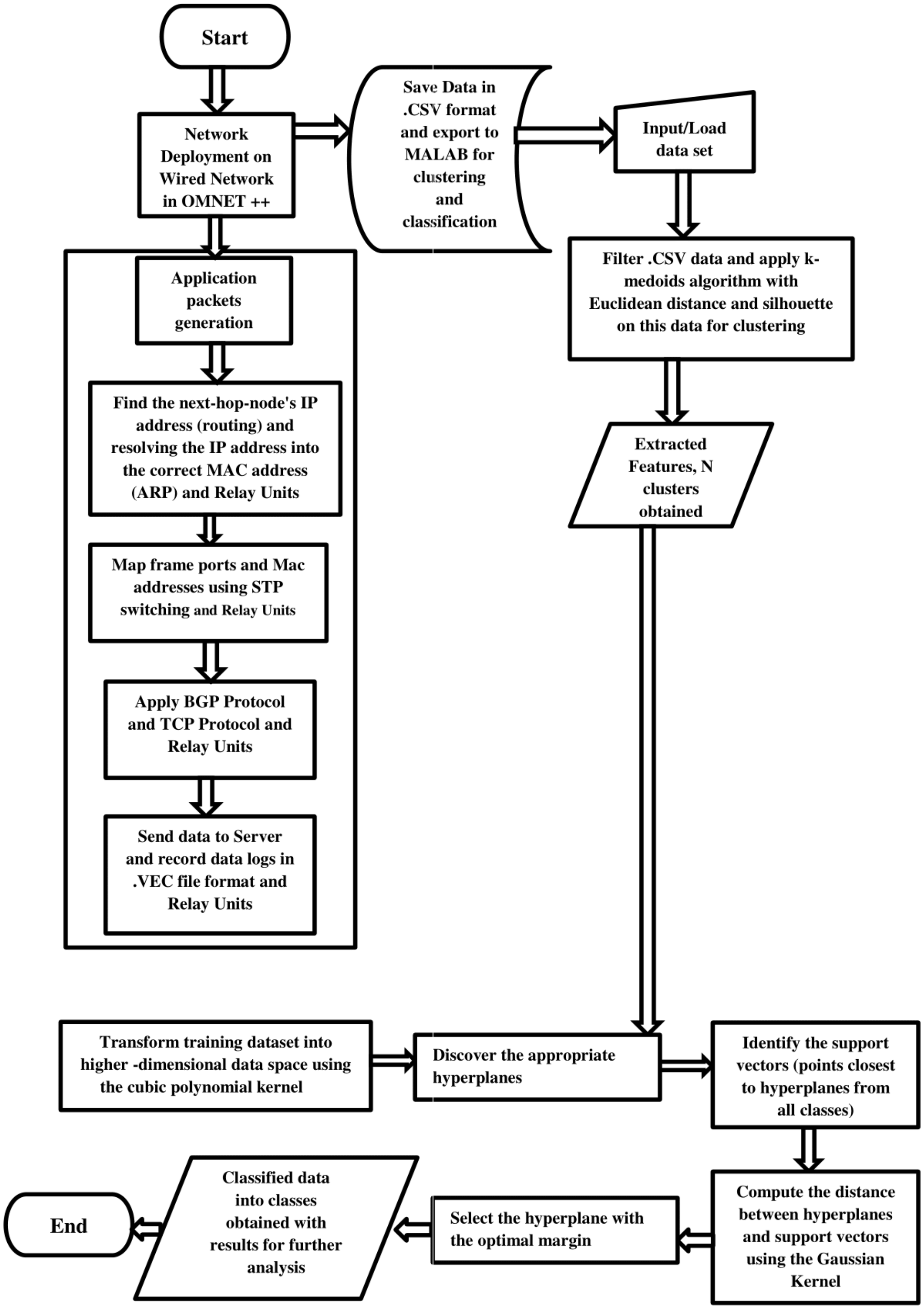

Start
Network Deployment on Wired Network in OMNET ++
Save Data in .CSV format and export to MALAB for clustering and classification
Input/Load data set
Application packets generation
Find the next-hop-node's IP address (routing) and resolving the IP address into the correct MAC address (ARP) and Relay Units
Map frame ports and Mac addresses using STP switching and Relay Units
Apply BGP Protocol and TCP Protocol and Relay Units
Send data to Server and record data logs in .VEC file format and Relay Units
Filter .CSV data and apply k-medoids algorithm with Euclidean distance and silhouette on this data for clustering
Extracted Features, N clusters obtained
Transform training dataset into higher -dimensional data space using the cubic polynomial kernel
Discover the appropriate hyperplanes
Identify the support vectors (points closest to hyperplanes from all classes)
Compute the distance between hyperplanes and support vectors using the Gaussian Kernel
Select the hyperplane with the optimal margin
Classified data into classes obtained with results for further analysis
End

Figure 23: Overall Flow of SVKM Design and Implementation

The clustering resulted in 5 clusters. Hence, the our polynomial function will have a dimension of degree 5. The points closest to the hyperplanes (lines) also described as support vectors, among all the classes are fetched. The extent of separation from the lines to the support vectors is computed using a Gaussian kernel function given in equation (7).

$$K\ (x, y) = \exp(-\ \frac{\|x-y\|^2}{2\sigma^2}) \qquad - \qquad (7)$$

Where $\|x - y\|^2$ denotes the squared Euclidean distance. The line for which the margin is maximized is selected as the optimal hyperplane. The optimal hyperplane classifies or the data points into their appropriate classes.

4.3.5 Complexity of SVKM

The first phase of SVKM algorithm runs with linear time. However, the second phase takes calls from cubic polynomial functions. Therefore the time it takes for the algorithm to completely execute is directly proportional to the cube of the input size. The above complexity of SVKM lies between *$O(n^2)$* and *$O\ (n^3)$*, with the highest being *$O(n^{3.}\ )$*.

4.3.6 Results of SVKM (FPL 1)

The classifier is evaluated with metrics of precision, accuracy, area under ROC, error rates and time complexity with processing time. In terms of precision, the proposed achieved values ranging from 85% to 94% as depicted in Figure 24. A confusion matrix graph is generated after classification as shown in Figure 25. The proposed also predicted 20 classes. However, 14 classes were predicted correctly and 6 classes misclassified. For P2P classification, traffic belonging to Skype class is misclassified to Edonkey class. Traffic for Gnutella class is predicted wrongly assigned to bit torrent. Although YouTube class is predicted correctly, Netflix class is wrongly assigned to FTP class. Mail_Imap is misclassified as Mail_smtp while Direct Link downloads are assigned to Instant Messages. The misclassification of the classifier at most instances bears no similarities with respect to its properties. However, features like packet length and size could have some similarities which may lead to wrong predictions.

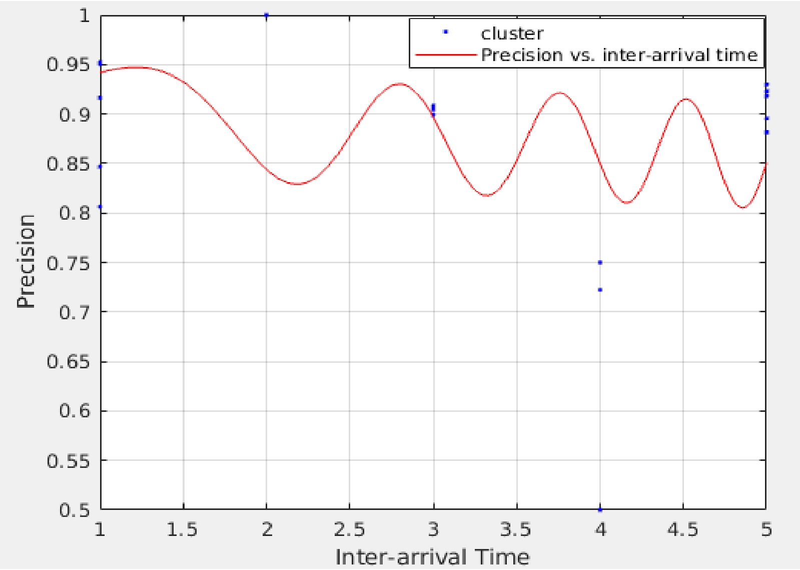


Figure 24: Precision Sine Wave Curve for SVKM Algorithm

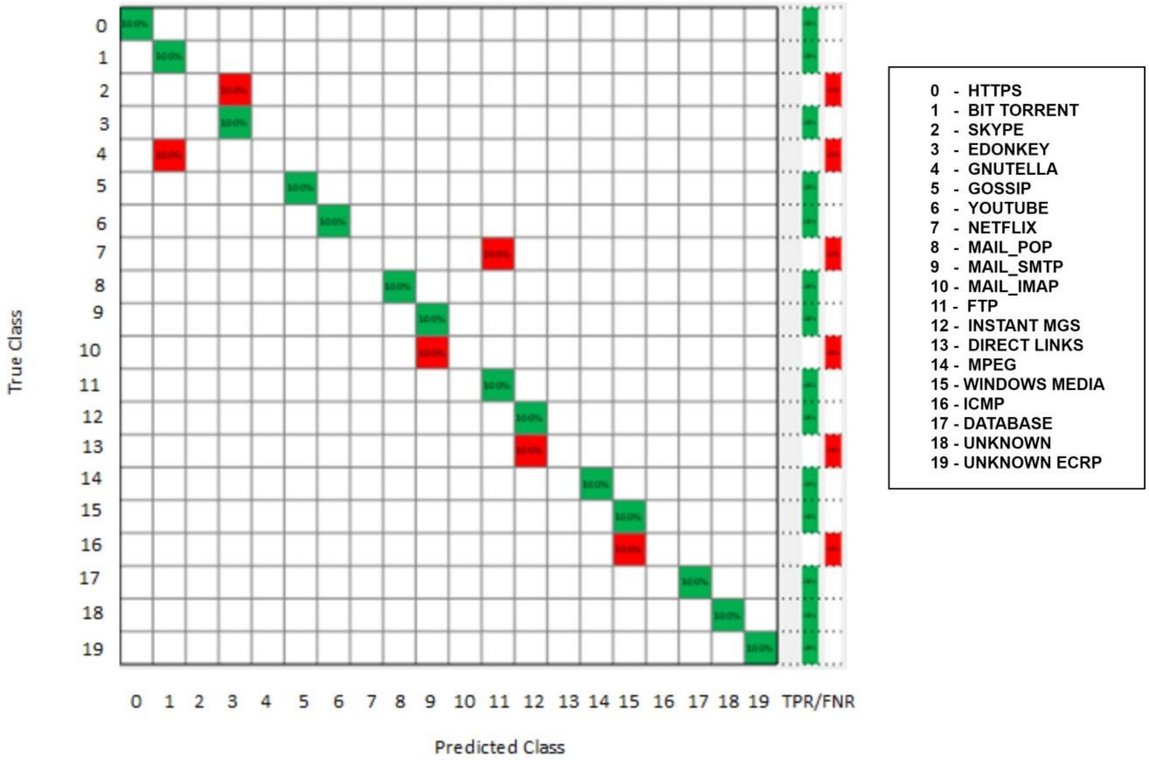


Figure 25: Confusion Matrix of Accuracy Showing True Class against Predicted Class

The overall accuracy achieved after classification is 92.4%, showing an increased margin of 1.1%. The area covered under the ROC curve for the proposed is 1.0 validating that the classifier is efficient for making accurate predictions as shown in Figure 26. The error percentage obtained is 7.6%. In terms of processing time, the classifier utilized 2.9839 seconds showing a decrease of 1.5643 seconds from the time utilized by KNN+K-Medoids.

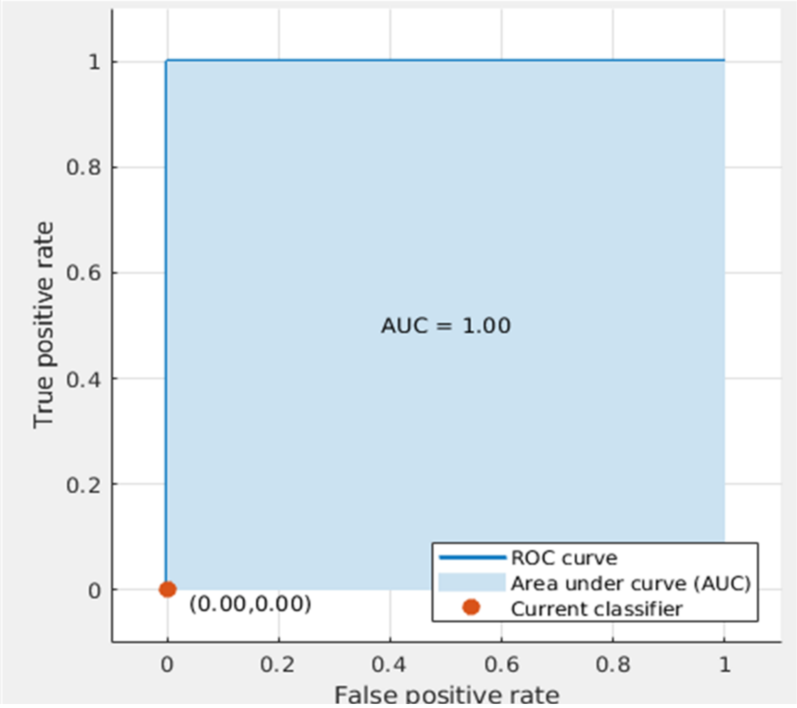


Figure 26: Area Under ROC for SVKM Classifier

4.3.7 Results on SVKM (FPL 2)

The proposed algorithm on the wireless dataset generated similarly revealed 20 classes. 7 out of the 20 classes were misclassified and 13 classes were predicted accurately. Misclassification of application traffic like skype, Mail_imap, windows media, and instant messages occured. However, classification of unknown traffic and encrypted traffic was accurately predicted. The overall classification accuracy achieved is 91.94%

as shown in Figure 27 with error rate of 8.6% incurred. The accuracy achieved compared to the accuracy from the wired environment shows a decline or decrease of 0.86%.

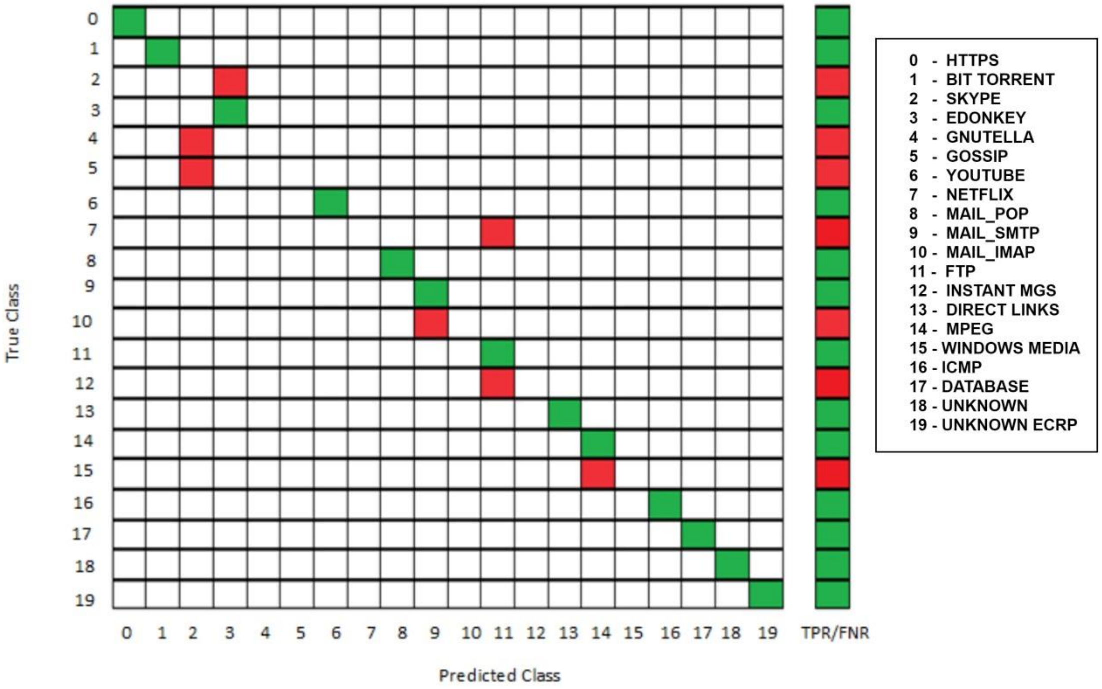


Figure 27: Confusion Matrix Graph Showing the True Class against Predicted Class in Wireless Environment (SVKM)

4.4 Real-Time Application-Based Clustering (R-TAC)

4.4.1 Motivation of Algorithm

The previously proposed hybrid algorithms resulted in good accuracy in classification considering the parameters and circumstances of the network under study. However, there is a need for improvement. Also, the complexity of SVKM algorithm though fair needs to be reduced since the network is already experiencing the resource starvation. To address these issues, Real-time application-based clustering is proposed.

4.4.2 The Proposed Algorithm

R-TAC is a semi-supervised clustering approach with two phases namely clustering and classification. From the classification (label) phase of SVKM, we introduced the transformation of the testing dataset into a higher dimensional space to fit the real-time data from the simulation. A true reflection of the dataset should be implemented both in both clustering and labelling phase. R-TAC initial step is to execute this task using the quadratic function in equation (8) to transform the dataset T, into a two-dimensional feature space T '. The function takes vectors *x and y* as input representing the data points in the two-dimensional space and n denotes the sum of all data points.

$$K(x, y) = \sum_{i=1}^{n}(x_i\, y_i\, + 1)^2 \quad - \qquad (8)$$

The clustering phase sets off by first choosing at random pivotal points C_p within the data space to represent the centers. These points are assumed centers for the cluster. The distance between the data points and the centers are computed with the Minkowski distance metric in equation (9). This metric gives as a parameter *p*, with which the order of the distance between two data points can be set which is an advantage to decrease the errors incurred during distance calculation. When *p* is assigned the value of 2, the weighted distance can be obtained which gives more weight or value to data points close to the testing data point. Data points are assigned to the cluster C with minimum distance. The pivotal points for the clusters are rediscovered by finding the point within the cluster, which has an average distance to all other points utilizing the same distance metric. The process is repeated until there are no further cluster assignments.

$$d\ = (\sum_{i=1}^{n}((x_i\, y_i)^{\mathrm{p}})^{1/\mathrm{p}} \quad - \qquad (9)$$

At this stage, the clusters are likely to overlap each other due to overlapping features. The second phase classifies the data points into finer classes or clusters. For all data objects belonging to two or more clusters, the distance to the respective cluster centers is computed utilizing the Gaussian function in equation (10). The data point is assigned to the cluster with the minimum distance. However, the C_p for all C will have to be recomputed and the process repeated till no overlapping clusters are formed. Table 13 illustrates the pseudo-code for the algorithm. The flow of the algorithm is represented in Figure 28.

$$K\ (x, y) = \exp(-\ \frac{\|x-y\|^2}{2\sigma^2})\ \ -\qquad (11)$$

Table 13: R-TAC Proposed Algorithm

Step 1: ∀ (x, y) ∈ T transform dataset T to T ' T → $K(x, y) = (\sum_{i=1}^{n}(x_i\ y_i + 1)^{\ 2}$ → T '
Step 2: Select pivotal points C_p within the data space at random
Step 3: ∀ x_i ∈ T' compute the distance *d*, to all C_p $d = (\sum_{i=1}^{n}((x_i\ y_i)^{\ k})^{1/2}$
Step 4: Assign the data objects a *C* with minimum distance and recompute C_p for all C using *d*
Step 5: Repeat steps 3 and 4 until a convergence is reached if and there is no change in the assignment process.
Step 6: ∀ x_i ∈ C ≥ 2, Calculate the $d(x_{i,}\ C_p)$ with the Gaussian function $K\ (x, y\ ) = \exp(-\ \frac{\|x-y\|^2}{2\sigma^2})$, y = C_p
Step 7: Assign x_i to cluster with the minimum *d*
Step 8: Re-compute C_p for all C's and repeat Step 6 and 7 till all C's are distinct.

4.4.3 Complexity of R-TAC

Due to the quadratic functions utilized, the execution time of R-TAC is directly proportional to the square of the input size it takes. The time complexity of R-TAC is initially equivalent to *O(nm)* where *n* is the input size and *m* is the number of clusters. However, the complexity is bounded from the above to $O(n^2)$ with the incorporation of the quadratic functions.

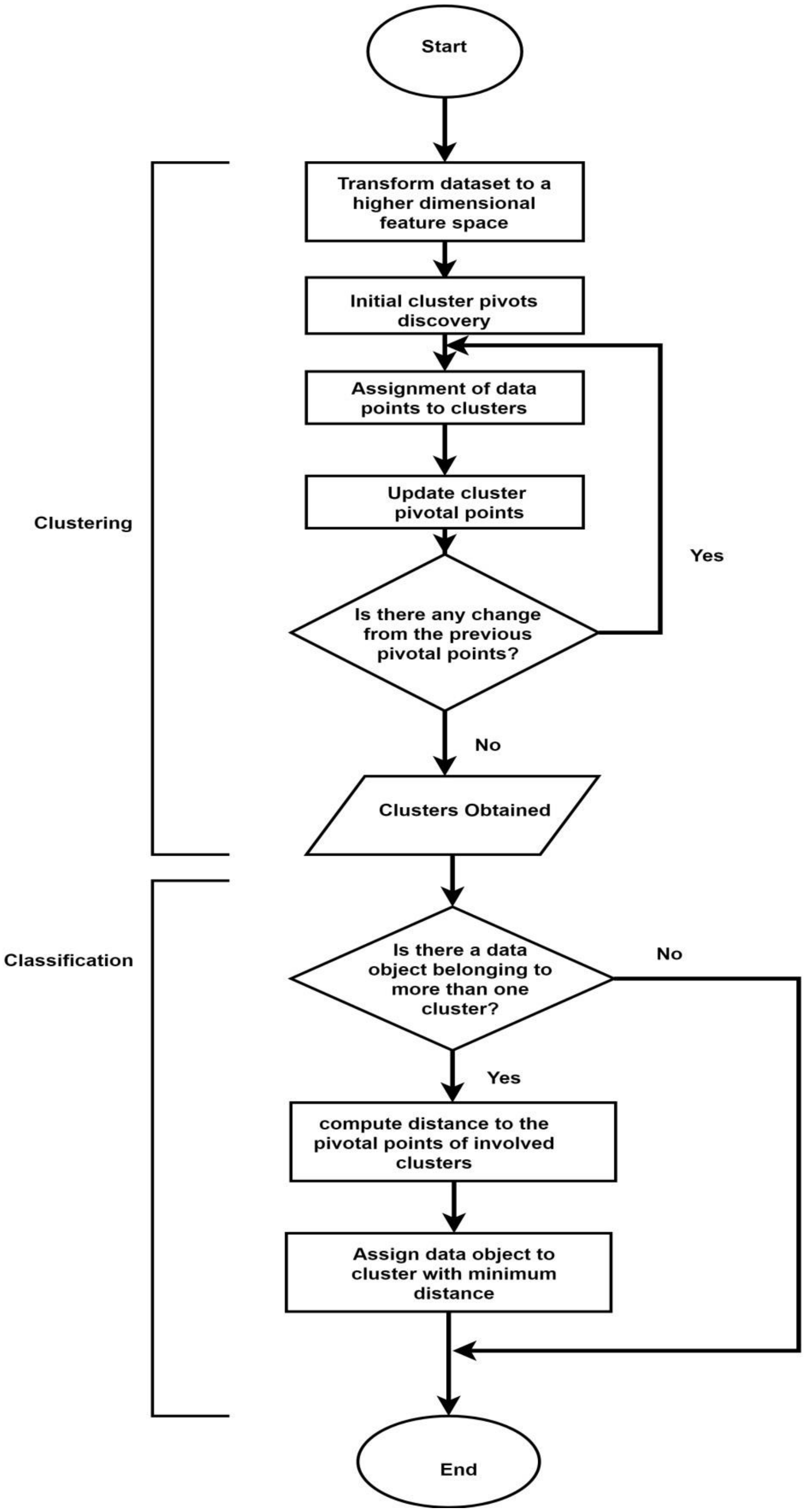
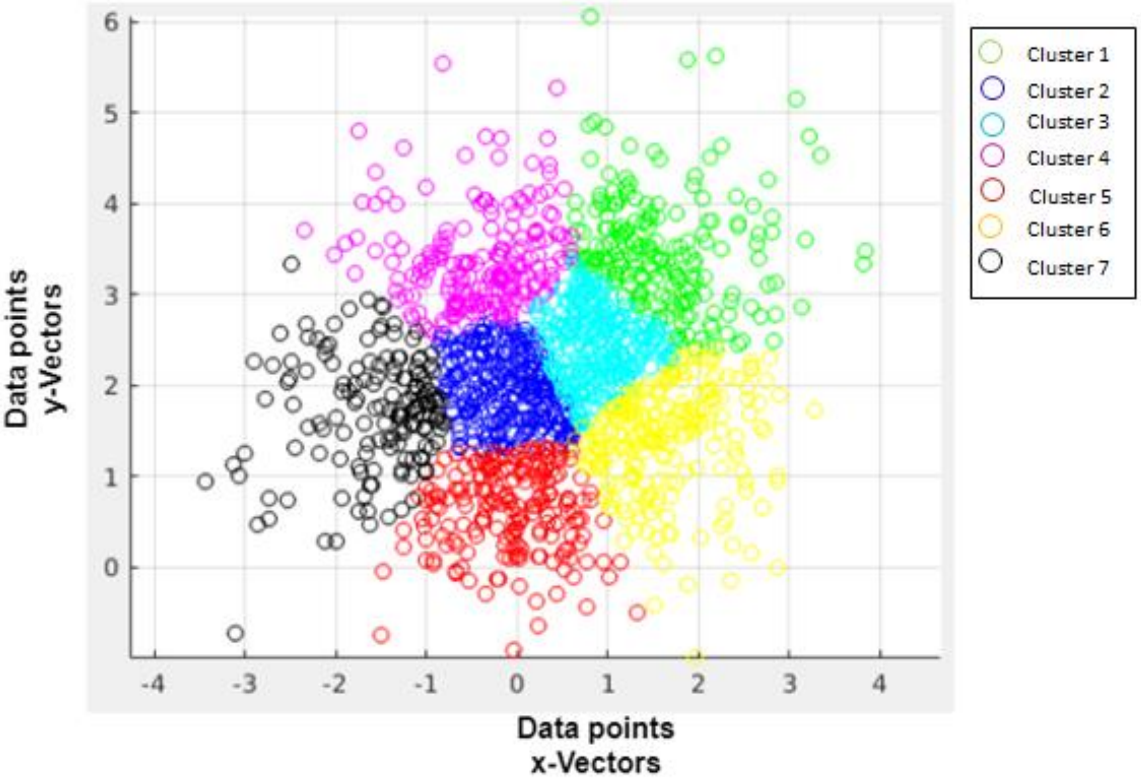

Start
Transform dataset to a higher dimensional feature space
Initial cluster pivots discovery
Assignment of data points to clusters
Update cluster pivotal points
Clustering
Yes
Is there any change from the previous pivotal points?
No
Clusters Obtained
Classification
Is there a data object belonging to more than one cluster?
No
Yes
compute distance to the pivotal points of involved clusters
Assign data object to cluster with minimum distance
End

Figure 28: Flow diagram of R-TAC

4.4.4 Results of R-TAC

The dataset from the proposed scenario is used to test the algorithm. After the clustering phase, 7 overlapping clusters are formed as shown in Figure 29. After the classification procedure was completed, 20 classes are identified. Precision values ranging from the lowest being 85% and highest 99% were obtained for R-TAC as displayed in Figure 30. This range is quite higher compared to KNKM and SVKM.



.

Figure 29: Clusters Revealed after Initial Clustering Phase

In terms of accuracy, R-TAC predicted both mail_pop and mail_imap as mail_smtp. Furthermore, Mpeg and ICMP classes are also predicted as windows media as depicted in Figure 31. These false negative predictions were almost prevalent among all the proposed algorithms. Overall, R-TAC had 16 classes predicted rightly and 4 classes wrongly predicted, resulting in 96.40% accuracy. The area under ROC covered is 1 which further confirms the good prediction propensity of the algorithm, displayed in Figure 32.

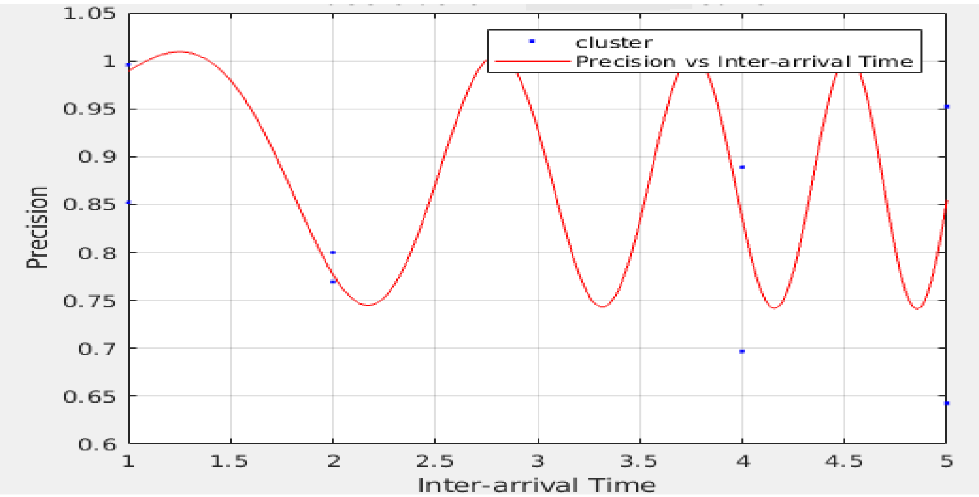


Figure 30: Sine Wave Graph of Precision against Inter-arrival time for R-TAC

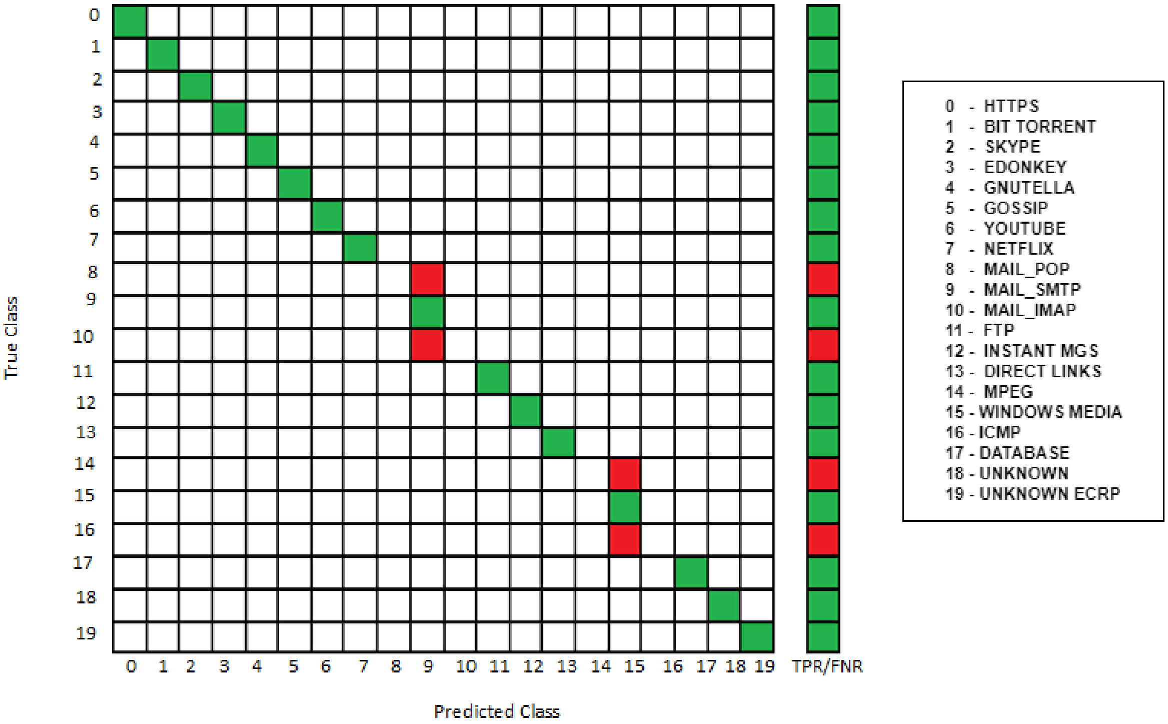


Figure 31: Confusion Matrix Graph Showing True class against Predicted Class (R-TAC)

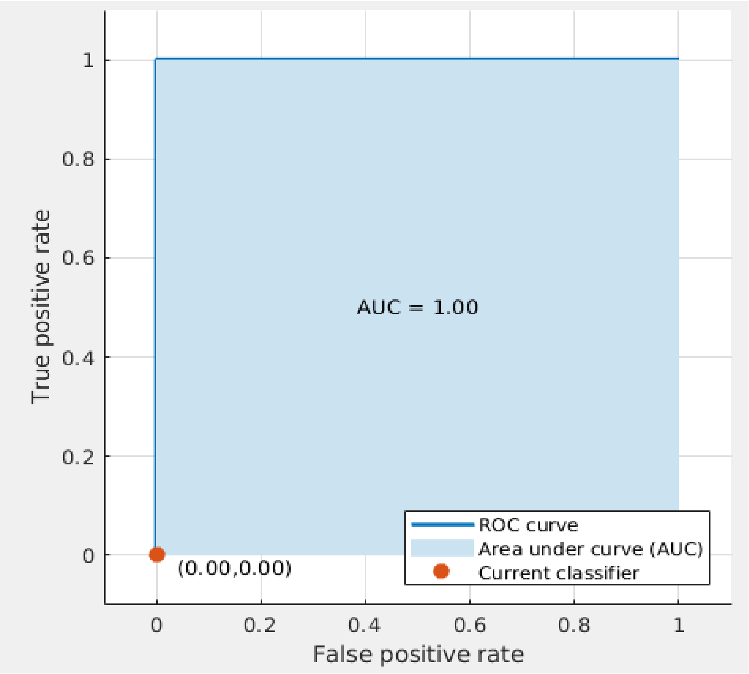


Figure 32: Area Under ROC Covered for Classification with R-TAC

4.4.5 Results of R-TAC on FPL 2

With respect to the dataset gathered from the wireless scenario, R-TAC similarly like KNKM and SVKM also revealed 20 classes. From the graph in Figure 33, 15 classes were accurately predicted with 5 classes misclassified which includes FTP, Mail_imap, Instant Messages, MPEG and Windows Media applications. The overall accuracy achieved for the classification is 95.63% with 4.37 error rates incurred. The accuracy in comparison with the results with FPL 1 from the wired scenario shows a decrease of 0.77% .

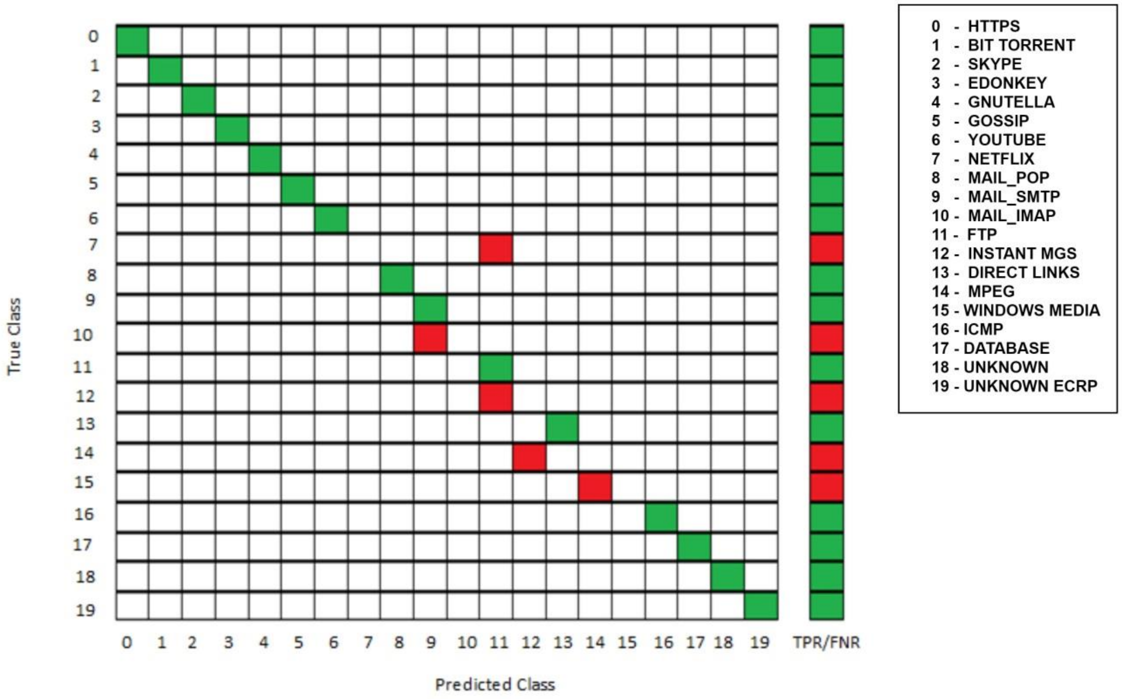


Figure 33 : Confusion Matrix Graph Showing the True Class against Predicted Class in Wireless Environment (R-TAC)

4.5 Summary

The architecture and implementation of our suggested algorithms are discussed in the content of this chapter. The first proposed algorithm, KNN+K-Medoids hybrid algorithm as the name implies constitutes the KNN algorithm and K-Medoids algorithm. The advantages of this primary algorithm are combined in the design. The novelty of the algorithm stems from its hybrid nature. The three phases of the algorithm are data filtration, clustering, and labelling. The dataset is first filtered to eliminate unwanted data fields and convert the required fields into their appropriate formats. In the clustering phase, the distance metric used to in the computation of the medoids is the Euclidean formula for distance in place of the traditional Manhattan formula, because the target is to reduce the separation dissimilarity to its minimum. Also the weighted KNN is used instead of the classical KNN in the labelling phase to enable the classifier to emerge with finer classes. Results after implementation show that the proposed algorithm is achieved a precision range of 82 % to 94%, and able to predict 20 classes out of which 12 were predicted correctly to result in 91.3% accuracy.

The next proposed hybrid SVKM employs the two kernel tricks of the SVM algorithm to transform the dataset to higher dimensional space which is the right format for our dataset. This improves the classification precision by a margin of 3% and accuracy with a margin of 1.1%. However, time complexity higher compared to KNKM and accuracy needs to be improved. This contributed to the development of R-TAC algorithm.

R-TAC algorithm transforms the dataset to a higher dimensional space before clustering and classification. It employs the Minkowski distance metric in all its distance computations. The accuracy of classification is improved to 96.40% with a higher precision range from 75% to 99%. The results of the proposed algorithms are validated in the next trailing chapters to confirm its efficiency.

Chapter 5: Analysis of Results

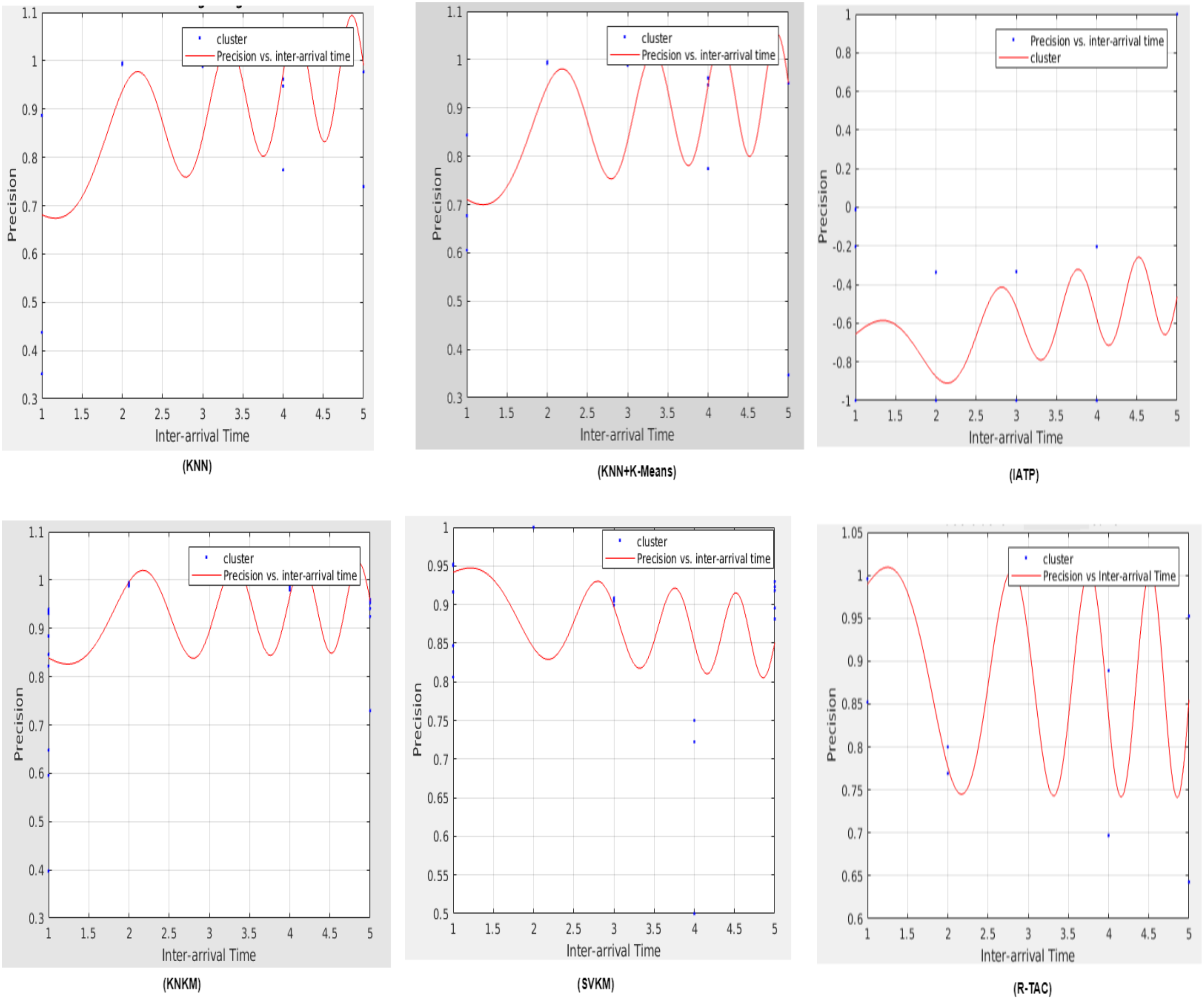
This chapter reviews the outcomes, findings, and conclusions after experiments are conducted employing the evaluation with the classification metrics chosen for the study. The proposed works are further validated by making a comparison with other three existing works namely, KNN [127], KNN+K-Means [72] and IATP [121]. The overall outcomes are further used to draw conclusions for the study.

5.2 Precision

Precision is a prediction of accuracy measure. The precision range for each mechanism is compared in Figure 34. The outcome reveals that KNN achieved a precision range of 62 to 94%, IATP with 58% to 69% and KNN+K-Means resulted in 70% to 93%. The proposed classifiers achieved the overall best with proposed KNKM with 82% to 94%, SVKM classifier with 85% to 94% and R-TAC with the highest precision, ranging from 85% to 99%. Because the higher delimiter of the precision range of all classifiers is almost the same, the lower delimiter is used to determine the best of all the classifiers in terms of precision. We can affirm from the outcome that the proposed model is likely to perform exceptionally and give a high accuracy prediction value.

5.3 Accuracy

In terms of accuracy in classification, the classifiers are evaluated by a confusion matrix graph to show the total number of classes, true classes and predicted classes. From Figure 35, the proposed models together with KNN+K-Means resulted in 20 classes while KNN and IATP resulted in 14 classes. KNN and IATP could not distinguish further between the P2P traffic and classified all unidentified traffic such as database and encrypted traffic as unknown traffic. This shows the former three classifiers produced finer classes. Out of the 14 classes, KNN classified 3 classes correctly (True Positive Rate (TPR)) while 6 classes were wrongly classified (False Positive Rate (FPR)). Due to



(KNN)

(KNN+K-Means)

(IATP)

(KNKM)

(SVKM)

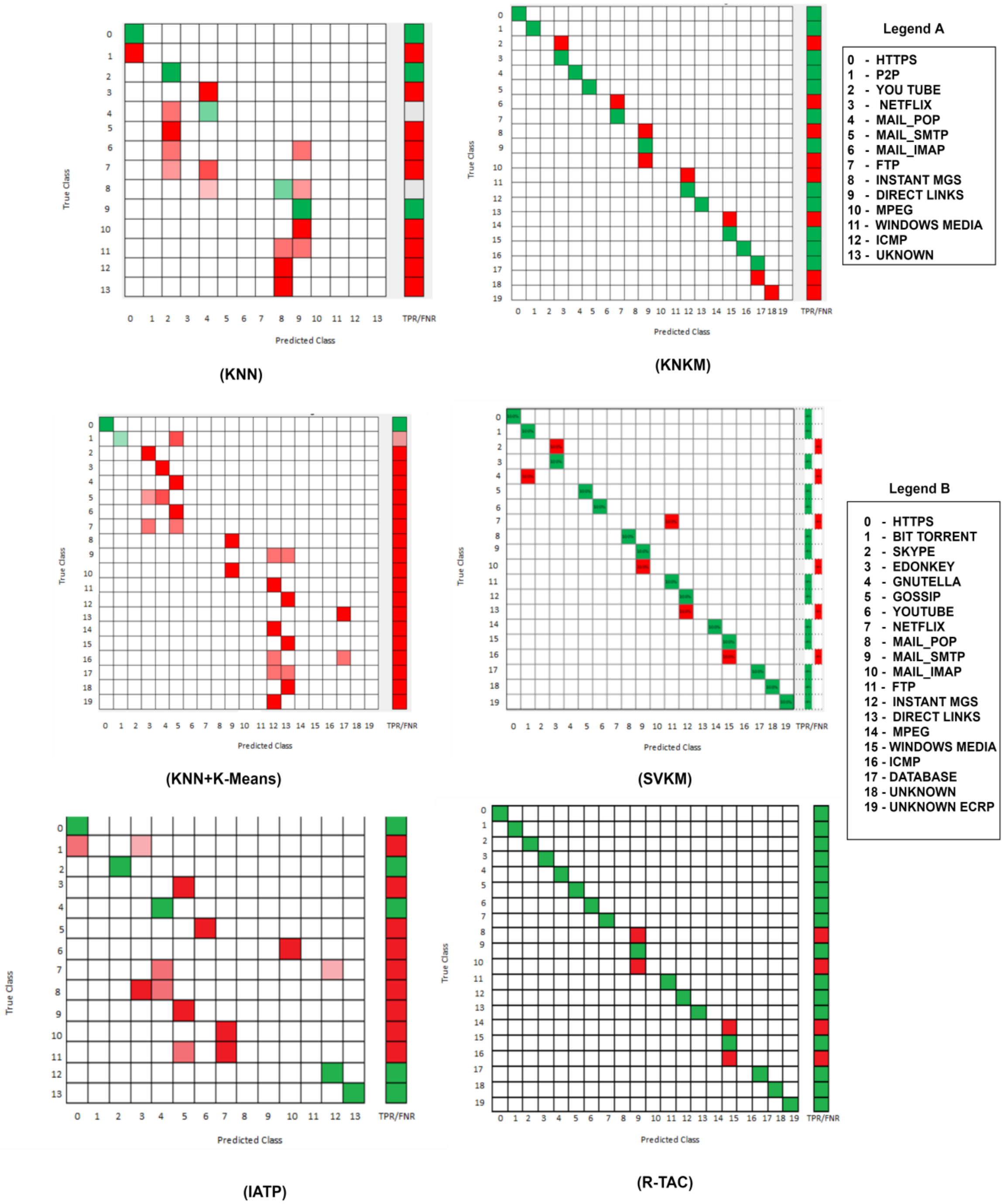
(R-TAC)

Figure 34: Comparative Graph of achieved Precision Range for all Classifiers

overlapping features, 5 classes were predicted or classified into different classes giving an accuracy of 73.79%. KNN+K-Means predicted 1 class perfectly and 6 classes were predicted into different classes. 13 classes were completely misclassified out of the total of 20 classes identified resulting in an accuracy of 65. 2%. IATP predicted 5 classes correctly and 9 classes wrongly with 4 classes due to overlapping flow features. IATP resulted in 66.5% average accuracy. The proposed KNKM had 12 classes predicted correctly and 8 classes misclassified giving an accuracy of 91.3%. SVKM predicted 16 classes correctly out of 20 classes and 4 classes were misclassified resulting in an accuracy of 92.4%. R-TAC had 16 classes predicted correctly and 4 classes misclassified resulting in an accuracy of 96.40%. It can further be inferred from the results that the proposed approaches do not have an instance of predicting a class into multiple classes. This shows that the proposed methods are able to accurately distinguish between features to avoid the problem of overlapping features in classification resulting in higher accuracy values. Also, all accuracy values achieved falls into the precision range of each classifier respectively validating the accuracy values obtained. From the above revelations from the outcomes, it can be confirmed that the suggested methods perform better in classifying packet flows correctly compared to the existing renowned works in literature in presence of extreme packet loss and fragmentation or wired networks with limited resources.

5.4 Error Rates

Error rate is the amount of misclassification allowed by each classifier, thus the inverse of accuracy values obtained. After classification KNN had an error rate of 26.21%, KNN+K-Means achieved 34.8% and IATP with 33.5%. KNKM resulted in 8.7% and SVM+K-Medoids with 7.6% and R-TAC had the lowest with 3.6 error rate. The lower error rates of the proposed methods are attributed to the labelling method adopted for each algorithm. A chart of accuracy with error rate for each classifier is shown in Figure 36.



(KNN)

(KNKM)

(KNN+K-Means)

(SVKM)

(IATP)

(R-TAC)

Figure 35: Comparative Graph of Confusion Matrix Showing Classified and Misclassified classes for Each Classifier

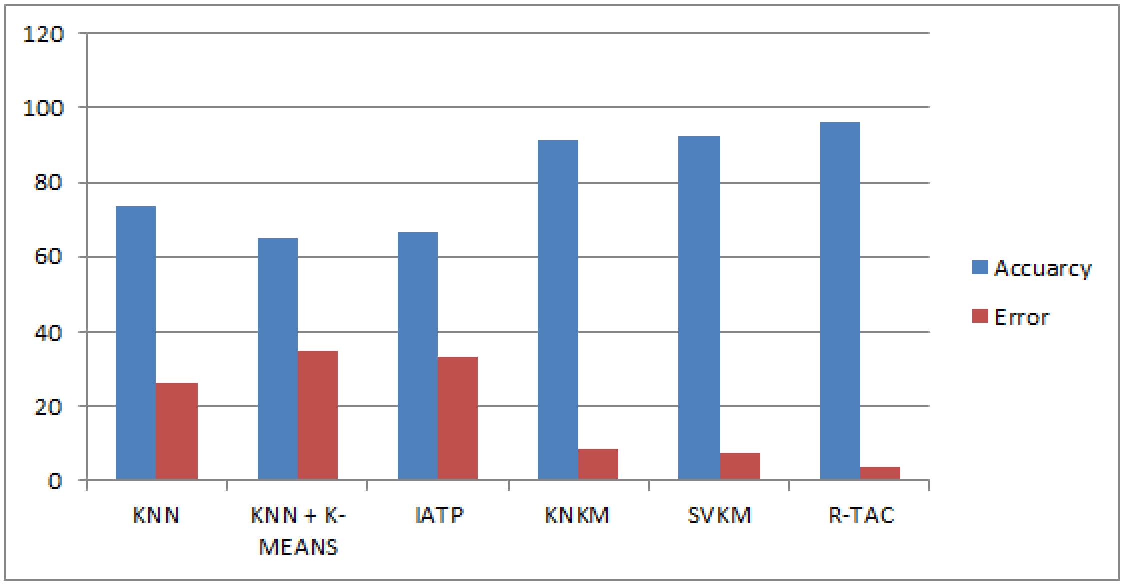


Figure 36: Comparative Graph of achieved Accuracy with Error Rates for Each Classifier

5.5 Area under ROC

ROC curve validates the classifier's ability to make good classification predictions. The greater the proximity of the curve is to the leftmost axis in a range of 0 to 1, the better its performance. The results from Figure 37 depicts that AUC for KNN is 0.9, KNN+K-Means is 0.87 and IATP with 0.5. The proposed models (KNKM, SVKM, and R-TAC) have an AUC of 1.0. This shows that though all the classifiers have good prediction property, the proposed models have the best capacity to classify traffic flows in scenarios of networks with increased packet drop count, or where congestion is likely to occur.

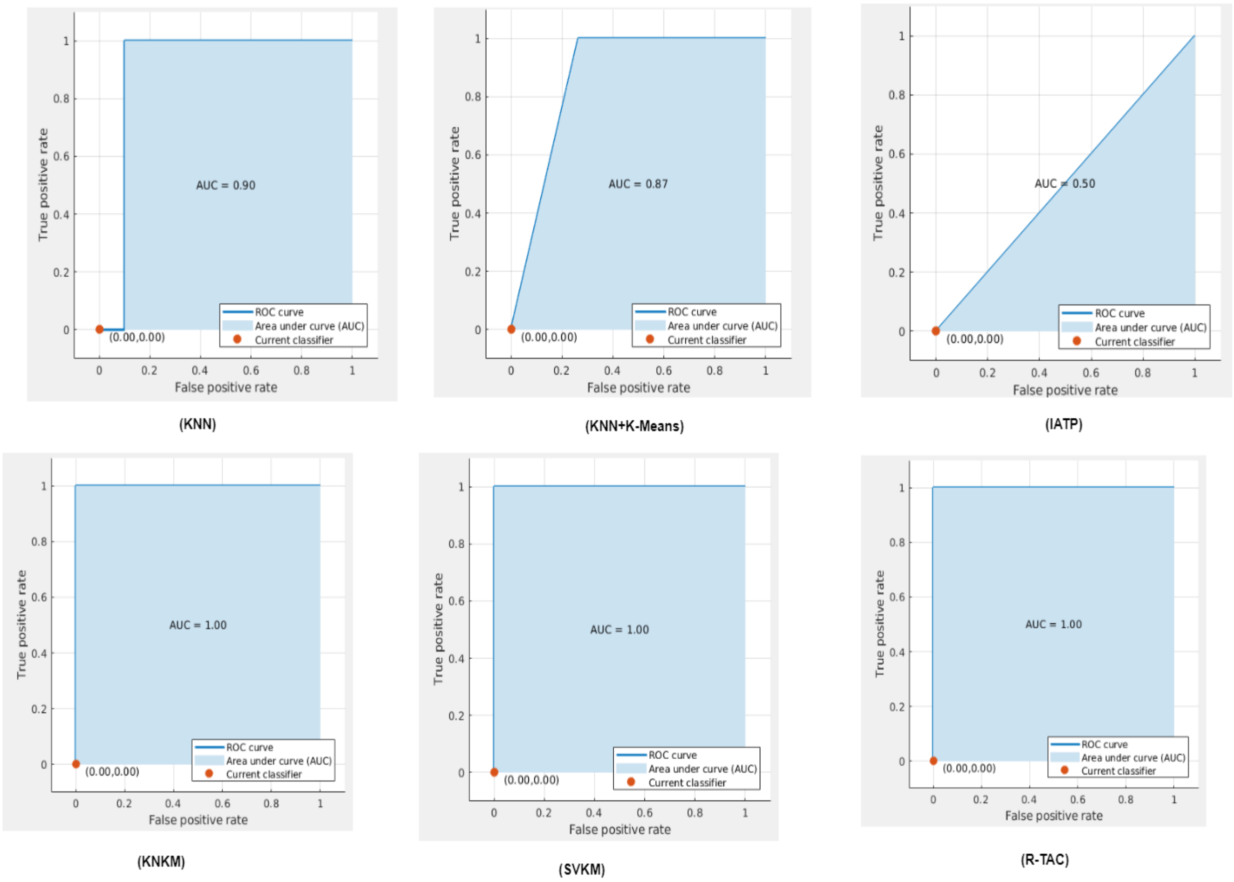


Figure 37: Comparative Graph of ROC Curves Showing the Area Under ROC for Each Classifier

5.6 Processing Time

The time it takes for each classifier to complete the entire process is significant in evaluation. The longer the processing the (time complexity), more costs are incurred in terms of resource usage. The complexity of the algorithm affects the overall processing time of the algorithm. KNN classifier utilized 1.0767 seconds and KNN+K-Means used 4.1804 seconds. IATP utilized 3.2521 seconds. It took 4.5482 seconds for KNKM classification, SVKM utilized 2.9839 seconds and R-TAC with 1.9983 seconds. The least time utilized by the KNN classifier can be attributed to the fact that less number of instructions is executed in the algorithm. Out of the other hybrid methods, SVKM took the least time to process. This is due to the advantage of the kernel trick employed in the classification processes. The real-time or non-linear dataset transformation into higher dimensional space limits the time it takes for the classes to be discovered and data points to be grouped into their respective classes. The comparative rundown of the analyzed results is depicted in Table 14.

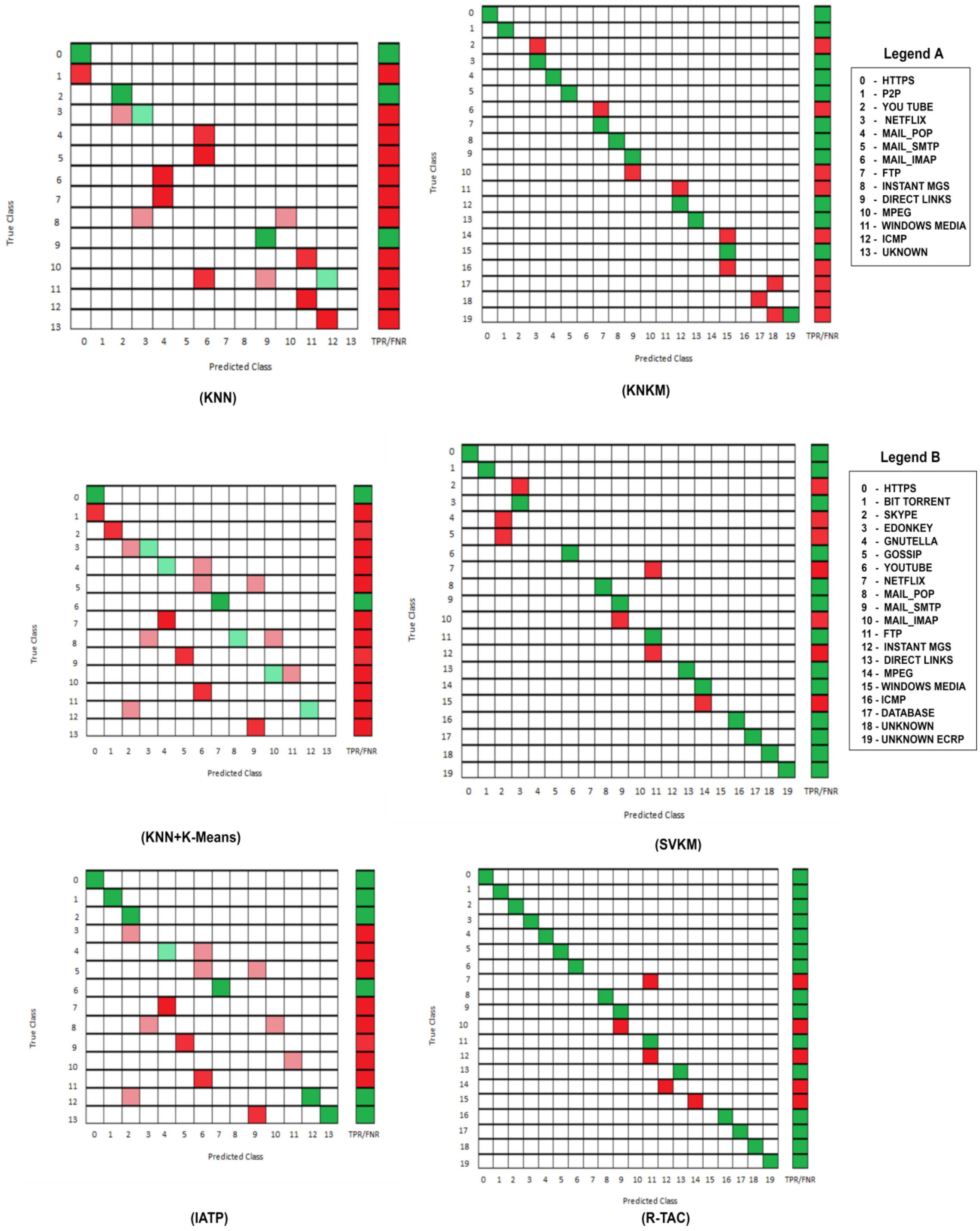
Table 14: Comparative Summary of Attained Outcomes for Each Classifier with FPL 1 Dataset

CLASSIFIER	**CLASSIFICATION METRIC (Parameter)**				
	Precision Range (%)	**Accuracy (%)**	**Error (%)**	**AUC**	**Time (s)**
KNN	62 - 94	73.79	26.21	0.9	1.0767 *O(nd)*
KNN+K-MEANS	70 -93	65.20	34.80	0.87	4.1804 *O(n)*
IATP	58.69	66.50	33.50	0.5	3.2521 *O(n*log n)*
KNKM	82-94	91.30	8.70	1.0	4.5482 $O(k(n-k)^2$
SVKM	85-94	92.40	7.60	1.0	2.9839 $O(n^3)$
R-TAC	85.94	96.40	3.6	1.0	1.9983 $O(n^2)$

5.7 Analysis of Results from Wired Scenario (FPL 1) in comparison to Wireless Scenario (FPL 2)

The existing algorithms are also tested on the dataset derived from the wireless scenario and compared with the proposed works in terms of classification accuracy. From Figure 38, All existing classifiers revealed 14 classes with the proposed classifiers revealing 20 classes. The issue of overlapping or multiple classifications persisted with respect to the existing algorithms.

On the other hand, the proposed works in all cases classified application traffic distinctively. KNN achieved a classification accuracy of 70.45%, which is a reduction of 3.34% in classification accuracy. KNN+K-Means resulted in 68.70% classification accuracy with IATP achieving 72.50% accuracy. Both KNN+K-Means and IATP classifier improved in classification accuracy of 3.5% and 6.0% respectively. This asserts the fact that the latter two existing algorithms have better propensities in working in wireless environments when the problem of some resource starvation exists. On the hand the existing works, though experienced a decline in classification accuracy in wireless environments, the decline rate is minimal and exceeds the classification accuracy of all the existing classifiers. We can therefore conclude that the proposed works can work efficiently in the both wired and wireless environments where there is some form of resource starvation where the speed of links are not able to meet the requirements of the network leading to extreme packet loss and fragmentation. A comparison of classification accuracy in both the proposed wired and wireless environments is depicted in Table 15.



(KNN)

(KNKM)

(KNN+K-Means)

(SVKM)

(IATP)

(R-TAC)

Figure 38: Confusion Matrix of Classification of Classifiers in Wireless Scenario (FPL 2)

Table 15: Comparison of Classification Accuracy (Wired Scenario Vs. Wireless Scenario)

CLASSIFIER	CLASSIFICATION ACCURACY (%)	
	Wired Scenario (FPL 1)	Wireless Scenario (FPL 2)
KNN	73.79	70.45
KNN+K-MEANS	65.20	68.70
IATP	66.50	72.50
KNKM	91.30	90.86
SVKM	92.40	91.54
R-TAC	96.40	95.63

5.8 Summary

This chapter contributes to the third objective of the study by validating the proposed works. We analyze the finding of the works and compare with existing works using the proposed dataset from the proposed wired and the wireless environments. The proposed works revealed finer classes compared to the existing ones and have higher accuracy and precision values. The results show that the proposed algorithms implemented in MATLAB as classifiers are better models in application and protocol classification than the already existing models in literature in all cases of the proposed scenarios.

Chapter 6: Validation of Proposed Algorithms

To further ensure that the proposed approaches are efficient as discussed prior to this chapter, we test the algorithms with existing datasets to unravel its classification efficiency. Some existing datasets in relation to traffic analysis and classification are discussed in this section. As at the time of the conduction of this research, none of the datasets collected or used in literature has considered the parameter of limited resources experienced by networks. The study aims to provide a solution in the phase of a network cycle where such limitations occur. However, to prove that the proposed work can efficiently, we validate using the most closely related dataset with respect to the parameters considered by our study.

6.1 Datasets for Traffic Classification

With respect to datasets for traffic analysis and classification on the internet, enterprise and organizational networks, several datasets are in existence. CAIDA has a collection of these datasets and updates them from time to time. The environment for the collection of the datasets discussed originated from the University of California at San Diego Academic & Science (UCSD). The datasets include

- Historical and Near-Real-Time UCSD Network Telescope Traffic Dataset [135] [136]
- The CAIDA Anonymized Internet Traces Dataset 2008 [137]
- Statistical information for the CAIDA Anonymized Internet Traces [138]
- UCSD Network Telescope Educational Dataset [139][140]
- OC48 Peering Point Traces [141]
- ITA Network Traces [142]
- Waikato Internet Traffic Storage (WITS) Data Catalogue [143]

- Cup KDD 1999 [144]
- Mid-Atlantic CCDC dataset [145]

Two datasets in literature are selected namely Cup KDD’99 and MACCDC IDS trace for the validation of the proposed works. The results are compared in terms of classification overall accuracy with the three existing works as well and discussions are made to that effect.

6.1.1 Historical and Near-Real-Time UCSD Network Telescope Traffic Dataset

The dataset is made of near to real-time raw traces of traffic captured on UCSD with a telescope instrument. The dataset comes in the form of highly compressed packet capture (pcap) files collected over a running time of an hour. The dataset contains some worms in the traffic efficient for detection of spoofing related denial of service attacks from a source.

6.1.2 The CAIDA Anonymized Internet Traces Dataset 2008

This is an ongoing dataset which has been collected since 2008. It contains passive traffic traces captured from monitors with high speed installed on the backbone links of commercial networks. CryptoPan prefix is employed to protect the anonymity of traffic traces. Only header information is kept in the captured snap which varies from 64 to 94 bytes in length. This is to help avoid high rates of packet loss. It is useful for research in scopes covering security issues, attributes of traffic from the internet, duration of flows, application disseminations and distributions related to geography and topology.

6.1.3 Statistical information for the CAIDA Anonymized Internet Traces

From various backbone links of OC 192, monthly traces of traffic were gathered routinely, each in an hour. With parameters of size and costs incurred for storage, traces were captured quarterly (three month period). The dataset consists of statistical information, which includes characteristics of protocols and flows. Among these includes start and stop time of flows, number of packets, packets unknown, rates of transmission, and packet size.

6.1.4 UCSD Network Telescope Educational Dataset

The dataset comprises of analyzed unidirectional IP Traffic to dark space and outlines various methods to analyze traffic generated with the internet protocol for IP addresses which have not been assigned yet (dark space). Network addresses for all destinations are masked with the initial 8bits zeroed. Source addresses are also protected by anonymizing with a cryptopan single key. The data captures are traces from the year 2012, in which the raw data in pcap format.

6.1.5 OC48 Peering Point Traces

The OC48 dataset contains passive traces of network traffic gathered in 2002 and 2003 from OC48 peer links within a large ISP network. Any software that can read files in pcap and tcpdump file formats such as Wireshark, tcpdump can be adopted to read them. Its application is similar to CAIDA Anonymized Internet Traces Dataset 2008 and can be used for research related to security, application breakdowns, and internet traffic analysis.

6.1.6 ITA Network Traces

From the Internet Traffic Analysis (ITA) organization, this dataset of network trace is useful for study and research related to dynamics of networks, patterns in network growth, end-user usage statistics of usage and simulations inspired by trace.

6.1.7 Waikato Internet Traffic Storage (WITS)

WITS is a compilation of Internet traffic datasets generated at the University of Auckland, University of Waikato and selected Indianapolis and New Zealand ISP networks for research purposes. These datasets of network traces were captured in different years. Versions of the datasets include Auckland I - Auckland X, IPLS I - IPLS III, Waikato I – Waikato VIII and NZIS I – NZIS II.

6.1.8 Cup KDD 1999

The cup KDD dataset was used for the competition organized by KDD conference to develop a predictive model (detector) for Intrusion detection in the field of data mining. The goal of the model is to determine whether a connection is bad (attacks) or good (no attacks/ normal). The dataset was generated in a military environment which depicts a wired or wireless environment and contains numerous amount of intrusions infused into it.

6.1.9 IDS Traces (Mid-Atlantic CDCC)

The dataset consists of traces from Intrusion Detection gathered from System National Cyber Watch Mid-Atlantic Collegiate Cyber Defense Competition (MACCDC). It comes in the form of captured packet traces collected from 2010 to 2012. This public dataset of traces serves as ground truth for the testing of IDS and traffic classifiers in different network environments. The traces consist of traces of traffic from both cryptographic and non-cryptographic application protocols used for the transmission of traffic from source to destination.

6.2 Validation of Algorithms with CUP KDD'99 Dataset

6.2.1 Results of Proposed Algorithms

All the classifiers except KNKM revealed 7 distinct classes labelled by the respective application protocol type. KNKM resulted in 6 distinct classes. The class types are Hypertext Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), Post Office Protocol version 3(POP3), Edonkey, Bit Torrent (BT), I Seek You (ICQ) VoIP application and Real-Time Transport Protocol (RTP). The results in terms of accuracy prediction for all classifiers are discussed and analyzed for conclusions.

KNKM predicted 4 classes perfectly (SMTP, POP3, BT, ICQ,) against the true class as depicted in Figure 39. However, class HTTP had half of the total flows predicted as SMTP. Edonkey traffic is also predicted wrongly as ICQ traffic. The overall average accuracy achieved by the KNKM classifier is 96.76%.

From Figure 40, SVKM had 5 classes (HTTP, SMTP, POP3, Edonkey, ICQ) rightly predicted against the true class. Class BT is predicted wrongly into class ICQ. In addition, RTP class is predicted into Edonkey class. The overall average accuracy achieved is 97.64%.

Out of the 7 classes revealed for the true class in Figure 41, R-TAC predicted 6 classes correctly. Only the BT class was predicted wrongly into ICQ class. The overall average accuracy achieved by R-TAC algorithm on the dataset is 98.95%.

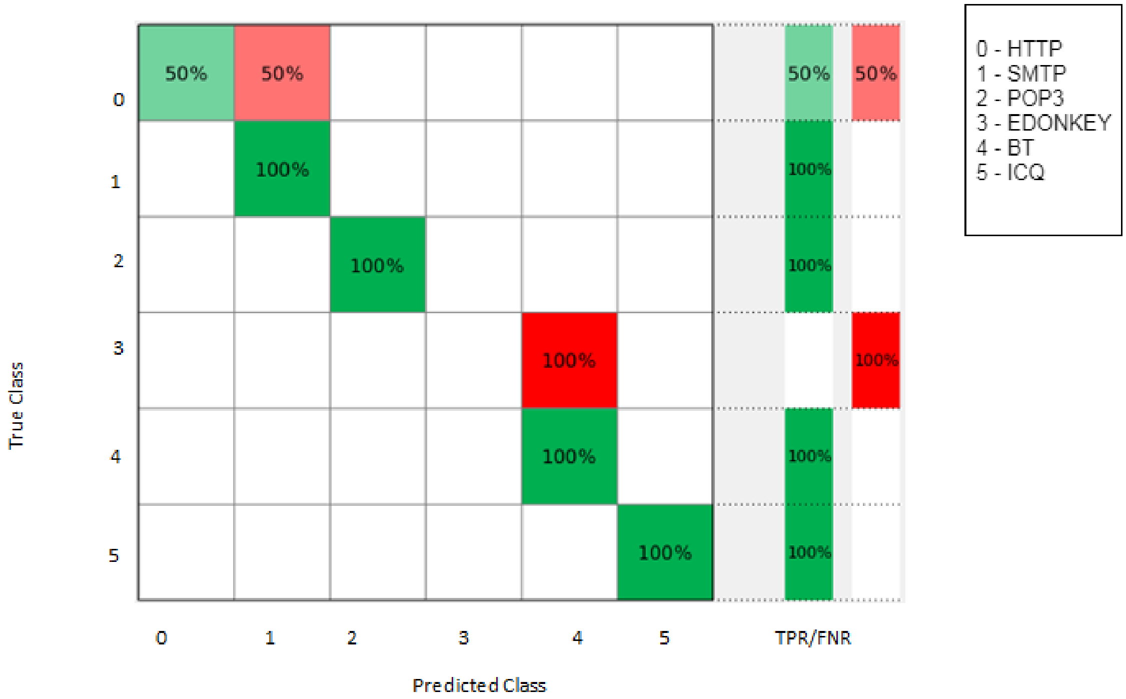


Figure 39: Confusion Matrix Graph of Proposed KNKM on Cup KDD’99 Dataset

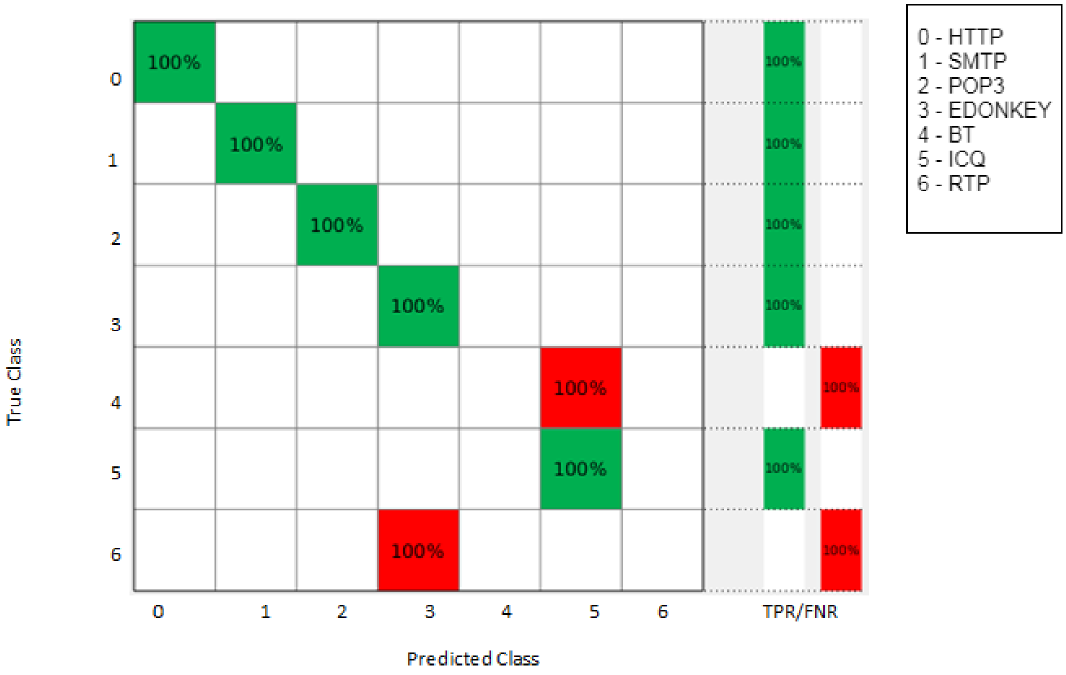


Figure 40: Confusion Matrix Graph of Proposed SVKM on Cup KDD'99 Dataset

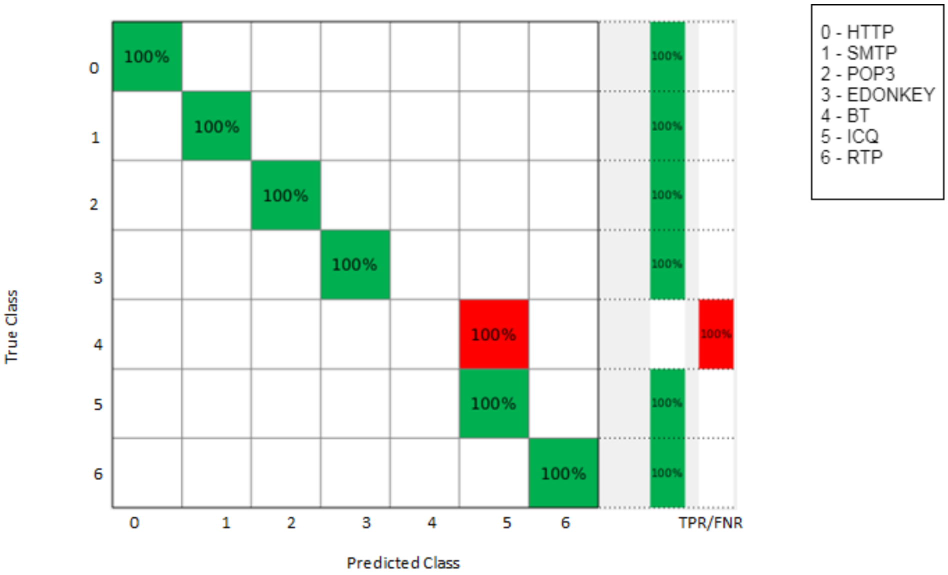


Figure 41: Confusion Matrix Graph of Proposed R-TAC on Cup KDD'99 Dataset

6.2.2 Results on Existing Algorithms

The dataset is tested on the three previously selected existing algorithms in literature. All algorithms revealed 7 classes except IATP with 6 classes revealed. The classes identified are Hypertext Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), Post Office Protocol version 3(POP3), Edonkey, Bit Torrent (BT), I Seek You (ICQ) VoIP application and Real-Time Transport Protocol (RTP).

KNN classifier assigned 3 classes (HTTP, POP3, ICQ) rightly against the true class. It wrongly assigned Edonkey to HTTP and RTP to SMTP class in Figure 42. However, part of SMTP is correctly assigned with the other portion grouped into POP3. The overall average accuracy achieved by the KNN classifier is 95.71%.

The hybrid KNN+K-Means classifier predicted POP3 and BT correctly into their respective classes as revealed in Figure 43. Classes HTTP, Edonkey and RTP were classified wrongly into SMTP, ICQ and HTTP respectively. Class SMTP was partly predicted correctly and other portion into class HTTP. Accuracy of KNN+K-Means classifier on the dataset resulted in 93.39%.

IATP classifier correctly and perfectly classified 3 classes (Edonkey, BT, ICQ) out of the total 6 classes in Figure 44. 50% of HTTP flows are correctly assigned while 50% is classified as ICQ. Furthermore, 75% of SMTP is predicted as its true class and 25% assigned to Edonkey class. 79% of POP3 generated traffic is correctly assigned and 21% wrongly assigned to ICQ. The overall accuracy achieved by the classifier is 79.30%.

6.3 Analysis of Results on CUP KDD’99 Dataset

In relation to HTTP generated traffic identified, the proposed algorithms were able to classify them better compared to the existing algorithms. KNKM predicted 50% of it correctly and the other two proposed works classified them perfectly. Out of the existing algorithms, only KNN algorithm classified HTTP traffic correctly. With SMTP, all the existing algorithms could not classify them correctly but all the proposed works classified them perfectly. From the above, it can be inferred that the proposed works are better classifiers for HTTP and SMTP traffic compared to the existing algorithms.

With the exception of IATP classifier, all other classifiers classified POP3 correctly with respect to the true class. Most of the existing classifiers had difficulties in classifying Edonkey traffic and BT. This is due to the dynamic nature and features of P2P traffic. IATP correctly classified Edonkey correctly out of the existing models. Out of the proposed works, only KNKM wrongly classified Edonkey. With BT traffic, KNN+K-Means classifier and IATP achieved assigning them correctly. Only KNKM out of the proposed models predicted it correctly. From this, it can be concluded that IATP classifier is a better option for classifying P2P traffic.

The proposed works perfectly classified ICQ traffic. IATP, in addition, classified ICQ perfectly out of the existing works. KNKM and IATP could not identify RTP traffic on the whole which affected its overall accuracy prediction on the dataset. However, only R-TAC could classify RTP traffic perfectly out of the other classifiers which were able to identify RTP.

From the analysis conducted above, the proposed algorithms performed better in comparison to the existing works. R-TAC evolved as the overall best classifier on the dataset with an accuracy of 98.95%. Table 16 displays the summary of the analysis of the classifiers with the traffic types they predicted perfectly. The checkmark denotes perfectly classified and the cross mark denotes misclassified.

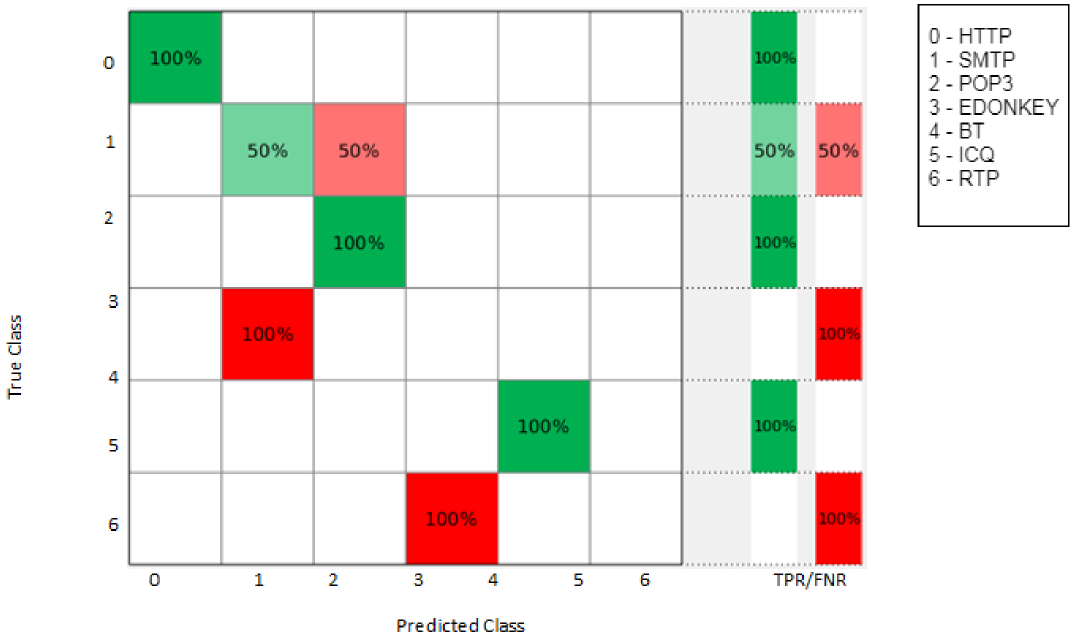


Figure 42: Confusion Matrix Graph of KNN on Cup KDD'99 Dataset

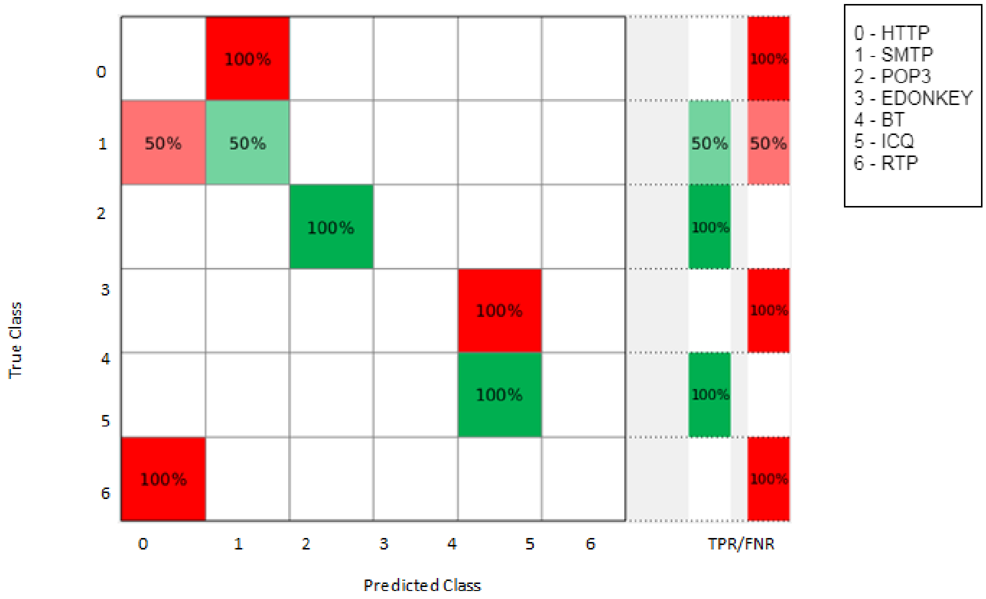


Figure 43: Confusion Matrix Graph of KNN+K-Means on Cup KDD'99 Dataset

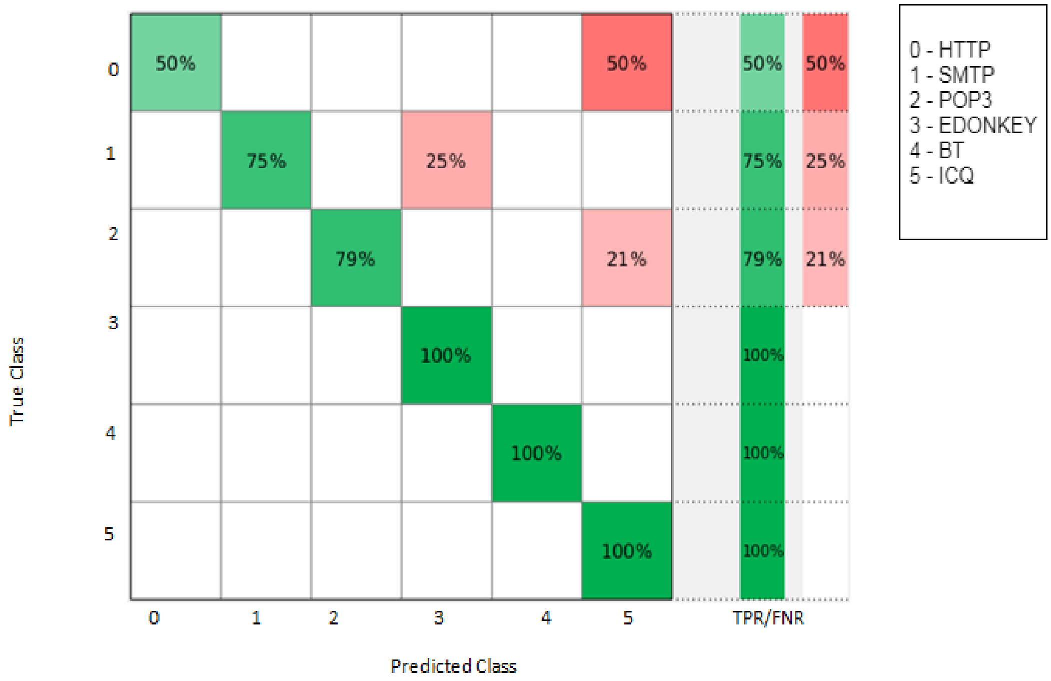


Figure 44: Confusion Matrix Graph of IATP on Cup KDD'99 Dataset

Table 16: Classification Analysis of Classifiers with Traffic Flows Classified and Misclassified on Cup KDD'99 Dataset

CLASSIFIER	APPLICATION						
	HTTP	SMTP	POP3	EDONKEY	BT	ICQ	RTP
KNN	✓	×	✓	×	×	×	×
KNN+K-MEANS	×	×	✓	×	✓	×	×
IATP	×	×	×	✓	✓	✓	
KNKM	×	✓	✓	×	✓	✓	
SVKM	✓	✓	✓	✓	×	✓	×
R-TAC	✓	✓	✓	✓	×	✓	✓

6.4 Validation of Algorithms with MACDCC IDS Trace

6.4.1 Results on Proposed Algorithms

All the proposed algorithms after classification revealed 9 classes where each class denotes the application protocol type used to generate the traffic trace. They include File Transfer Protocol (FTP), TCP (Transport Control Protocol), Hypertext Transfer Protocol (HTTP), Transport Layer Security (TLS), Server Message Block (SMB), Spanning Tree Protocol (STP), Internet Control Message Protocol (ICMP), Secure Sockets Layer (SSL), and Secure Shell (SSH). The accuracy results achieved from the classifiers are discussed and analyzed.

KNKM classified 7 classes accurately and predicted 2 classes (ICMP, SSH) wrongly. ICMP is predicted as STP and SSH as SSL as shown in Figure 45. The overall accuracy achieved for KNKM is 97.50% with the underlying dataset.

Out of the 9 classes, SVKM also classified 7 classes accurately and predicted 2 classes wrongly (TCP, STP). TCP is classified into HTTP class and STP is assigned to FTP class as depicted in Figure 46 constituting to an accuracy of 96.47%. Though both KNKM and SVKM had two classes predicted wrongly, the difference in classification accuracy is a result of the difference in the number of flows for the application protocols that were classified.

R-TAC had 8 out of 9 classes predicted accurately as displayed in Figure 47. Aside from the TCP class classified as FTP, all other classes are predicted rightly against the true class. The overall accuracy achieved by R-TAC is 98.97%.

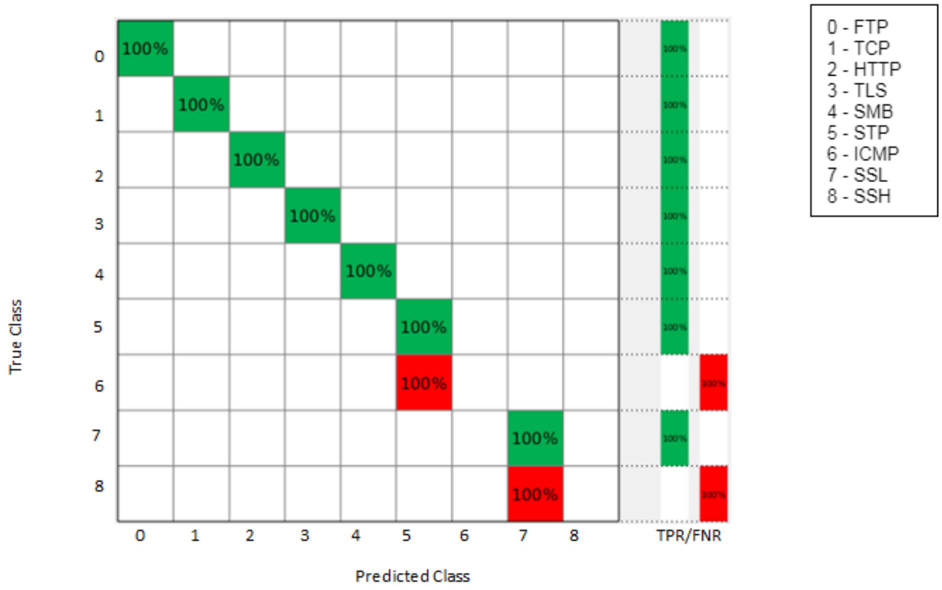


Figure 45: Confusion Matrix Graph of KNKM on MACDCC IDS Trace

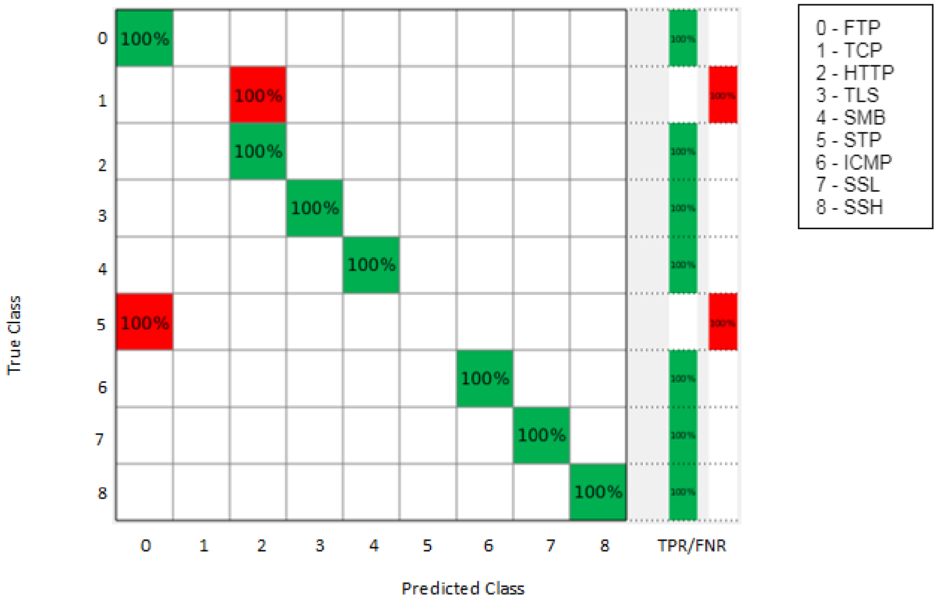


Figure 46: Confusion Matrix Graph of SVKM on MACDCC IDS Trace

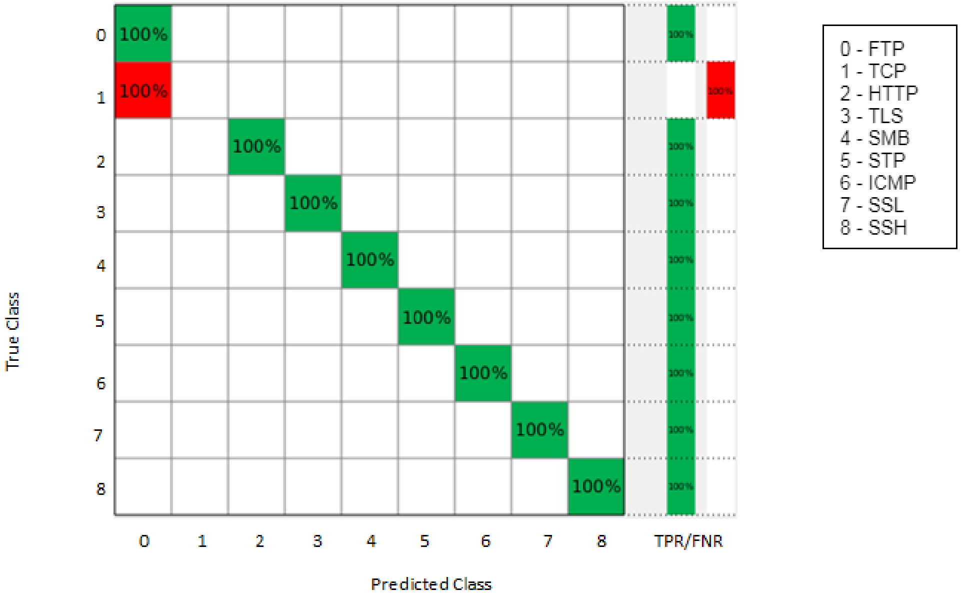


Figure 47: Confusion Matrix Graph of R-TAC on MACDCC IDS Trace

6.4.2 Results on Existing Algorithms

The existing works on the underlying dataset also revealed 9 classes (FTP, TCP, HTTP, TLS, SMB, STP, ICMP, SSL, SSH). However, the issue of overlapping classification was prevalent in with all classifiers.

KNN classifier predicted 4 classes (FTP, TCP, ICMP, SSL) perfectly. On the other hand, as displayed in Figure 48, 2 classes (TLS, STP) are classified incorrectly. Both TLS and STP classified are into SMB class. FTP class had 50% of flows predicted accurately, 25% into ICMP and 25% into SSH. The SMB class also had multiple classifications. 67% of the flows were classified accurately with 33% misclassified as ICMP. KNN achieved a classification accuracy of 91.30% after classification.

The KNN+K-Means hybrid classifier had 5 classes (FTP, TCP, HTTP, SMB, SSL) classified accurately into their respective classes. With misclassification in Figure 49, 3 classes namely STP, ICMP, SSH are assigned wrongly into SMB, STP, SSL respectively.

Class TLS had 50% classified as HTTP and 50% as SMB. The overall accuracy achieved by the hybrid classifier is 94.20%.

Out of the 9 classes revealed IATP had only 1 class (TCP) accurately classified with respect to its true class from Figure 50. Classes FTP, HTTP, TLS, SMB, STP, ICMP, and SSH were misclassified as TCP, FTP, SMB, FTP, SMB, SSL, and FTP respectively. SSL is also misclassified with 67% as TLS and 33% % as SMB. This resulted in an overall accuracy of 78.43%.

6.5 Analysis of Results on MACDCC IDS Trace

All classifiers achieved the same matrix dimension by revealing 9 classes where each class represents the application protocol used to generate the traffic type. In relation to FTP traffic, only the IATP classifier misclassified this traffic type. All other classifiers were able to predict it accurately with respect to the true class. With TCP traffic, the existing classifiers had no problem with its classification as they all perfectly classified it. However, SVKM and R-TAC could not classify TCP traffic accurately. Only KNKM out of the proposed works was able to achieve a perfect classification of TCP traffic. With respect to TCP traffic on the underlying IDS Trace, the existing works performed better than the proposed.

With HTTP traffic flows, the proposed works perfectly assigned them into their respective classes. Only KNN+K-Means classified this traffic type accurately out the existing works. For SMB traffic flows, again the proposed works classified them accurately while only KNN+K-Means hybrid classifier achieved a correct classification out of the existing models. Therefore, it can be inferred that the proposed works outperformed the existing works in classifying HTTP and SMB traffic.

Furthermore, two of the proposed works experienced no misclassification with STP generated traffic compared to the existing works. SVKM had a misclassification problem with this class. None of the existing models could classify STP class perfectly as they were all misclassified into other classes. With ICMP traffic, two of the proposed works

(SVKM and R-TAC) assigned all flows to their respective classes. Only the KNN classifier out of the existing works could achieve this. Therefore it can be stated that the existing works have poor performance with classifying STP and ICMP flows compared to the proposed models. In addition, with the exception of IATP classifier, all other classifiers had no misclassification issues with SSL traffic flows. With SSH traffic flows, only 2 proposed classifiers (SVKM and R-TAC) were able to classify these flows accurately. All existing works misclassified the flows into other classes.

From the analysis above, the proposed models did not experience the classification of a class into multiple classes as compared to the existing models. The least performing classifier in terms of accuracy for the proposed works (96.47% - SVKM) algorithm outperforms the best of the existing models (94.20% - KNN+K-Means). The overall best classifier again is R-TAC with 98.93% overall accuracy. From the underlying dataset or traces incorporated, it can be stated that the proposed works outperform the existing models with R-TAC being the overall best classifier. Table 17 demonstrates a summary of the analysis.

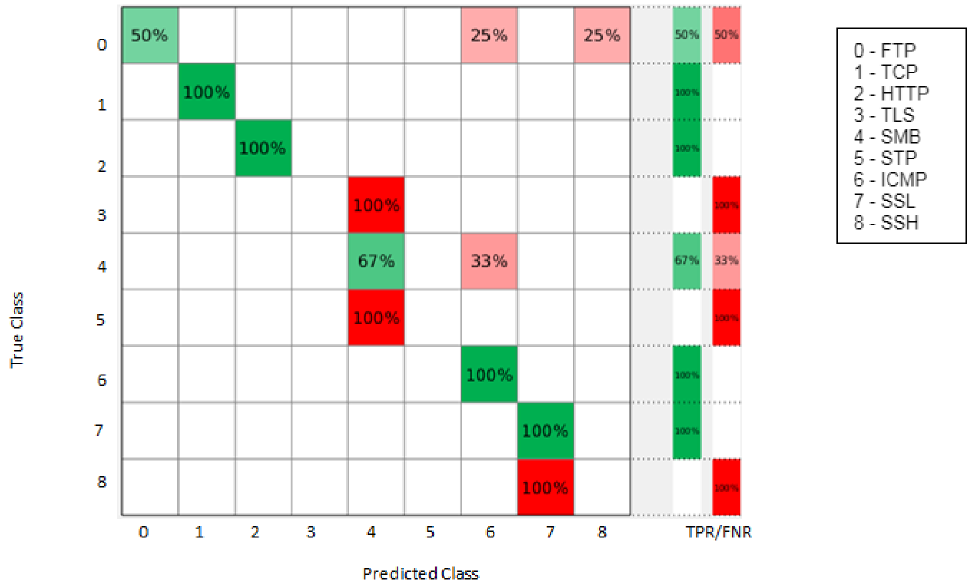


Figure 48: Confusion Matrix Graph of KNN on MACDCC IDS Trace

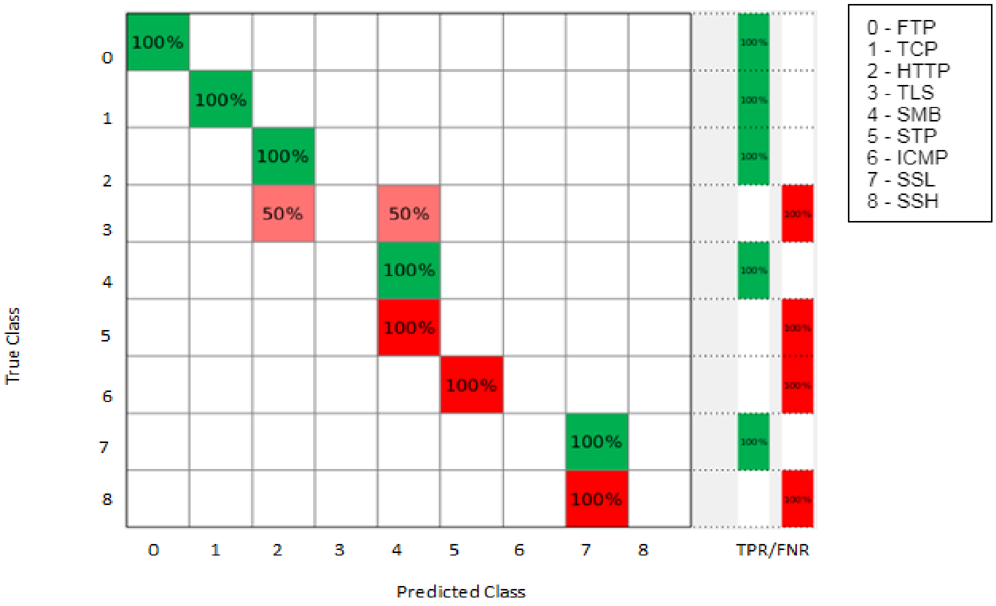


Figure 49: Confusion Matrix Graph of KNN+K-Means on MACDCC IDS Trace

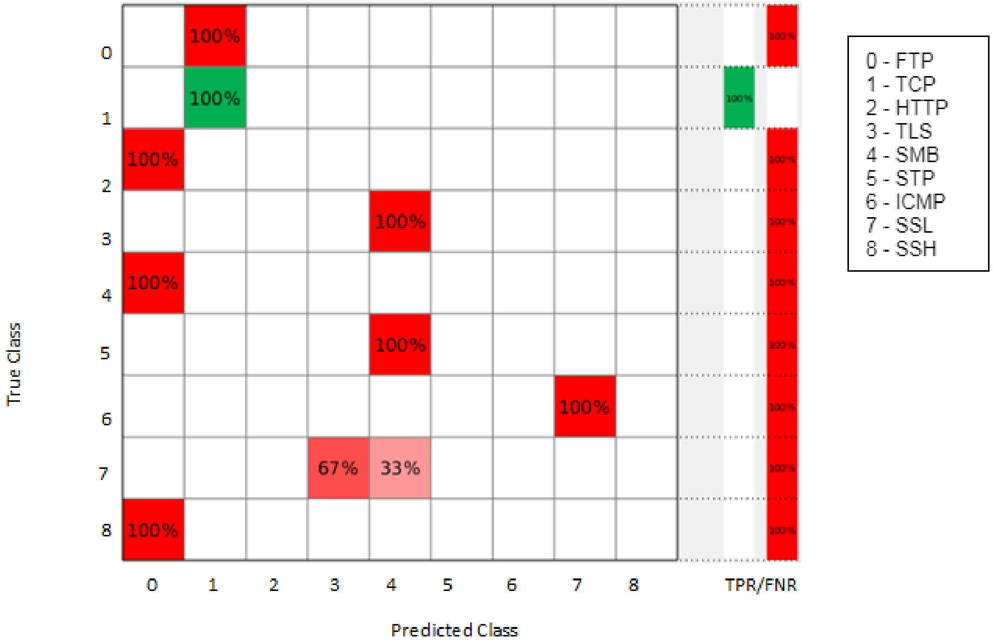


Figure 50: Confusion Matrix Graph of IATP on MACDCC IDS Trace

Table 17: Classification Analysis of Classifiers with Traffic Flows Classified and Misclassified on MACDCC IDS Trace

CLASSIFIER	APPLICATION PROTOCOL							
	FTP	TCP	HTTP	SMB	STP	ICMP	SSL	SSH
KNN	✓	✓	×	×	×	✓	✓	×
KNN+K-MEANS	✓	✓	✓	✓	×	×	✓	×
IATP	×	✓	×	×	×	×	×	×
KNKM	✓	✓	✓	✓	✓	×	✓	×
SVKM	✓	×	✓	✓	×	✓	✓	✓
R-TAC	✓	×	✓	✓	✓	✓	✓	✓

6.6 Conclusions of Validated Results

In both cases of the two datasets utilized as well as the proposed dataset that suited the scenario of the study best, the proposed works (KNKM, SVKM, and R-TAC) achieved higher accuracy values when compared to the existing works. The least accuracy rate among them always exceeded the best of the existing models in all cases of the datasets used in the validation process. Hence, a conclusion can be drawn from the results and analysis in the above sections that the proposed algorithms when integrated as classifiers can achieve great results in application protocol classification and can also be implemented in areas of intrusion detection systems to identify different protocols used in traffic generation of traffic.

Furthermore, the R-TAC algorithm achieved tremendous accuracy results in all cases of validation with the datasets. Because a classifier’s prediction and accuracy rates justify its performance and efficiency, a conclusion can be drawn that, the proposed R-TAC algorithm is the overall best classifier and also possess competitive time complexity with all other algorithms. It can be concluded that the proposed algorithms can be implemented as stand-alone classifiers, able to classify traffic flows or packet flows in the presence of rapid packet loss with variable fragmentation parameters. The comparative rundown of results after the analysis is depicted in Table 18 with the proposed and existing datasets.

The QoS parameters latency and throughput in the scenario deteriorates with increasing inter-arrival time. Therefore only a limited amount of flows with a small number of packets and features can be extracted. With small amount of features, the problem of overlapping classes are likely to occur which limits the classification accuracy of a classifier. The proposed algorithms can work efficiently in the presence of these QoS parameters using the limited features available to classify the traffic and produce fine classes or clusters.

Table 18: Summary of Classification Accuracy Results

CLASSIFIER	CLASSIFICATION ACCURACY (%)			
	CUP KDD’99	MACDCC IDS TTRACE	FPL 1 DATASET (Wired)	FPL 2 DATASET (Wireless)
KNN	95.71	91.30	73.79	70.45
KNN+K-MEANS	93.39	94.20	62.5	68.70
KNKM	79.30	78.43	69.40	72.50
IATP	96.76	97.50	91.30	90.86
SVKM	97.64	96.47	92.40	91.54
R-TAC	98.95	98.97	96.40	95.63

Chapter 7: Conclusion and Future Works

7. 1 Conclusion

The groundwork and thesis explain machine learning, clustering techniques and its applications in networking, particularly in wide and wireless area networks. Furthermore, pervasive literature scrutiny is conducted to fathom the diversified methods adopted by researchers to classify packets or traffic flows, their findings, drawbacks, and achievements. The gap in research identified is that not much investigation has been done to find out the effects of quality of service parameters which includes packet loss, congestion, fragmentation, bandwidth on the classification procedure. Moreover, we realized that these parameters that contribute to poor quality of service results mainly from limited resources not able to satisfactorily serve the requirements of a network. Therefore, we investigate the negative effects of packet loss with packet fragmentation as a result of varied packet length on the classification procedure. We design a topology scenario as such in simulation to obtain a dataset. The initial topology is in a wired environment which covers the scope of the research. However, a second scenario in a wireless environment is proposed to validate the efficiency of the proposed algorithms in such scenarios. The results show that the latency of transmission is increased rapidly when these parameters are highly prominent in a network. The QoS parameters latency and throughput in the scenario deteriorates with increasing inter-arrival time. Therefore only a limited amount of flows with a small number of packets and features can be extracted. With small amount of features, the problem of overlapping classes are likely to occur which limits the classification accuracy of a classifier. The proposed works implemented as classifiers can work efficiently in the presence of these QoS parameters using the limited features available to classify the traffic and produce fine classes or clusters.

The first algorithm combines the advantages of K-Medoids algorithm and KNN to obtain a semi-supervised strategic algorithm. The ability of the K-Medoids algorithm to cut down dissimilarity of separation between data traces is optimized with the use of the Euclidean Distance in the distance computation. KNN algorithm's ability to label or classify data objects with low time complexity is optimized with the incorporation of Weighted KNN where the weights of each data object to the query point are used instead of the distance. After its application on the dataset, the results from Table 14 demonstrates that the algorithm under proposition achieves tremendous accuracy and precision values in classification. A time complexity of $O(k(n\text{-}k)^2$ is achieved. We therefore proposed the second algorithm to improve upon the first in terms of all the evaluation metrics.

The second algorithm (SVKM) replaces the labelling procedure of the first by exploiting the robustness of the SVM algorithm with kernel tricks. Two kernels are utilized which are the Cubic Polynomial kernel and Gaussian kernel to identify the optimal hyperplanes and classify the support vectors into their appropriate classes. From Table 14, objective to improve upon the proposed KNKM is achieved in terms of all the evaluation metrics with 92.4% accuracy in classification is achieved. Also, the processing time is reduced from 4.5482 seconds in the first proposed algorithm to 2.9839 seconds which is almost a half-rate reduction.

However, the complexity of SVKM algorithm turned out to be a little higher compared to KNKM algorithm. Furthermore, the accuracy improved by a minimal rate of 1.1% which needs further improvement. To this effect, Real-Time Application Clustering (R-TAC) is proposed. Unlike the previously proposed algorithms, this semi-supervised algorithm prepares the training dataset into the real-time format by transforming into a higher dimensional feature space. The incorporation of a quadratic function is employed for this transformation. With respect to the clustering and classification phases, the Minkowski distance and Gaussian functions metrics are utilized for distance computations. The algorithm is robust and achieves a lower computational complexity of $O\ (n^2)$ compared to

SVKM of *O* *(n^{3})*. The accuracy rate achieved is as high as 96.40%, a massive raise from 1.1% to 5.1% with respect to the initial KNKM proposed work.

To validate the proposed works, the algorithms incorporated into a classifier are compared with other 3 renowned classifiers (KNN KNN+K-Means, IATP classifiers). Though as at the time of the study there were no datasets in literature that suits the proposed scenario, two datasets for Intrusion Detection which bears feature similarities with our proposed FPL dataset is employed. The Cup KDD 1999 and MACDCC IDS Trace are selected. In both cases of the datasets, the results in terms of accuracy metrics show that the proposed works are efficient classifiers compared to the existing renowned classifiers. However, with respect to specific application traffic like TCP and Bit torrent traffic, the existing classifiers performed better than the proposed models unlike validation with the proposed dataset. The overall best classifier is the proposed R-TAC with the high accuracy rates of 96.40%, 98.95%, and 98.93% for FPL dataset, Cup KDD 1999 dataset, and MACDCC IDS Trace respectively.

7.1.2 Summary of Findings

The summary of the findings in the study conducted can be summarized as

1. Packet Loss and Fragmentation has an effect on the inter-arrival time between packets flows leading to increased latency and declining throughput rates in the classification procedure.
2. As a result, only a few flows can be classified at a time which limits the amount of features that can be identified for clustering and classification
3. The proposed algorithms can be implemented as classifiers that are able to withstand scenarios of extreme packet loss to classify the packets with high accuracy and lesser error rates.
4. The proposed algorithms have higher prediction propensities of classes compared to the existing algorithms.

5. Hybrid clustering or classification algorithms can increase performance in classification but can be limited by time complexity.

7.2 Future Works

The situation of limited resources in networks will continue to be evident as requirements of networks change each day to meet the needs of network users. For further investigation in the future, other quality of service parameters including bandwidth leading to congesting will be investigated in diverse networks such as wireless networks with wireless sensor nodes. Though the accuracy achieved for R-TAC is high, we aim to achieve a perfect score of accuracy levels. The parameter for the algorithm will be investigated further to achieve this.

Furthermore, the issues pertaining to computational complexity still persist for clustering algorithms. For future studies, much concentration will be given to this area of research to reduce the heaviness of all the algorithms. For this study, the scope was limited to wide area networks. We aim to expand the solutions to various networks types such as Adhoc networks and Wireless Sensor Networks.

Bibliography

[1] V. Carela-Español, P. Barlet-Ros, A. Cabellos-Aparicio, and J. Solé-Pareta. “Analysis of the impact of sampling on NetFlow traffic classification.” *Computer Networks*, vol. 55, no. 5, pp.1083-1099, 2011.

[2] Z. S. Hosseini, S. J. S. M. Chabok and S. Kamel. “DOS intrusion attack detection by using of improved SVR.” *In Technology, Communication and Knowledge (ICTCK), 2015 International Congress* on, pp. 159-164, IEEE, 2015.

[3] A. Garg and P. Maheshwari. “Identifying anomalies in network traffic using hybrid Intrusion Detection System.” *In Advanced Computing and Communication Systems (ICACCS), 2016 3rd International Conference on*, vol. 1, pp. 1-6. IEEE, 2016.

[4] M. Ahmed, A. N. Mahmood, and J. Hu. "A survey of network anomaly detection techniques." *Journal of Network and Computer Applications,* vol. 60, pp. 19-31, 2016.

[5] C. M. Tseng, G. T. Huang, and T. J. Liu. "P2P traffic classification using clustering technology." *In System Integration (SII), 2016 IEEE/SICE International Symposium on*, pp. 174-179, IEEE, 2016.

[6] H. Jiang, A. W. Moore, Z. Ge, S. Jin and J. Wang. "Lightweight application classification for network management." *In Proceedings of the 2007 SIGCOMM workshop on Internet network management,* pp. 299-304. ACM, 2007.

[7] S. H. Yoon, J.W. Park, J.S. Park, Y.S. Oh and M.S. Kim. "Internet application traffic classification using fixed IP-port." *Management Enabling the Future Internet for Changing Business and New Computing Services (2009),* pp. 21-30, 2009.

[8] Internet Assigned Numbers Authority (IANA), http://www.iana.org/assignments/port-numbers, as of May 25, 2017.

[9] M. Roughan, S. Sen, O. Spatscheck and N. Duffield. "Class-of-service mapping for QoS: a statistical signature-based approach to IP traffic classification." *In Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pp. 135-148. ACM, 2004.

[10] A. W. Moore and D. Zuev. “Internet traffic classification using bayesian analysis techniques.” *In ACM SIGMETRICS Performance Evaluation Review*, vol. 33, no. 1, pp. 50-60, ACM, June 2005.

[11] T. Auld, A. W. Moore and S. F. Gull. “Bayesian neural networks for internet traffic classification.” *IEEE Transactions on neural networks,* vol. *18,* no. 1, pp.223-239, 2007.

[12] T. T. Nguyen and G. Armitage. “Training on multiple sub-flows to optimise the use of machine learning classifiers in real-world ip networks.” *In Proceedings. 2006 31st IEEE Conference on Local Computer Networks*, pp. 369-376, IEEE, November 2006,

[13] M. Crotti, M. Dusi, F. Gringoli and L. Salgarelli. “Traffic classification through simple statistical fingerprinting.” *ACM SIGCOMM Computer Communication Review*, vol. *37*, no.1, pp. 5-16, 2007.

[14] D. Schwartzman, W. Mühlbauer, T. Spyropoulos and X. Dimitropoulos, “Digging into HTTPS: flow-based classification of webmail traffic.” *In Proceedings of the 10th ACM SIGCOMM conference on Internet measurement,* pp. 322-327, ACM, November 2010.

[15] R. Alshammari and A. N Zincir-Heywood. “Machine learning based encrypted traffic classification: Identifying ssh and skype.” *In 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 1-8, IEEE, July 2009.

[16] T. T. Nguyen and G. Armitage. "A survey of techniques for internet traffic classification using machine learning." *IEEE Communications Surveys & Tutorials* 10, no. 4, pp. 56-76, 2008.

[17] P. Schneider, Patrick. "TCP/IP traffic Classification Based on port numbers." *Division of Applied Sciences, Cambridge, MA*, vol. 2138, 1996.

[18] A. W. Moore and K. Papagiannaki. "Toward the Accurate Identification of Network Applications." *In PAM*, vol. 5, pp. 41-54, 2005.

[19] F. Dehghani, N. Movahhedinia, M. R. Khayyambashi, and S. Kianian. "Real-time traffic classification based on statistical and payload content features." *In Intelligent Systems and Applications (ISA), 2010 2nd International Workshop on*, pp. 1-4. IEEE, 2010.

[20] J. Levandoski. “Application layer packet classifier for Linux.” *http://l7-filter. sourceforge. net/*, 2008

[21] V. Paxson. “Bro: a system for detecting network intruders in real-time.” *Computer networks*, vol. *31*, no. 23-24, pp. 2435-2463, 1999.

[22] S. Sen, O. Spatscheck, and D. Wang. “Accurate, scalable in-network identification of p2p traffic using application signatures.” *In Proceedings of the 13th international conference on World Wide Web*, pp. 512-521. ACM, May 2004.

[23] A. Finamore, M. Mellia, M. Meo, and D. Rossi. “Kiss: Stochastic packet inspection classifier for udp traffic.” *IEEE/ACM Transactions on Networking (TON)*, vol. 18, no. 5, pp. 1505-1515, 2010.

[24] P. Haffner, S. Sen, O. Spatscheck and D. Wang. “ACAS: automated construction of application signatures.” *In Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data*, pp. 197-202, ACM, August 2005.

[25] J. Ma, K. Levchenko, C. Kreibich, S. Savage, and G.M. Voelker. “Unexpected means of protocol inference.” *In Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pp. 313-326, ACM, October 2006.

[26] T. Karagiannis, A. Broido and M. Faloutsos. “Transport layer identification of P2P traffic.” *In Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pp. 121-134, ACM, October 2004.

[27] K. Xu, Z. L. Zhang and S. Bhattacharyya. “Profiling internet backbone traffic: behavior models and applications.” *In ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4, pp. 169-180, ACM, August 2005.

[28] M. Iliofotou, P. Pappu, M. Faloutsos, M. Mitzenmacher, S. Singh and G. Varghese. “Network monitoring using traffic dispersion graphs (tdgs).” *In Proceedings of the 7th ACM SIGCOMM conference on Internet measurement* pp. 315-320, ACM, October 2007.

[29] Y. Jin, N. Duffield, P. Haffner, S. Sen and Z. L. Zhang. “Inferring applications at the network layer using collective traffic statistics.” *In 2010 22nd International Teletraffic Congress (lTC 22),* pp. 1-8, IEEE, September 2010.

[30] P. Bermolen, M. Mellia, M. Meo, D. Rossi and S. Valenti. “Abacus: Accurate behavioral classification of P2P-TV traffic.” *Computer Networks*, vol. *55,* no. 6, pp.1394-1411, 2011.

[31] T. Z. Fu, Y. Hu, X. Shi, D. M. Chiu and J. C. Lui. “Pbs: Periodic behavioral spectrum of p2p applications.” *In International Conference on Passive and Active Network Measurement,* pp. 155-164, Springer, Berlin, Heidelberg, April 2009.

[32] T. Karagiannis, K. Papagiannaki, N. Taft and M. Faloutsos. “Profiling the end host.” *In International Conference on Passive and Active Network Measurement*, pp. 186-196, Springer, Berlin, Heidelberg. April 2007.

[33] A. McGregor, M. Hall, P. Lorier and J. Brunskill. "Flow clustering using machine learning techniques." *Passive and Active Network Measurement,* pp. 205-214, 2004.

[34] V. Paxson, “Empirically derived analytic models of wide-area TCP connections.” *IEEE/ACM Transactions on Networking*, vol. 4, pp. 316-336, 1994.

[35] C. Dewes, A. Wichmann and A. Feldmann. “An analysis of Internet chat systems.” *In Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement.”* pp. 51-64, ACM, October 2003.

[36] K. C. Claffy. *Internet traffic characterization* (Doctoral dissertation, University of California, San Diego, Department of Computer Science & Engineering). 1994.

[37] T. Lang, G. Armitage, P. Branch and H. Y. Choo. “A synthetic traffic model for Half-Life.” *In Australian Telecommunications Networks & Applications Conference*, Vol. 2003, December 2003.

[38] T. Lang, P. Branch and G. Armitage. “A synthetic traffic model for Quake3.” *In Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pp. 233-238, ACM, September 2004.

[39] U. R. Hodeghatta and U. Nayak. "Unsupervised Machine Learning." *In Business Analytics Using R-A Practical Approach*, pp. 161-186. Apress, 2017.

[40] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas. "Supervised machine learning: A review of classification techniques." *Emerging artificial intelligence applications in computer engineering*, vol. *160*, pp. 3-24. pp. 3-24, 2007.

[41] B. Hu & Y. Shen. “Machine learning based network traffic classification: a survey.” *Journal of Information & Computational Science*, vol. 9, no.11, pp. 3161-3170. 2012.

[42] O. Chapelle, B. Scholkopf and A. Zien. "Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]." *IEEE Transactions on Neural Networks*, vol.20, no. 3, pp. 542-542, 2009

[43] G. W. Milligan and M. C. Cooper. "Methodology review: Clustering methods." *Applied psychological measurement*, vol. 11, no. 4, pp. 329-354, 1987.

[44] W. H. Day and H. Edelsbrunner. "Efficient algorithms for agglomerative hierarchical clustering methods." *Journal of classification*, vol. 1, no. 1 pp. 7-24, 1984.

[45] A. Amini, T. Y. Wah, M. R. Saybani and S. R. A. S. Yazdi. "A study of density-grid based clustering algorithms on data streams.*" In Fuzzy Systems and*

Knowledge Discovery (FSKD), 2011 Eighth International Conference on, vol. 3, pp. 1652-1656. IEEE, 2011.

[46] P. Berkhin. “A survey of clustering data mining techniques”. *Grouping multidimensional data*, vol. 25, pp. 71, 2006.

[47] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal. *“Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann”*, Second Edition, Morgan Kaufman Publishers, 2016.

[48] M. Ester, H.P. Kriegel, J. Sander, and X. Xu. "A density-based algorithm for discovering clusters in large spatial databases with noise." *In Kdd*, vol. 96, no. 34, pp. 226-231, 1996.

[49] R. Agrawal, J. Gehrke, D. Gunopulos and P. Raghavan. “*Automatic subspace clustering of high dimensional data for data mining applications*”. Vol. 27, no. 2. ACM, 1998.

[50] W. Wang, J. Yang and R. Muntz. "STING: A statistical information grid approach to spatial data mining." *In VLDB*, vol. 97, pp. 186-195. 1997.

[51] Y. D. Lin, C. N. Lu, Y. C. Lai, W. H. Peng and P. C. Lin. "Application classification using packet size distribution and port association." Journal of Network and Computer Applications, vol. 32, no. 5, pp. 1023-1030, 2009.

[52] V. Estivill-Castro. "Why so many clustering algorithms: a position paper." *ACM SIGKDD explorations newsletter*, vol. 4, no. 1, pp. 65-75, 2002.

[53] L. Kaufman and P. J. Rousseeuw. “*Finding groups in data: an introduction to cluster analysis.*” Vol. 344, John Wiley & Sons, 2009.

[54] C. K. Reddy and B. Vinzamuri. “A Survey of Partitional and Hierarchical Clustering Algorithms.” *Data Clustering: Algorithms and Applications*, vol. 87, 2013.

[55] C. C. Aggarwal and C. K. Reddy. "Data clustering." *Algorithms and Applications, Chapman & Halls*, 2014.

[56] M. J. Zaki, W. M. Jr, and W. Meira. “*Data mining and analysis: fundamental concepts and algorithms.”* Cambridge University Press, 2014.

[57] A. Gersho and R. M. Gray. “Variable Rate Vector Quantization.” *In Vector Quantization and Signal Compression,* pp. 631-689, Springer, Boston, MA. 1992.

[58] I. V. Cadez, P. Smyth and H. Mannila. “Probabilistic modeling of transaction data with applications to profiling, visualization, and prediction.” *In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 37-46, ACM, August 2001.

[59] A. Ben-Dor, R. Shamir and Z. Yakhini. “Clustering gene expression patterns.” *Journal of computational biology*, vol. *6,* no. 3-4, pp. 281-297, 1999.

[60] J. Heer and E. H Chi. “Identification of web user traffic composition using multi-modal clustering and information scent.” *In Proc. of the Workshop on Web Mining, SIAM Conference on Data Mining*, pp. 51-58, April 2001.

[61] Takyi, K., Bagga, A. and Goopta, P. “Clustering Techniques for Traffic Classification: A Comprehensive Review.” *In 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO),* pp. 224-230. IEEE, August 2018.

[62] Takyi, K. and Bagga, A. “An Improved Classification Model for Wide
Area Networks with Low Speed Links.” *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 8, no. 8S3, June 2019

[63] C.D. Nocito and M.S. Scordilis, “January. Monitoring jitter and packet loss in VoIP networks using speech quality features.” *In Consumer Communications and Networking Conference (CCNC), 2011 IEEE,* pp. 685-686, IEEE. 2011.

[64] J. Pope and R. Simon. “The impact of packet fragmentation on internet-of-things enabled systems.” In *Information Technology Interfaces (ITI), Proceedings of the ITI 2013 35th International Conference on*, pp. 13-18, IEEE. June 2013.

[65] A.S Tanenbaum. Computer networks, 4-th edition. ed: Prentice Hall, 2003

[66] https://www.paloaltonetworks.com/

[67] A. Tongaonkar, R. Torres, M. Iliofotou, R. Keralapura and A. Nucci. 2015. “Towards self adaptive network traffic classification.” *Computer Communications*, vol. *56*, pp. 35-46.

[68] P. A. Branch, A. Heyde and G. J. Armitage. “Rapid identification of Skype traffic flows.” *In Proceedings of the 18th international workshop on Network and operating systems support for digital audio and video*, pp. 91-96, ACM, June, 2009.

[69] L. Bernaille, R. Teixeira and K. Salamatian. “Early application identification.” *In Proceedings of the 2006 ACM CoNEXT conference,* pp. 6, ACM December 2006.

[70] N. Williams, S. Zander and G. Armitage. “A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification.” *ACM SIGCOMM Computer Communication Review,* vol. *36*, no. 5, pp. 5-16, 2006.

[71] T. Karagiannis, K. Papagiannaki and M. Faloutsos. “BLINC: multilevel traffic classification in the dark.” *In ACM SIGCOMM computer communication review,* vol. 35, no. 4, pp. 229-240, ACM, August 2005.

[72] R. Bar-Yanai, M. Langberg, D. Peleg and L. Roditty. “Realtime classification for encrypted traffic.” *In International Symposium on Experimental Algorithms*, pp. 373-385, Springer, Berlin, Heidelberg, May 2010.

[73] J. Zhang, Y. Xiang, Y. Wang, W. Zhou, Y. Xiang and Y. Guan. “Network traffic classification using correlation information.” *IEEE Transactions on Parallel and Distributed Systems*, vol. *24*, no. 1, pp.104-117, 2013.

[74] P. Dorfinger, G. Panholzer and W. John. “Entropy estimation for real-time encrypted traffic identification (short paper).” *In International Workshop on Traffic Monitoring and Analysis*, pp. 164-171, Springer, Berlin, Heidelberg, April 2011.

[75] J. MacQueen. "Some methods for classification and analysis of multivariate observations." *In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14, pp. 281-297, 1967.

[76] S. Lloyd. "Least squares quantization in PCM." *IEEE transactions on information theory*, vol.28, no. 2, pp. 129-137, 1982.

[77] M. Hirvonen and J. P. Laulajainen. "Two-phased network traffic classification method for quality of service management." *In Consumer Electronics, 2009. ISCE'09. IEEE 13th International Symposium on*, pp. 962-966. IEEE, 2009.

[78] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman and A. Y. Wu. "An efficient k-means clustering algorithm: Analysis and implementation." *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 881-892, 2002.

[79] J. Z. Xiao and L. Xiao. "Analysis and improvement for K-Means Algorithm." *In Applied Mechanics and Materials*, vol. 52, pp. 1976-1980. Trans Tech Publications, 2011..

[80] P. Luarn, H. W. Lin, Y. P. Chiu, Y. L. Shyu and P. C. Lee. "The Categorising Characteristics of Facebook Pages: Using the K-Means Grouping Method." *International Journal of Business and Management*, vol. 11, no. 2, pp. 60, 2016.

[81] T. Zhang, R. Ramakrishnan, and M. Livny. "BIRCH: A new data clustering algorithm and its applications." *Data Mining and Knowledge Discovery*, vol.1, no. 2, pp. 141-182, 1997.

[82] T. Chiu, D. Fang, J. Chen, Y. Wang and C. Jeris. "A robust and scalable clustering algorithm for mixed type attributes in large database environment." *In Proceedings of the seventh ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 263-268. ACM, 2001.

[83] S. Ramaswamy, R. Rastogi and K. Shim. "Efficient algorithms for mining outliers from large data sets." *In ACM Sigmod Record*, vol. 29, no. 2, pp. 427-438. ACM, 2000.

[84] S. Guha, R. Rastogi, and K. Shim. "CURE: an efficient clustering algorithm for large databases." *In ACM Sigmod Record*, vol. 27, no. 2, pp. 73-84, ACM, 1998.

[85] R. T. Ng and J. Han. "Efficient and Effective Clustering Methods for Spatial Data Mining." In Proceedings of VLDB, pp. 144-155. 1994.

[86] M. Ankerst, M. M. Breunig, H. P. Kriegel and J. Sander. "OPTICS: ordering points to identify the clustering structure." *In ACM Sigmod record*, vol. 28, no. 2, pp. 49-60. ACM, 1999.

[87] K. Subramanian, A. Velkov, I. Ntoutsi, P. Kroger and H. P. Kriegel. "Density-based community detection in social networks." *In Internet Multimedia Systems Architecture and Application (IMSAA), 2011 IEEE 5th International Conference on*, pp. 1-8. IEEE, 2011.

[88] X. Xu, N. Y., Z. Feng and T. A. Schweiger. "Scan: a structural clustering algorithm for networks." *In Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 824-833. ACM, 2007.

[89] T. Falkowski, A. Barth and M. Spiliopoulou. "Dengraph: A density-based community detection algorithm." *In Web Intelligence, IEEE/WIC/ACM International Conference on*, pp. 112-115. IEEE, 2007.

[90] F. Hajikarami, M. Berenjkoub and M. H. Manshaei. “A modular two-layer system for accurate and fast traffic classification.” *In* Information *Security and Cryptology (ISCISC), 2014 11th International ISC Conference on*, pp. 149-154, IEEE, September 2014.

[91] Y. Jin, N. Duffield, J. Erman, P. Haffner, S. Sen and Z. L. Zhang. “A modular machine learning system for flow-level traffic classification in large networks.” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. *6*, no.1, p. 4, 2012.

[92] S. Zander, T. Nguyen, and G. Armitage. "Automated traffic classification and application identification using machine learning." *In Local Computer Networks, 2005, 30th Anniversary, The IEEE Conference on*, pp. 250-257, IEEE, 2005.

[93] NetMate, http://sourceforge.net/projects/netmate-meter (as of August 2005)

[94] P. Cheeseman and J. Stutz. "*Bayesian classification (autoclass): Theory and results in advances in knowledge discovery and data mining eds.*" Articles FALL, pp. 51, 1996.

[95] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule and K. Salamatian. “Traffic classification on the fly.” *ACM SIGCOMM Computer Communication Review*, vol. *36*, no.2, pp. 23-26, 2006.

[96] J. Erman, A. Mahanti and M. Arlitt. “Qrp05-4: Internet traffic identification using machine learning.” In *IEEE Globecom 2006*, pp. 1-6, IEEE, November 2006.

[97] T. T. Nguyen, G. Armitage, P. Branch and S. Zander. “Timely and continuous machine-learning-based classification for interactive IP traffic.*” IEEE/ACM Transactions On Networking*, vol. 20, no. 6, pp. 1880-1894, 2012.

[98] T. Nguyen and G. Armitage. “Synthetic sub-flow pairs for timely and stable IP traffic identification.” *In Proc. Australian Telecommunication Networks and Application Conference*. December, 2006.

[99] J. Erman, A. Mahanti, M. Arlitt, I. Cohen, and C. Williamson. Semi-supervised network traffic classification. *In SIGMETRICS*, pp. 369-37, June 2007.

[100] X. Zhu. “Semi-Supervised Learning Literature Survey.” world, vol. 10, p.10, 2005.

[101] A. Blum and T. Mitchell. “Combining labeled and unlabeled data with co-training.” *In Proceedings of the eleventh annual conference on Computational learning theory*, pp. 92-100. ACM. July 1998.

[102] J. Erman, A. Mahanti, M. Arlitt, I. Cohen and C. Williamson. "Offline/realtime traffic classification using semi-supervised learning*." Performance Evaluation*, vol. 64, no. 9, pp. 1194-1213, 2007.

[103] J. Erman, A. Mahanti and M. Arlitt. "Byte me: a case for byte accuracy in traffic classification." *In Proceedings of the 3rd annual ACM workshop on Mining network data*, pp. 35-38. ACM, 2007.

[104] Y. Wang, Y. Xiang, J. Zhang, W. Zhou, G. Wei and L. T. Yang. "Internet traffic classification using constrained clustering." *IEEE Transactions on Parallel and Distributed Systems*,vol. 25, no. 11, pp. 2932-2943, 2014.

[105] D. B. Shukla and G.S. Chandel. “An approach for classification of network traffic on semi-supervised data using clustering techniques.” In *Engineering (NUiCONE), 2013 Nirma University International Conference on,* pp. 1-6, IEEE, November 2013.

[106] J. Zhang, C. Chen, Y. Xiang, and W. Zhou. “Semi-supervised and compound classification of network traffic.” *In 2012 32nd International Conference on Distributed Computing Systems Workshops*, pp. 617-621, IEEE, June 2012.

[107] J. M. Keller, M. R. Gray and J.A. Givens. “A fuzzy k-nearest neighbor algorithm.” *IEEE transactions on systems, man, and cybernetics,* vol. 4, pp. 580-585, 1985.

[108] J. Yan, X. Yun, Z. Wu, H. Luo, S. Zhang, S. Jin and Z. Zhang. “Online traffic classification based on co-training method.” *In 2012 13th International Conference on Parallel and Distributed Computing, Applications and Technologies,* pp. 391-397, IEEE, December 2012.

[109] K. Gakhar, M. Achir, and A. Gravey. “How many traffic classes do we need in WiMAX?.” *In 2007 IEEE Wireless Communications and Networking Conference,* pp. 3703-3708, IEEE. March 2007.

[110] A. Dainotti, A. Pescape, and K. C. Claffy. “Issues and future directions in traffic classification.” *IEEE network,* vol. 26, no. 1, pp. 35-40, 2012.

[111] W. Zai-jian, Y. N. Dong, H. X. Shi, Y. Lingyun, and T. Pingping. "Internet video traffic classification using QoS features." *In Computing, Networking and Communications (ICNC), 2016 International Conference on,* pp. 1-5. IEEE, 2016.

[112] A. Gerber, J. Houle, H. Nguyen, M. Roughan and S. Sen. “P2P the gorilla in the cable.” *National Cable & Telecommunications Association (NCTA) 2003 National Show,* pp. 8-11, 2003.

[113] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble and H. M Levy. “An analysis of internet content delivery systems.” *ACM SIGOPS Operating Systems Review,* vol. *36,* no. SI, pp. 315-327, 2002.

[114] S. Senand and J. Wang. “Analyzing peer-to-peer traffic across large networks.” In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurement*, pp. 137-150, ACM, November 2002.

[115] T. Karagiannis, A. Broido, N. Brownlee, K. C. Claffy and M. Faloutsos. “Is p2p dying or just hiding? [P2P traffic measurement].” *In IEEE Global Telecommunications Conference, 2004. GLOBECOM'04.* vol. 3, pp. 1532-1538. IEEE. November 2004.

[116] L. Bin and T. Hao. “P2P Traffic Classification Using Semi-Supervised Learning.” *In Artificial Intelligence and Computational Intelligence (AICI), 2010 International Conference on,* Vol. 1, pp. 408-412, IEEE, October 2010.

[117] C.M Tseng, G.T. Huang and T. J. Liu. “P2P traffic classification using clustering technology.” In *2016 IEEE/SICE International Symposium on System Integration (SII)* (pp. 174-179). IEEE, December 2016.

[118] X. Du and X. Ou. “P2P flow classification based on wavelet transform.” *In Communication Problem-Solving (ICCP), 2015 IEEE International Conference on*, pp. 382-386, IEEE. October 2015.

[119] Y. T. Han. "Game traffic classification using statistical characteristics at the transport layer." *ETRI journal* vol. 32, no. 1, pp. 22-32, 2010.

[120] L. H. Do and P. Branch. "Real time VoIP traffic classification." *Tech. Rep. 090914A*, 2009.

[121] D. Achunala, M Sathiyanarayanan & B. Abubakar. “Traffic classification analysis using omnet++.” *In Progress in Intelligent Computing Techniques: Theory, Practice, and Applications*, pp. 417-422. Springer, Singapore 2018.

[122] P. Wang, S. C. Lin and M. Luo. "A framework for QoS-aware traffic classification using semi-supervised machine learning in SDNs." *In Services Computing (SCC), 2016 IEEE International Conference on*, pp. 760-765. IEEE, 2016.

[123] A. Dainotti, W. De Donato, A. Pescape and P. S. Rossi. "Classification of network traffic via packet-level hidden markov models. "*In Global*

Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE, pp. 1-5. IEEE, 2008.

[124] H. F. Hsiao, A. Chindapol, J. A. Ritcey, Y. C. Chen and J. N. Hwang. “A new multimedia packet loss classification algorithm for congestion control over wired/wireless channels.” *In Acoustics, Speech, and Signal Processing, 2005. Proceedings (ICASSP'05). IEEE International Conference on,* vol. 2, pp. ii-1105. IEEE, March 2005.

[125] S. H. Yoon, J. S. Park, M. S. Kim, C. Lim and J. Cho. "Behavior signature for big data traffic identification." In Big Data and Smart Computing (BIGCOMP), 2014 International Conference on, pp. 261-266. IEEE, 2014.

[126] CAIDA Data - *Overview of Datasets, Monitors, and Reports, [Online].* Available: http://www.caida.org/data/overview/, accessed on December, 2018

[127] T. M. Cover and P. E. Hart. “Nearest neighbor pattern classification.” *IEEE transactions on Information theory,* vol. 13 no. 1, pp. 21-7, 1967.

[128] H. S. Park and Jun C.H. “A simple and fast algorithm for K-medoids clustering.” *Expert systems with applications*, vol. *36*, no. 2, pp. 3336-3341, 2009.

[129] F. Pereira, T. Mitchell, and M. Botvinick. “Machine learning classifiers and fMRI: a tutorial overview.” *Neuroimage,* vol. 45, no.1, pp. S199-S209, 2009.

[130] R. Herbrich. *Learning kernel classifiers: theory and algorithms.* MIT press. 2001.

[131] L.I Kuncheva. *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons. 2004.

[132] M.A. Maloof. *Machine learning and data mining for computer security: methods and applications.* Springer Science & Business Media. ed.2006.

[133] L. Wang. *Support vector machines: theory and applications*, vol. 177, Springer Science & Business Media. ed. 2005.

[134] V. Vapnik, I. Guyon and T. Hastie. “Support vector machines.” *Machine Learning*, vol. *20, no.* 3, pp. 273-297, 1995.

[135] A. Dainotti, C. Squarecella, E. Aben, K. Claffy, M. Chiesa, M. Russo, and A. Pescape, "Analysis of Country-wide Internet Outages Caused by Censorship",

Internet Measurement Conference (IMC), Berlin, Germany, Nov 2011, pp. 1-18, ACM

[136] A. Dainotti, R. Amman, E. Aben, and K. Claffy, "Extracting benefit from harm: using malware pollution to analyze the impact of political and geophysical events on the Internet", *ACM SIGCOMM Computer Communication Review (CCR),* vol. 42, no. 1, pp. 31-39, Jan 2012.

[137] The CAIDA UCSD Anonymized Internet Traces 2008 to 2018 http://www.caida.org/data/passive/passive_dataset.xml

[138] The CAIDA UCSD Statistical information for the CAIDA Anonymized Internet Traces, 2008. http://www.caida.org/data/passive/passive_trace_statistics.xml

[139] A. King, A. Dainotti, B. Huffaker, and k. claffy, "A Coordinated View of the Temporal Evolution of Large-scale Internet Events", *Computing,* vol. 96, no. 1, pp. 53--65, Jan 2014. Originally from the Proceedings of the Workshop on Internet Visualization (WIV), November 2012.

[140] The CAIDA UCSD Network Telescope Educational Dataset, 2012. http://www.caida.org/data/passive/telescope-educational_dataset.xml

[141] CAIDA OC48 Peering Point Traces dataset', 2012 to 2013, www.impactcybertrust.org, DOI 10.23721/107/1421849

[142] Internet Traces Archive, accessed on 14th December 2018 http://ita.ee.lbl.gov/index.html

[143] Waikato Internet Traffic Storage accessed on 21st December 2018 https://wand.net.nz/wits/catalogue.php

[144] S. Chaudhuri, D. Madigan, and U.M. Fayyad. “KDD-99: the fifth ACM SIGKDD international conference on knowledge discovery and data mining.” *SIGKDD Explorations*, vol. 1 no. 2, pp. 49-51, 2000.

[145] Capture files from Mid-Atlantic (MACCDC), 2012 https://www.netresec.com/?page=MACCDC

List of Publications

I. K. Takyi, A. Bagga and P. Gupta, “Clustering Techniques for Traffic Classification: A Comprehensive Review,” 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), pp. 224-230, 2018.

II. K. Takyi and A. Bagga. “An Improved Classification Model for Wide Area Networks with Low Speed Links.” International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 8, no. 8S3, pp. 165-174, 2019

III. K. Takyi, B. Amandeep and P. Gupta “A Semi-Supervised QoS-Aware Classification Approach for Wide Area Networks with Limited Resources” International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 8, no. 11, pp. 970-981, 2019

IV. K. Takyi and B. Amandeep “A Novel Clustering Strategy for Real-Time Application Traffic Classification in Wide Area Networks” International Journal of Communication Systems. (SCI Indexed) [Under Review]