Predicting Congestion in Network using Machine Learning Techniques

Y V Sneha1, Vimitha1 and Adesh N D\*

*1Shri Madhwa Vadiraja Institute of Technology and Management, Bantakal - 574115, Udupi, India*

*\*Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal -576104, Udupi, India*

***Abstract-* When a burst of packets enters the network, the existing capacity of the network may not be sufficient to support the traffic which leads to congestion in the network. The main problem of congestion is the loss of packets during transmission which affects the performance of the system. The packet loss can be avoided if congestion is detected prior and reduces the packet generation rate at source with effective measures. In the current protocols, there is a predefined mapping between the observed state and the corresponding action. For example, when there is a packet drop in the network (observed state), the congestion window is reduced (action) irrespective of other parameters related to the networking environment such as resource utilization by each user, moving average, etc. Therefore, these protocols are unable to adapt their behaviour in the new environment or learn from past experience for better performance. To overcome these issues, the Machine Learning (ML) technique is required in the field of networking to learn from past experience and analyze the current network scenario to take certain actions. ML has the ability to deal with huge amounts of complex data which becomes one of the reasons for applying ML in the field of networking.**

**Keywords: Router based congestion control, Machine learning, Supervised learning, Congested network, Queue overflow.**

1. INTRODUCTION

The Cisco annual internet report indicates that nearly ⅔ of the world's population will have internet access by 2023. There will be a hike in the number of internet users from 3.9 billion to 5.3 billion by the end of 2023 [1]. The rapid increase in network traffic will pave the way to increase in network congestion. Congestion is one of the major problems in the network due to excess traffic in the network compared to its execution capacity. There are many algorithms which are developed to detect congestion in a network and to handle it [2-5]. However, these existing mechanisms are based on fundamental assumptions. In the existing schemes, there is a pre-decided mapping between the network’s state and the corresponding action [6]. As a result of this, the rules in changing the congestion window size remain the same irrespective of the network behaviour. Different kinds of network differ in their behaviour depending on bandwidth capacity, network topology, queue size at router, and tolerant time of packet delay. Therefore, the existing protocols are unable to adapt to varying network behaviour or learn from the past experience. In order to overcome this issue, Machine Learning is being applied in networking [6] to adjust the congestion window size with respect to different network behaviour.

Machine learning (ML) is an emerging technology which is applied in various sectors such as healthcare, marketing and sales, financial services, government agencies and so on [7]. In recent years, networking has also benefited from this trending technology such as machine learning in security [8], routing [9], IoT [10], etc. But the need for ML in networking is because of the following reasons [7]. First, ML will help in decision making i.e, assigning correct class labels for the unseen data or deciding what actions to perform on the current states of the network environment based on the training process. Second, interaction with the complex network is made possible using machine learning. Finally, machine learning provides a way to develop a generalized model such that researchers can independently solve network problems associated with different scenarios.

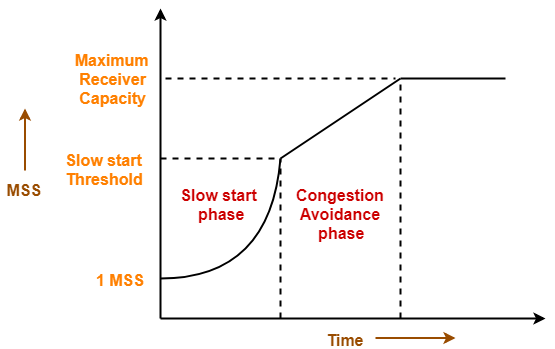
The limitations of rule-based protocols [6] inspired us to build a model which will make use of machine learning intelligence to learn out the possibilities of congestion at the router. The following objectives can be achieved using a machine learning approach in network: (1) Router is able to learn about the possibilities of congestion from the training provided. (2) The amount of reduction in congestion window size is estimated at the router based on queue parameters and sent to the sender. (3) The sender reduces the sending rate depending on the amount of reduction in the congestion window size received in the TCP acknowledgement [11].

The rest of the paper is as follows: Section 2 presents a brief overview on various existing congestion control mechanisms and the limitations of those mechanisms. Section 3 describes the proposed methodology to handle congestion in an effective manner. The simulation setup and performance analysis are presented in Section 4 and Section 5 highlights the conclusion of research work.

2. LITERATURE REVIEW

2.1 Background theory

Congestion occurs when load on the network is greater than the execution capacity of the network. Congestion is a dynamic problem which cannot be solved but can be reduced [6]. The various congestion control mechanisms in wired and wireless networks are discussed in [2]. There are three phases in congestion control mechanisms such as slow start, congestion avoidance, and congestion detection phase [2] is as shown in Figure 1. TCP congestion control starts with a slow start phase and it increases the congestion window size exponentially until it reaches a threshold point. The next phase is congestion avoidance (Additive Increase Multiplicative Decrease) where congestion window size is incremented one by one until congestion is detected. The last phase is congestion detection where congestion window size is reduced when congestion occurs [11].



**Figure 1:** Phases of congestion control mechanism

Machine learning has made a tremendous breakthrough in a variety of application areas such as text recognition [12], computer vision [13], network security [8] and so on. ML tries to construct agents or models that learn without using predefined rules. For example, halving the congestion window after receiving three duplicate acknowledgements is a predefined rule in the existing congestion control mechanisms which can be avoided using ML. ML is a subset of artificial intelligence where the system will learn from its past experience and make decisions on its own without being programmed explicitly [7]. It is a universal tool that can be used for several purposes such as classification, clustering and so on. ML algorithms are basically grouped into three categories: Supervised Learning (SL), Unsupervised Learning (USL) and Reinforcement Learning (RL) [14]. Specifically, SL is used for tasks such as classification and regression from labelled data. USL algorithms conduct clustering from unlabeled data, and the RL algorithm finds the best action to be executed based on the varying environment condition. It is a tedious task for the existing methods to deal with different scenarios of the network, but Machine Learning tries to build models in order to make decisions from the collected data [15]. Machine learning algorithms find natural patterns in the data that produce observation and help us to make better decisions and predictions. Machine learning is an efficient technique because it will perform better than human beings during classification, clustering, regression, and decision making including large amounts of data [7].

2.2 Related work

In this section, related work and recent progress in the field of congestion control mechanisms are grouped into three categories: end-to-end schemes, router-based schemes and smart schemes.

2.2.1 End-to-end schemes

The different variety of TCP protocols such as TCP Tahoe, TCP Reno and TCP Vegas are being discussed in [2]. These mechanisms follow rule-based design principles in which actions are already predefined for a particular state of the network environment but congestion is a dynamic problem. As the number of users and the size of the network increases, the existing mechanism cannot solve the congestion issue with static solutions alone which results in unsatisfactory performance. The delay-based algorithm and loss-based algorithms are being discussed in [3]. Here, the authors proposed a methodology that enhances TCP’s popular algorithm: TCP NewReno which is loss-based (reactive) in nature. When there occurs a packet loss in the network, the congestion window size is halved in TCP NewReno. Whereas, ENewReno [4] considers the network state while reducing congestion window size but it is considered as inefficient in terms of packet loss. Thus, NewReno\_A was proposed which will take into consideration both congestion state and maintain maximum congestion window by using the 80:20 thumb rule. Thereby, decreases packet loss and increases throughput.

2.1.2 Router based schemes

Most of the congestion control mechanisms focus on congestion control at the end-devices rather than intermediate devices such as routers. In [16], the authors discussed both source-based and router-based approaches. Source-based approach like TCP is reactive in nature which uses implicit signals to notify congestion. That is, packet loss or delay or sometimes a combination of both is used. Router-based approaches are proactive in nature which will indicate congestion to the end-devices before packet drop and source reduces transmission rate accordingly [16]. Network congestion normally occurs when expected bandwidth exceeds the available link capacity and when the incoming packet rate to the router is greater than the outgoing rate. In order to overcome this issue, there is a need to design an algorithm to detect and avoid congestion in the router itself. Active Queue Management (AQM) mechanism was proposed to prevent buffer overflow [5]. AQM algorithms run on routers and detect congestion by monitoring average queue size. When average queue size exceeds a certain threshold but it is less than maximum size of the queue, then end systems are notified about the congestion by dropping some of the packets at the router or marking some packets as explicit notification. But these approaches require modification in the intermediate devices and routers which may not be a practical solution [17].

2.1.3 Smart schemes

Machine Learning has found its application in various areas such as text recognition, image recognition, pattern recognition and natural language processing which has led researchers in applying machine learning to solve networking problems as well [17]. The authors [7] explained how to apply machine learning technology in the networking domain. They investigated how machine learning technology can benefit network design and optimization and summarized the typical workflows and requirements for applying machine learning techniques in the network domain. QTCP (Q-learning based TCP) [6], which is based on Reinforcement Learning (RL) was developed to allow senders to dynamically learn different strategies to better adapt to varying networking scenarios, instead of mechanically following fixed rules. The authors showed that QTCP performs better when compared to TCP NewReno scheme.

After going through the various existing congestion control schemes, we conclude that in the existing schemes, there is a pre-decided mapping between the network’s state and the corresponding action. Therefore, the existing protocols are unable to adapt to varying network behaviour or learn from the past experience. In order to address this problem, there is a need for a smart mechanism using ML to estimate the possibility of the congestion in the router.

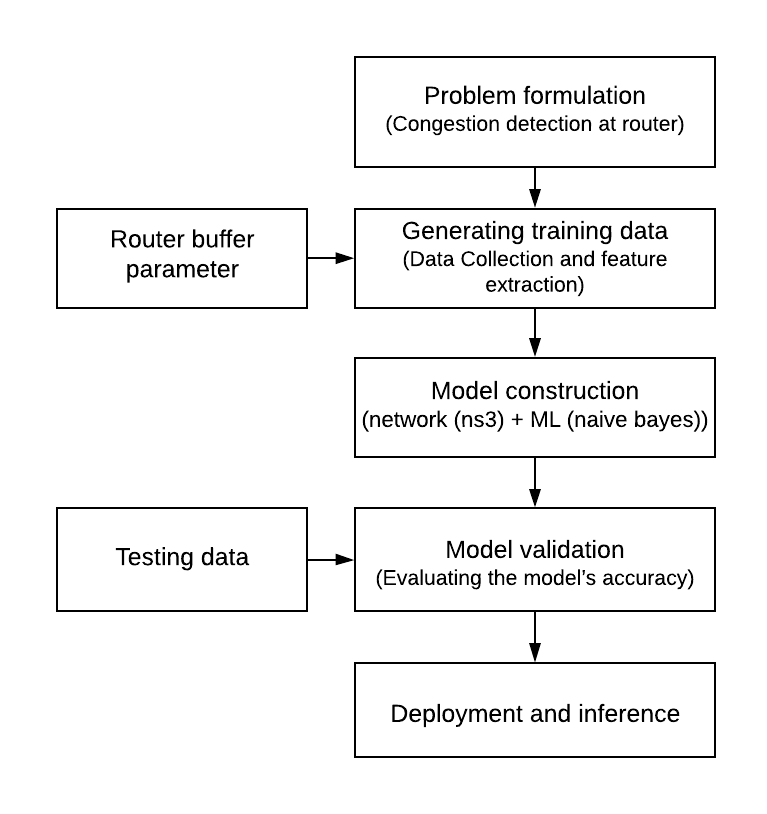
3. PROPOSED METHODOLOGY

Most of the TCP congestion control algorithms follow end-to-end principle i.e, either on the sender side or on the receiver side. But the proposed model works on the router side. The proposed model does the following 3 main tasks:

* Predict the possibility of congestion at the router side with the help of machine learning.
* If there is a chance of congestion, then the model estimates the amount of reduction in congestion window size.
* Then the sender is intimated to adjust the congestion window size in order to handle congestion effectively.

3.1 Predicting network Congestion using Machine Learning

The steps involved in the proposed methodology are represented in Figure 2.

**Figure 2:** Proposed methodology for predicting congestion in network

3.1.1 Problem formulation

The first step is to correctly abstract and formulate the problem statement. Network congestion is one of the major networking problems. Congestion in the network occurs due to excess generation of data traffic compared to the bandwidth capacity of the network which forces the excess data packets to store in queues (buffer space) of intermediate devices (router). When the queue space of the router is not sufficient to store the excess data packets which leads to drop the packets from the queue (buffer overflow). Usually routers drop packets to reduce congestion without informing the sender. The sender may not be able to detect the reason behind packet loss. Packet loss may be either due to buffer overflow or due to link error. When there is a bottleneck link in the end-to-end path, there is a possibility of congestion. In order to overcome these issues, there is a need to detect the possibility of congestion in the router before the packet loss occurs at the router by using Machine Learning (ML) techniques.

3.1.2 Generating training dataset

The goal of this step is to collect a network data set. Data can be either collected offline or online. In offline mode, a large amount of network data is collected, which plays a very important role in training the ML model. In online mode the live data from network simulator 3 (ns3) [18] is fed to the ML model for training the model.

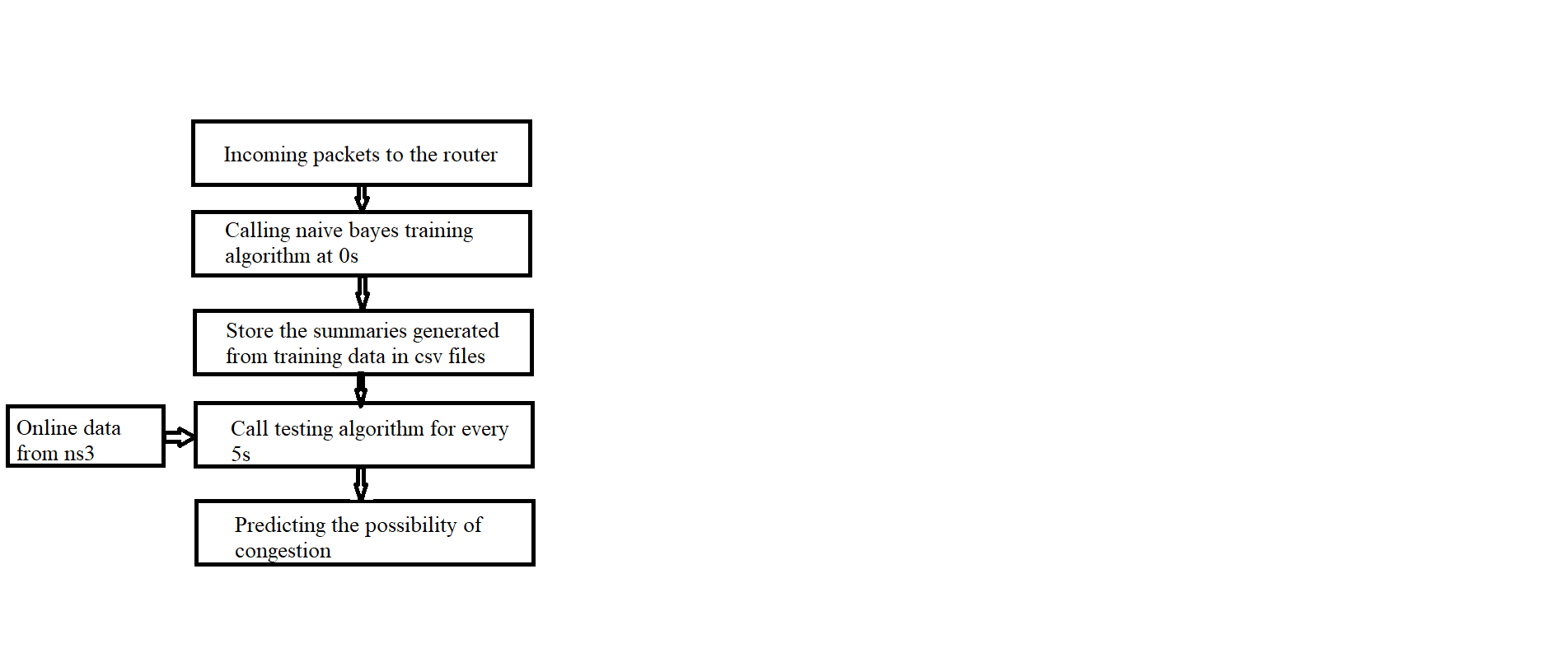
The training data consists of queue parameters such as current number of packets stored in queue (curpkts), current queue percentage (cur%), remaining packets that can be stored in queue (rempkts), remaining queue percentage (rem%), current queue average ( curavg), old average (oldavg) and class label indicating whether there occurs congestion or not. The maximum queue size and maximum queue percentage are denoted by maxpkts and max% respectively.

|  |
| --- |
| *Algorithm1: Generating dataset for training* |
| 1. procedure DatasetGeneration(parameters): 2. maxpkts = total capacity of queue in packets 3. curpkts= number of the packets in queue 4. rempkts=maxpkts-curpkts 5. max%=(maxpkts/maxpkts)\*100 6. cur%=(curpkts/maxpkts)\*100 7. rem%= max%- cur% 8. curavg = (oldavg\*beta)+((1-beta)\*curpkts) 9. oldavg= curavg 10. estimation= ((cur%-rem%) / (max%-cur%)) \* (curavg/oldavg) 11. if(estimation>=1) 12. Class label=1 13. else 14. Class label=0 |

Initially, simulation is performed to generate a dataset of predicting attributes. The class label is assigned based on congestion estimation formula (refer lines from 10 to 14 of Algorithm 1). The value of beta in curavg is set to 0.95 in order to smoothen the output values of curavg (refer line 8 of Algorithm 1). The generated data is stored into a csv file which is used to train the model.

3.1.3 Model construction

Model construction includes constructing a model using an appropriate machine learning algorithm. The ML algorithm is chosen with respect to the problem statement (in section 3.1.1) and generated training dataset (in section 3.1.2). Since network data associated with each feature is continuous-value, a Naive Bayes classifier [19] is chosen to predict and classify whether the network is going to be congested or not.



**Figure 4:** The proposed ML model for congestion detection

The steps involved in model construction are shown in Figure 4. Naive Bayes algorithm is being split into two parts such as training, and testing part. In the training part, the generated training dataset (as per algorithm1) is given as input to the Naive Bayes algorithm at 0 second which gives summaries of the entire training data. The generated summaries are stored into a file which will be used for future predictions of the class labels of testing data. The testing data is an online data which is generated during simulation of ns3 scenario. The testing part of Naive Bayes algorithm is called for every 5 seconds once to predict the congestion state of the network and the class labels are assigned to the testing dataset based on the summaries of the training data. The key benefit of using ML technique in networks is to detect possibilities of congestion before it occurs and thus avoiding packet drop from the network.

3.1.4 Model Validation

During this step, the model’s accuracy is tested to determine whether the class label is predicted correctly or not. This process is termed as cross validation. After the completion of the training process, the summaries generated by the training dataset will be used for future predictions. The proposed model is tested by providing testing data and validate for correct predictions.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Current number of  packets in queue | Remaining space to  store the packets | Current queue size  in percentage (%) | Remaining queue  size in percentage (%) | Old Average | Current Average | Prediction | reduce CWND |
| 2 | 48 | 4 | 96 | 0 | 0.1 | 0 | 0 |
| 12 | 38 | 24 | 76 | 0.1 | 0.19 | 0 | 0 |
| 29 | 21 | 58 | 42 | 27.35 | 27.68 | 0 | 0 |
| 30 | 20 | 60 | 40 | 27.68 | 28.05 | 0 | 0 |
| 35 | 15 | 70 | 30 | 28.05 | 28.35 | 1 | 13.19 |
| 39 | 11 | 78 | 22 | 28.35 | 28.68 | 1 | 29.20 |
| 44 | 6 | 88 | 12 | 28.68 | 29.04 | 1 | 57.40 |
| 48 | 2 | 96 | 4 | 29.04 | 29.34 | 1 | 85.35 |

**Figure 5:** Predicting congestion and window reduction

The testing data uses the summaries and assigns class labels as shown in the Prediction column of Figure 5. If the model meets the requirements after validation then it can be deployed. Otherwise, the procedures in the previous steps need to be re-taken based on the errors in the training process. The proposed model is validated by checking it with different data values. If it is predicting congestion for the congested network scenario and reducing the congestion window size as per the current network situation and vice-versa. Then, the model can be deployed in the router making the router perform in an intelligent manner.

3.1.5 Deployment and inference

The model is deployed typically once the training is completed. Then, the class labels are predicted for the online data generated during simulation. In the proposed approach, the model uses summaries generated by the training data to make predictions for the online data. But there is a possibility of the model making wrong predictions for the unseen or rapidly varying data which degrades the performance of the model. These are the practical issues which should be taken care of before deploying the model in order to obtain better performance.

3.2 Estimation of congestion window reduction

|  |
| --- |
| Algorithm2: Estimation of congestion window reduction |
| 1. procedure Reduce CWND (parameters): 2. prediction=Prediction from ML 3. if max(prediction) = = 1: 4. reduceCWND=max((((curpkts-rempkts)/(maxpkts-rempkts)) \*(curpkts-rempkts) \*((cur%-rem%)/(100-rem%))\*(curavg/oldavg)),0) 5. optional field of ipv4 =reduceCwnd 6. else: 7. follow the normal procedure |

Once the Naive Bayes is trained using the data generated as per Algorithm 1, then the summaries generated by it can be used for future prediction. After accomplishment of the training process, then Naive Bayes is fed with the live online data from ns3. Based on the summaries of the training process and the current network data from ns3, the Naive Bayes will predict the possibility of the network entering a congested state. Whenever the prediction is 1 (it indicates there is a possibility of congestion in the network), then the percentage of Congestion Window (CWND) reduction is calculated as shown in the reduce CWND column of Figure 5. Because the CWND is a variable which decides the amount of packets that can enter the network. This can be accomplished with the help of the formula presented in Algorithm2 line number 4. This computation takes place in a network layer. Since a router is a network layer entity it cannot access CWND which is a part of transport layer header. In order to adjust CWND, the reduction percentage is set in the option field of the ip header (refer line 5 of algorithm 2). Which is then forwarded to the receiver. On receiving the packet with the optional field being set, the receiver extracts those values and sets the advertisement window as per the optional field values sent from the router.

3.3 Adjustment of congestion window size

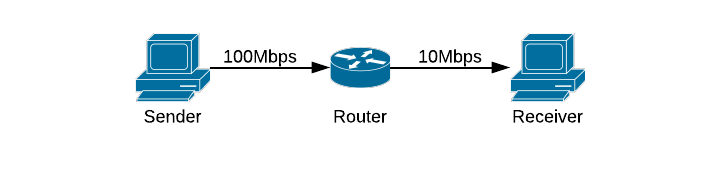
In this step, the congestion window size is adjusted at the sender side. The steps involved are as follows:

* Obtain the percentage of congestion window reduction from the acknowledgement received.
* Compute the amount of window reduction at the sender side.
* Reduce the window size based on the computed value.
* Set the window size to min (cwnd, rwnd) i.e., the minimum of congestion window size and receiver window size.

Thus, this mechanism monitors flow control by restricting the amount of data that a sender can send to the receiver and results in reliable data transmission.

4.SIMULATION STUDY

4.1 Simulation Setup



**Figure 6:** Network Topology with a single router

A simple topology is chosen for simulation which consists of a sender and receiver with a single router in between. A queue with a maximum size of 50 packets is being used in the router. The access bandwidth is usually chosen to be greater than bottleneck bandwidth in order to create a congested network environment and determine the working of the proposed method in such a network scenario as shown in Figure 6. The simulation parameters are represented in Table 1.

TABLE 1: Simulation parameters

|  |  |
| --- | --- |
| Parameter | Value setting |
| beta in current average | 0.95 |
| Simulation duration | 100 seconds |
| Access bandwidth | 100 Mbps |
| Bottleneck bandwidth | 10 Mbps |
| Buffer size | 50 packets |
| ML algorithm | Naive Bayes |

4.2 Performance Analysis

The performance of the proposed model is analyzed in this section. The performance metrics considered for the proposed model are discussed as follows:

* Accuracy: It is the ratio of number of correct predictions to the total number of predictions. It is essential to evaluate machine learning algorithms in order to check how accurately it works. For the proposed model, the accuracy is computed to be 0.96 or 96 %.
* Precision: It is the number of correct positive results divided by the number of positive results predicted by the classifier. The precision obtained for the proposed model is 1.0 which indicates the model does not produce false positives.
* Recall: Recall can be defined as the ratio of the total number of correctly classified positive examples to the total number of positive examples. The recall obtained for the proposed model is 0.94.

5. CONCLUSION

In this paper, the need for Machine Learning in networking has been discussed. Then, a survey is conducted on existing congestion control mechanisms and their limitations. The overall proposed methodology was divided into three phases. In the first phase, congestion is predicted using Machine Learning in the router. The Machine Learning model is trained using queue parameters. Supervised Learning algorithm Naive Bayes is being used to build a model to detect the congestion. Once congestion is detected, the amount of congestion window to be reduced is computed and sent to the receiver. Finally, the congestion window size is adjusted at the sender side based on the value received in the acknowledgement. This approach therefore helps us to eradicate the predefined mapping between observed state and corresponding action. Thus, handles congestion more effectively.

REFERENCES

[1] Cisco, Cisco Annual Internet Report (2018-2023), white paper, [Online]Available:https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html; Last accessed date: 12-June-2020.

[2] Gnv Vibhav Reddy, G. Vijay Kumar, L. Roshni, “Congestions and Control Mechanisms in Wired and Wireless Networks”, International Journal of Engineering And Science, Vol.4,pp-57-62,2014,Available:http://www.researchinventy.com/papers/v4i6/K046057062.pdf

[3] Sanjesh S. Pawale, Sanjeev J. Wagh; Ranjana S. Jadhav,” An efficient congestion control methodology to enhance performance of TCP in wired network”, Fourth International Conference on Image Information Processing (ICIIP), 2017, Available: https://doi.org/10.1109/ICIIP.2017.8313766

[4] Hanaa Torkey, Gamal Attiya, Ibrahim Z. Morsi," Modified Fast Recovery Algorithm for Performance Enhancement of TCP-NewReno”, International Journal of Computer Applications, February 2012 DOI:10.5120/5018-7351

[5] D. Amol and P. Rajesh, "A Review on Active Queue Management Techniques of Congestion Control," International Conference on Electronic Systems, Signal Processing and Computing Technologies, Nagpur, 2014, pp. 166-169. DOI: 10.1109/ICESC.2014.34

[6] Wei Li, Fan Zhou, Kaushik Roy Chowdhury, Waleed Meleis, QTCP: Adaptive Congestion Control with Reinforcement Learning,” IEEE Transactions on Network Science and Engineering”, vol. 6, pp. 455-458, 2019. Available: https://doi.org/10.1109/TNSE.2018.2835758

[7] Mowei Wang, Yong Cui, Xin Wang, Shihan Xiao, and Junchen Jiang,” Machine   Learning for Networking: Workflow, Advances and Opportunities”, IEEE networks, vol.32, pp-92-99,2018, Available: https://doi.org/10.1109/MNET.2017.1700200

[8] T. N. Nguyen, "The Challenges in ML-Based Security for SDN," 2018 2nd Cyber Security in Networking Conference (CSNet), Paris, 2018, pp. 1-9, doi: 10.1109/CSNET.2018.8602680.

[9] D. K. Sharma, S. K. Dhurandher, I. Woungang, R. K. Srivastava, A. Mohananey and J. J. P. C. Rodrigues, "A Machine Learning-Based Protocol for Efficient Routing in Opportunistic Networks," in IEEE Systems Journal, vol. 12, no. 3, pp. 2207-2213, Sept. 2018, doi: 10.1109/JSYST.2016.2630923.

[10] L. Xiao, X. Wan, X. Lu, Y. Zhang and D. Wu, "IoT Security Techniques Based on Machine Learning: How Do IoT Devices Use AI to Enhance Security?," in IEEE Signal Processing Magazine, vol. 35, no. 5, pp. 41-49, Sept. 2018, doi: 10.1109/MSP.2018.2825478.

[11] Adesh N.D., Renuka A., “Avoiding queue overflow and reducing queuing delay at eNodeB in LTE networks using congestion feedback mechanism”,. Computer Communications,2019, vol.146, 15 October 2019, doi: 10.1016/j.comcom.2019.07.015

[12] M. A. Panhwar, K. A. Memon, A. Abro, D. Zhongliang, S. A. Khuhro and S. Memon, "Signboard Detection and Text Recognition Using Artificial Neural Networks," 2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, 2019, pp. 16-19, doi: 10.1109/ICEIEC.2019.8784625.

[13] K. P. Seng, L. Ang, L. M. Schmidtke and S. Y. Rogiers, "Computer Vision and Machine Learning for Viticulture Technology," in IEEE Access, vol. 6, pp. 67494-67510, 2018, doi: 10.1109/ACCESS.2018.2875862.

[14] Clearly Explained: 4 types of machine learning online available : https://towardsdatascience.com/clearly-explained-4-types-of-machine-learning-algorithms-71304380c59a Last accessed : 12-June -2020.

[15] Boutaba, R., Salahuddin, M.A., Limam, N. et al. A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. J Internet Serv Appl 9, 16 (2018). https://doi.org/10.1186/s13174-018-0087-2

[16] Vandana Kushwaha, Ratneshwer Gupta, “Congestion control for high-speed wired network: A systematic literature review”, Journal of Network and Computer Applications, Vol. 45, PP- 62-78, 2014, Available: https://doi.org/10.1016/j.jnca.2014.07.005

[17 K. Xiao, S. Mao and J. K. Tugnait, "TCP-Drinc: Smart Congestion Control Based on Deep Reinforcement Learning," in IEEE Access, vol. 7, pp. 11892-11904, 2019, doi: 10.1109/ACCESS.2019.2892046.

[18] ns3, ns-3 Model Library, [online] available: https://www.nsnam.org/docs/release/3.30/models/html/index.html Last accessed date: 12-june-2020

[19] Naive Bayes Classifiers, [online] available: https://www.geeksforgeeks.org/naive-bayes-classifiers Last accessed date: 13-june-2020