FUDMA Journal of Sciences (FJS)
ISSN online: 2616-1370
ISSN print: 2645 - 2944
Vol. 3 No. 2, June, 2019, pp 77 - 89

AUTONOMOUS NETWORK MONITORING USING COLLABORATIVE LEARNING FOR DISTRIBUTED TRAFFIC CLASSIFICATION

[1]Joseph, S. B., [1]Dada, E. G. and [2]Yakubu, H. J.

[1]Department of Computer Engineering, Faculty of Engineering, University of Maiduguri, Borno State, Nigeria
[2]Department of Mathematical sciences, Faculty of Science, University of Maiduguri, Borno State, Nigeria

Corresponding Author's Email: sjbassi74@unimaid.edu.ng

ABSTRACT

Conventional traffic monitoring methods are becoming less efficient as numerous applications are rapidly adapting to counteract attempts to identify them, which creates new challenges for traffic monitoring. Autonomous Distributed Network Monitoring (ADNM) scheme is a promising approach to address these challenges, nonetheless each ADNM node has its own limitation, such as adapting to concept drift and self-learning, hence needs to collaborate with other autonomic nodes to monitor the network efficiently. This paper, presents a collaborative learning and sharing structure among self-managed network monitoring nodes, ensuring interaction for efficient information exchange for distributed autonomic monitoring towards the achievement of global network management objectives. A machine learning algorithm for collaborative learning among distributed autonomic monitoring nodes is proposed. This algorithm is based on the concept of online incremental k-means traffic classification model. Experimental results using publicly available real network traffic traces shows that network nodes participating in collaborative learning performs better with a higher overall total average accuracy of about 4.5% over centralized monitoring nodes. The overall performance indicates that collaborative learning is a promising technique that can value-add to the overall distributed network monitoring performance.

Keywords: *Network Monitoring, Machine Learning, k-mean, Cooperative Learning, Traffic classification*

INTRODUCTION

In the autonomic concept of network management, each entity (a network node) has the autonomy to self-govern its behavior. At the same time, each autonomic entity can participate in a collaborative management process, which requires a synergy among distributed autonomic entities to monitor, analyze, and make decision to achieve global network objectives. Cooperative learning allows the sharing of network state information among network nodes to allow network nodes to make autonomic decision, reacting to network changes such as load variations, concept drift, availability, and continuously optimizing the network resources according to certain optimization mechanisms (Guardalben, *et al.*, 2012). Thus, improving system performance in terms of fast reaction to changes, flexibility by adapting to changes in network dynamics, and scalability by adjusting to changes in network size. A schematic comparison of conventional centralized network management scheme and autonomic distributed network monitoring is shown in Figure 1. Several research works have proposed different techniques for ADNM (in-network monitoring) to solve the problem of real time autonomic distributed in-network traffic aggregation (Prieto, *et al., 2009*), (Prieto and Stadler, 2007) and (Raz, *et al.*, 2010). Although these methods have successfully demonstrated the ability to monitor network aggregates in real time, they do not have provision for nodes to collaborate with each other.

Guardalben, *et al.* (2012) and Guardalben *et al.,* (2010) proposed bootstrapping, discovery and election of entities to solve the problem of cooperative learning among ADNM management nodes. This collaboration method among autonomic distributed network nodes is computational-intensive and is not suitable for online network monitoring. To have a scalable, real-time monitoring, an efficient light-weight and information-sharing autonomic systems are required. Consequently, for ADNM entities to collaborate and cooperate in an efficient manner, exchange of information and/or dissemination of local decisions between nodes is paramount. Specifically, our research is concerned with the following questions:

1. How should information/knowledge be shared among distributed autonomic network nodes?
2. How can collaborative learning among autonomic network nodes help to improve network-wide monitoring (traffic classification)?
3. What information/ knowledge should be shared among nodes to enhance their knowledge base for distributed traffic classification?

In this paper, an algorithm which analyzes the effect of information exchange among nodes on the overall classification accuracy is proposed. This work is an extension of our earlier work in Joseph, *et al.,* (2015), that illustrated the feasibility of cooperative learning among autonomous network monitoring nodes. Precisely, bi-class distributed traffic classification was implemented using labeled flow instances as the sharing mechanism for cooperative learning. In this work, we proposed a scheme for multi-class traffic classification with two sharing mechanisms (labeled flow instances and newly created clusters) to show the adaptability and robustness of our monitoring technique for present day networks. Network streams are classified online as they arrive and the classification nodes are updated incrementally.

We tested our proposed method on two schemes: sharing of re-training flow instances in the form of information and sharing of clusters in the form of generative model on incremental k-means. Our proposed system has been applied to UNIBS (Unibs, 2009) and Cambridge (Moore, Zuev and Crogan, 2005) real network datasets in order to evaluate the performance of such system. Our proposed system is able to classify network traffic online with an average accuracy of 88.40% on UNIBS and 93.29% on Cambridge datasets, with an average cumulative improvement of about 4.5% over the non-sharing (centralized) approach. The results show that cooperative learning is a significant part of ADNM monitoring by demonstrating efficiency in terms of adaptability among nodes at a much lower overhead. The remainder of this paper is structured as follows. Section 2 introduces related works on present/future networks and Autonomous Distributed network management schemes. Section 3 presents online traffic classification. Section 4 analyzes the experimental results. Conclusion is in Section 5.

RELATED WORK

The present/future network as envisioned by International Telecommunication Union (ITU) to be packet based network capable of providing telecommunication with the ability to use multiple broadband, QoS-enabled transport technologies with self-standing related functions from the underlying transport related technologies. The fundamental characteristics of the present day networks originates from the problems faced by network administrators today. These problems include provision of services over broadband accesses;merging of different network services emerging peer-to peer and broadcast services and the desire of consumers to access services everywhere (Pirhadi, *et al.,* 2009) and (Alalousi, *et al.,* 2016).

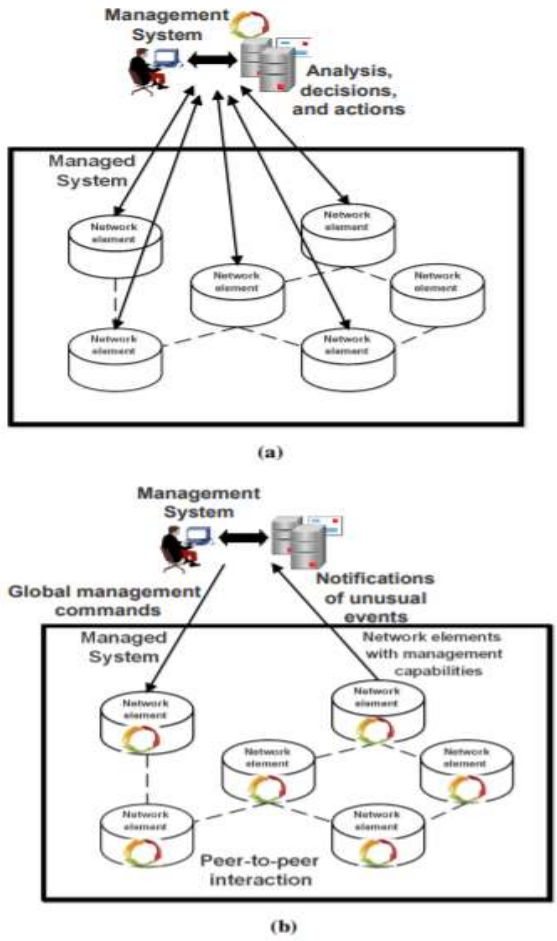


Fig. 1. Conventional network management (a) & Autonomic distributed network monitoring (b)

Some of the main characteristics of present networks include large data variety, volume, veracity and velocity (4Vs), from different sources having skewed class distribution (Olmezogullari and Ari, 2013) and concept drift (Deshpande, 2018). In view of the characteristics, management paradigms of embedding network management functions into the network itself has been proposed, e.g. Kephart and Chess (2003) and Lu, Zhou and Song (2010) towards easier management of networks, automatedly and with reduced complexity. Furthermore, software defined networking (SDN) (Kim and Feamster, 2013) such as Openflow (McKeown, *et al.*, 2008) has also been proposed for improving the efficiency of control and operations of autonomic network management (Tsagkaris, *et al.*, 2015). Several research works have also proposed different monitoring schemes for different network architecture over the years. Stadler *et al.* (2008) has outlined principles for decentralized monitoring using spanning trees and gossiping methods. This technique requires large amount of memory causing a high overhead. Raz *et al.* (2010) has presented an insight of network monitoring algorithms and also presented several monitoring algorithms that utilize monitoring paradigm. Cooperative learning as a strategy for collaboration among network nodes for various different network tasks has been proposed by different authors. Lee *et al.* (2008) presented cooperative learning for network load balancing, multi-class classification using cooperative learning was presented in (Xia and Ying, 2010), (Modi and Shen, 2001) and (Shi, *et al.*, 2009) proposed a collaborative multi-agent learning for distributed classification. Data stream mining for aggregate computation and prediction using cooperative online learners was presented in Canzian and van der Schaar (2014). However, these techniques are not for autonomous distributed online network monitoring. The primary aim of collaborative learning among network nodes is information dissemination towards the achievement of global network objectives with minimal overhead. Communication framework for group communication using IP multicasting was presented in Schönwälder (1996) and Parnes, Synnes and Schefstrom, (1999). However, this technique suffers the problem of flexibility during reconfiguration with changes in applications or network demand. Lee *et al*. (2008) presented cooperative learning for network wide load balancing using reinforcement learning for autonomous heterogeneous networks. However, this technique is not for distributed network performance monitoring. A flooding technique for information dissemination in distributed network system was presented in (Chi, *et al.*, 2007). Other techniques proposed for information dissemination include use of epidemic based method for multicast operation for mobile ad hoc networks (Genc and Ozkasap, 2007), cluster-based data dissemination scheme for wireless sensor networks (Chen, Ma and Salomaa, 2008) and probabilistic dissemination protocol for mobile wireless ad-hoc networks (Drabkin, *et al.*, 2007). In the case of DNM monitoring, Guardalben *et al.* (2012) presented a hide and seek method for information propagation. This technique though successful, is probabilistic in nature and computational intensive. Hence, it suffers high overhead and is not suitable for

online monitoring. This paper identifies the limitations of conventional centralized monitoring schemes for present day networks. From the machine learning point of view, several traffic classification methods can be employed for traffic monitoring. Our work focuses on using semi-supervised data stream mining strategy to address the issues of distributed network monitoring specifically cooperative learning among autonomous network nodes. The motivations are in three folds:

1. Data stream mining models can learn from both labeled and unlabeled data.
2. Data stream mining specifically k-means clustering algorithm allows for sharing among nodes.
3. Lastly, it is light weight and computationally simple, as such can be used for online network traffic monitoring. The uniqueness of our model is cooperative sharing of information/knowledge amongst autonomous network traffic monitoring nodes. In specific terms, we designed a cooperative learning algorithm for distributed network classification using incremental k-means clustering technique of data stream mining. This paper extends our proposal in Joseph *et al.* (2015), where we introduced and proved the feasibility of our cooperative learning concept among autonomous distributed network monitoring scheme.

ONLINE NETWORK TRAFFIC CLASSIFICATION WITH COOPERATIVE LEARNING

The characteristics of collaborative learning is essential for distributed network management system, where the classical centralized approaches cannot be used. Liakopoulos and Zafeiropoulos (2009) has presented the need for proper dissemination of knowledge among autonomic network nodes towards the achievement of autonomic self-management functions. This exchange with peers is to acquire more information about the overall network state. The Conceptual block diagram of our proposed system is illustrated in Figure 2.

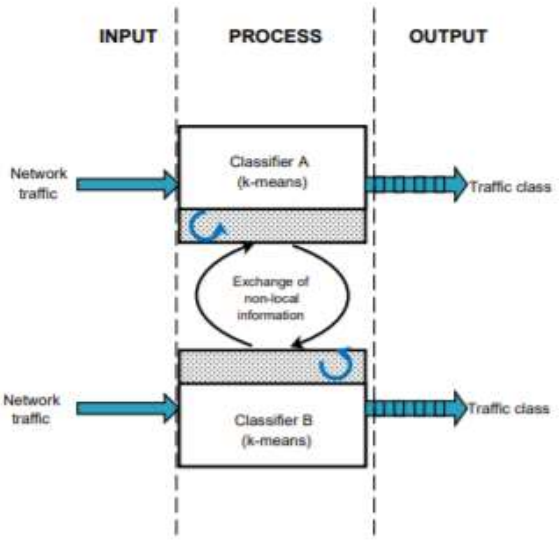


Fig. 2. Conceptual Block diagram

Network traffic are sensed by different network nodes. Possible functionality include the sharing of labeled data for retraining, sharing of newly formed clusters and/or sharing of the entire cluster model of nodes. This sharing is to improve their classification model states for monitoring network performance through in-network traffic classification. In order to adapt to new information through sharing, conventional traffic classification techniques such as port-based and supervised machine learning-based traffic classifiers are not suitable for online traffic classification since the learning mechanism is done off-line. In addition , both techniques are based on one-time learning, which suffers from reduced accuracy during the occurrence of concept drift in network traffic (Loo and Marsono, 2015). Incremental learning algorithms such as (Martínez *et al.*, 2011) and (Elwell and Polikar, 2011) overcome these shortcomings of the one-time learning algorithms by continuously adapting to new knowledge.

Distributed Network Monitoring with Cooperative Learning

To support the distributed autonomic network monitoring, we propose a network traffic classification paradigm with cooperative learning capabilities. This is to enhance the monitoring capabilities of the autonomic DNM nodes. The system architecture consist of three distinct layers, each performing a different successive task towards the achievement

of global management objectives. These layers are 1) pre-processing layer, 2) classification with cooperative learning layer and 3) decision layer. Each of these layers is segmentable and flexible. This is to make the system adaptable such that changes in any layer does not influence other layers. Details of layers is as presented in our earlier work (Joseph *et al.*, 2015).

k-mean Traffic Classification with Incremental Learning

This section reviews the traffic classification algorithm used in this work. A k-means clustering with the capability to incrementally learn from both labeled and unlabeled data (Loo and Marsono, 2015) is used in this work. This approach takes online traffic features information as an input to build the clusters model (or clusters). The architectural overview of the classifier is presented in Figure 3.

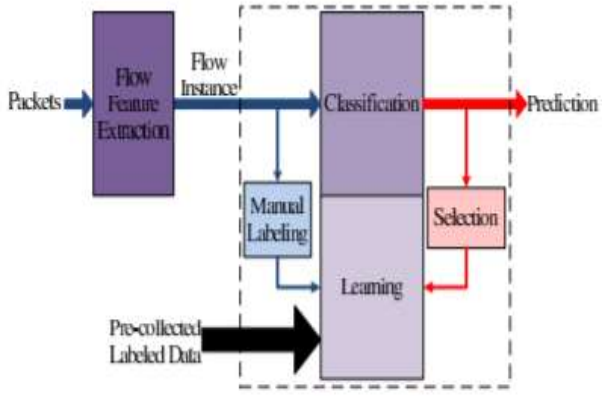


Fig. 3. Overview of k-means classifier architecture (Loo and Marsono, 2015)

The cluster model classifies each incoming data stream instance and determines the prediction confidence. The algorithm is divided into three sub-sections: off-line pre-training, online classification and training, and cluster reduction process. The pre-training stage uses a k-means to cluster labeled training set into k-base clusters. Using k-nearest neighbor method, a label can be predicted by the label of the cluster nearest to the sample. New clusters are formed and existing cluster are merged with successive inputs as a new concept is observed. If both of the nearest clusters are of the same class, the sample will be merged into the nearest cluster. A labeled sample requires the current model (after the merger) to be re-evaluated and modified. If the predicted label is true and it was not being trained in the above mentioned step, then a new cluster which consists of the sample will be added to the model. Otherwise, the sample is unmerged from the cluster. New cluster which consists of the sample will be added to the model. Algorithm 1 shows the classification process. Classification is initiated upon receiving an incoming traffic data stream instance x_i and the prediction confidence L_0, which is set by default. Let C1 and C2 be two nearest clusters, yC_i be the class of C_i , RC_i be the radius of C_i , and ϕC_i be the centroid of C_i . If $yC_1 = yC_2$, L_0 is incremented to L_1, where xi in C_i for yC_1 with more than an instance or xi $\approx \phi C_i$ for C_i with only an instance and the condition $yC_1 = yC_2$ is satisfied, then L_1 will be incremented to L_2. Instances are classified only if they achieve a confidence level L_2, as this is to minimize false learning. A periodic cluster reduction is initiated to eliminate outmoded and unutilized clusters to reduce memory footprint and classification time (Loo and Marsono, 2015). The k-means incremental clustering allows in-network cooperative learning and sharing of information among nodes. In this paper, labeled retraining data or newly formed clusters in the form of new knowledge can be shared among nodes.

Sharing Schemes

The sharing is performed based on two conditions. First, each node on receiving new labeled instance from network administrator will train itself and also share the entire received data with neighboring node. Secondly, a new labeled instance is shared only when the labeled instance is considered new for that node (i.e if a new cluster is created). This sharing is done based on Algorithm 2. The classification and learning are initiated upon receiving incoming data stream instances as in Algorithm 1. Cooperative learning is initiated by the host node upon:

Algorithm 1 Incremental k-mean Traffic Classification (Loo and Marsono, 2015)

1: x_i : Incoming data stream
2: C_1,C_2: First and second nearest cluster from xi
3: yi ,y 0 i : True and predicted labels for xi
4: ts: Timestamp for stored clusters
5: Pre-Training
6: Generate k-cluster using pre-collected data
7: Summarize k-cluster into clustering Feature, CF 8: Store clusters in time-series and set timestamp to 0
9: Classification & Learning
10: while new xi **do**
11: calculate C_1 and C_2
12: increase timestamp of C_1
13: compute confidence level
14: **if** (*confidencelevel* $\geq$ 2) **then**
15: merge x_i to C_i
16: **end if**

17: **end while**
18: **Cluster Reduction (Periodically)**
19: set ts = 0
20: **while** totalcluster ≥ user − definedthreshold, rk **do**
21: **if** timestamp = ts **then**
22: increase t_s by 1
23: **end if**
24: **end while**

1. Case 1: Receiving a labeled data instance for retraining. The host retrains its model and at the same time shares the labeled instance with neighboring nodes.

2. Case 2: The second case involves first training of model with received labeled data instance by the host node. Upon identifying new instances (for clusters creation) after retraining with received labeled data instance, the host shares newly identified instances with the neighboring node.

Algorithm 2 Proposed Cooperative Learning Algorithm

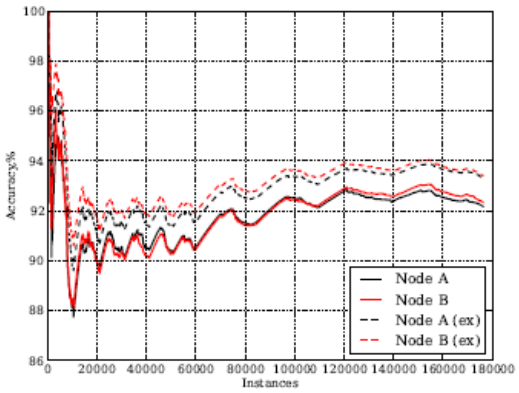
1: x_i: Incoming data stream
2: *M:* Initial training data
3: C_1,C_2: First and second nearest cluster from x_i
4: y_i,y_i: True and predicted labels for x_i
5: x_j ,y_j : Labeled data from nearest neighboring node
6: **while** new x_i **do**
7: y_i = *classify(M; x_i)*
8: **if** *($x_i$islabeled)* **then**
9: y_i *is known*
10: *retrain(M; x_i; y_i)*
11: *share(x_i; y_i)* to nearest neighbor
12: **end if**
13: **end while**
14: **Re-training with new knowledge**
15: **if** *((x_i; y_i)received)* **then**
16: *retrain(M; x_j ; y_j)*
17: **end if**

EXPERIMENTAL RESULTS AND DISCUSSION

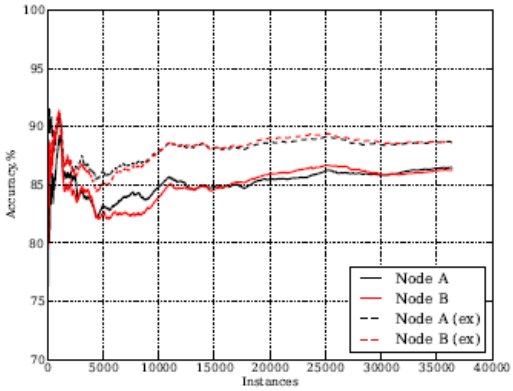
Datasets

In order to evaluate the performance of our proposed scheme, we implemented our proposed method using real network datasets. The Cambridge (Moore, Zuev and Crogan, 2005) and UNIBS (Unibs, 2009) datasets are used for the experiment.

The Cambridge dataset is an Internet traffic dataset captured from the University of Cambridge network. On the other hand, the UNIBS dataset was collected for three successive working days at the edge router of the campus network of the University of Brescia. The UNIBS dataset is in pcap format accompanied with its ground-truth. We pre-processed the traces extracting only online features with first 5 packets of each observed flows as these packets are sufficient to extract network statistics. With the provided ground truth labels, we labeled the flow and classified it into 5 classes (Web, Mail, P2P, SKYPE and MSN). For the Cambridge dataset, only the attributes that are in continuous form are selected from the total of 248 attributes (Jamil, *et al.*, 2014). The details of these datasets are summarized in Table 1.



(a)



(b)

Fig. 4. Cumulative accuracy plots for data sharing (a) Cambridge dataset (b) UNIBS dataset

Table 1: Datasets Statistics

	Cambridge		UNIBS
	Original	Selected	Extracted
# attributes	284	11	11
#Classes	12	10	6
#instances	397,152	397,030	81,753

Table 2: Model Parameters

No	Description	Symbol	Value
1	Percentage Labeling (%)	P	10
2	Chunk size	-	1000
3	Number of micro-model	B	10
4	Desired number of cluster	K_d	100
5	Boundary threshold	-	1

Experimental Setup

The model parameters for network nodes used in our experiment are as presented in Table 2, unless specified otherwise. In our experimental setup, the first chunk of data (first 1000 instances) are treated as pre-collected flows and they are used for model initialization. The remaining data are randomly labeled for different percentage P. The accuracy of the proposed model is verified using the interleaved test-then-train method where the data were first tested before being trained incrementally (Bifet, *et al.*, 2010). Each experiment was repeated 100 times, and the total cumulative average performance indicators are recorded and reported in this paper. The performance indicators used in this paper are the accuracy, cumulative accuracy. Accuracy refers to the accuracy of each chunk, while cumulative accuracy is the cumulative accuracy after the classification of each chunk. The experiments were performed using C++ programming language.

Case 1 (Sharing Labeled Data)

The accuracy plots of individual nodes A & B for Cambridge and UNIBS datasets are presented in Figure 5 and Figure 6, respectively. The figures show successive accuracy of nodes on each chunk with and without cooperative learning. In both plots, the nodes with cooperative learning capabilities demonstrate a higher accuracy for each chunk of data as compared to nodes without cooperative learning. Figure 4 shows the composite cumulative plots for nodes A & B on Cambridge and UNIBS datasets for models with and without cooperative learning, respectively. Both plots shows significant classification accuracy on each node with cooperative learning. The improvements in accuracy for nodes with cooperative learning ability are as a result of reinforcement from sharing with neighboring node.

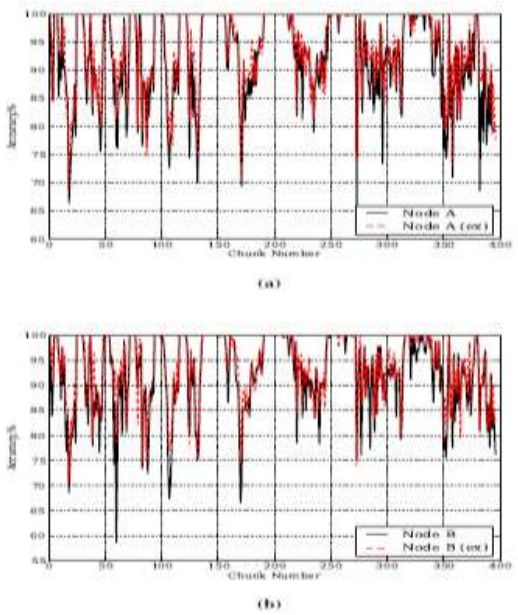


Fig. 5: Accuracy plots for Cambridge dataset (a) Node A without sharing capability & A(ex) with sharing capability (b) Node B without sharing capability & B(ex) with sharing capability

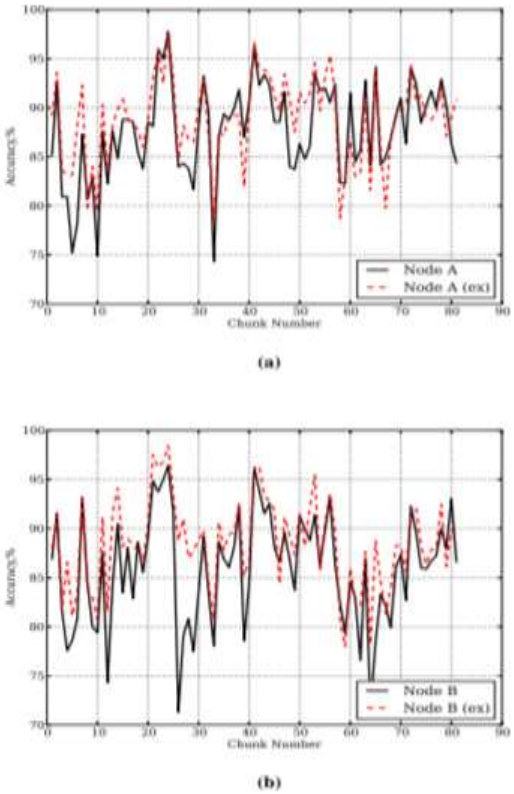


Fig. 6: Accuracy plots for UNIBS dataset (a) Node A without sharing capability & A(ex) with sharing capability (b) Node B without sharing capability & B(ex) with sharing capability.

Case 2 (Sharing Cluster Instance)

In this section we present the results of sharing new received knowledge in the form of instances for new cluster creation between monitoring nodes. The individual nodes accuracy plots for Cambridge dataset are presented in Figure 7, while UNIBS dataset is presented in Figure 8. The plots shows notable improvements on nodes with cooperative learning abilities over those without such capability.

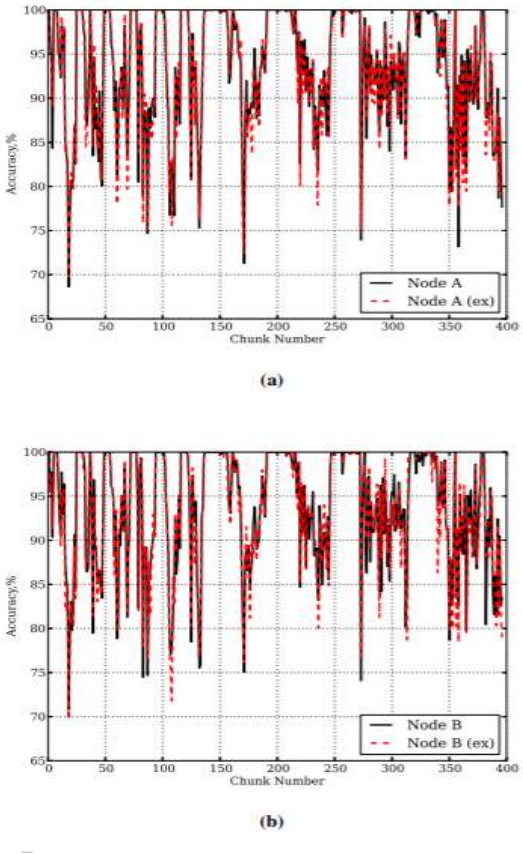


Fig. 7: Accuracy plots for Cambridge Dataset: (a) Node A without sharing capability & A(ex) with sharing capability; (b) Node B without sharing capability & B(ex) with sharing capability

The composite cumulative plots for both Cambridge and UNIBS datasets are presented in Figure 9. Both plots show higher classification accuracy on each node with cooperative learning compared to individual classification without cooperative learning. The improvement in accuracy is due to the fact that the base knowledge of the individual nodes has been complemented with the shared knowledge of the neighboring node. Node A(ex) represents accuracy of Node A after exchanging information with B.

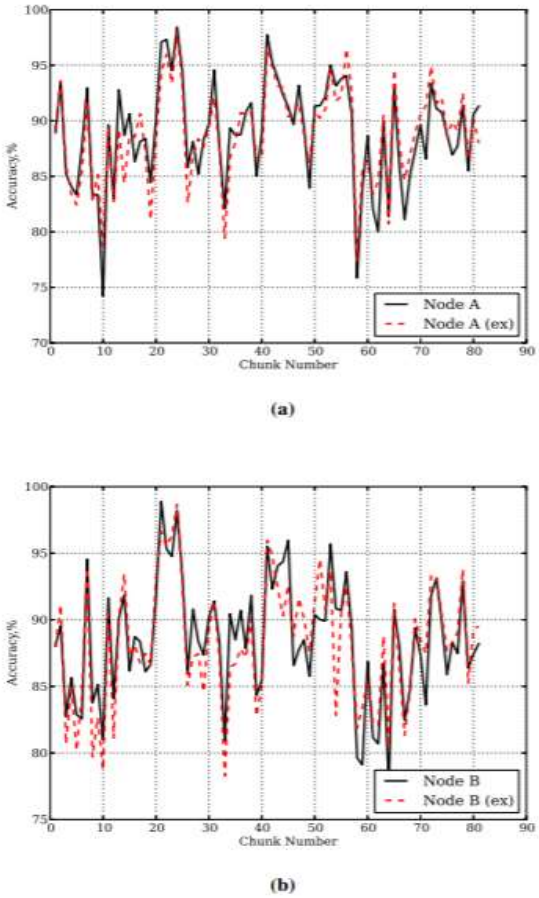


Fig. 8: Accuracy plots for UNIBS Dataset: (a) Node A without sharing capability & A(ex) with sharing capability; (b) Node B without sharing capability & B(ex) with sharing capability

Impact of Percentage Labeling on Performance Accuracy

This subsection, analyzes the outcome of changing certain algorithm parameters on the total system performance. The experiments are conducted by changing labeling parameter and fixing the others. Figures 10 and Figures 11 shows how labeling percentage, P affects the accuracy of performance on UNIBS and Cambridge datasets, respectively. Increasing the amount of labeled flow instances P provides more class information to the

classification model for improved accuracy. In both data sharing and cluster sharing, there is a significant improvement in classification accuracy for nodes with sharing capability and those without the sharing capability.

Performance

This subsection presents the overall performance of the proposed algorithm for both schemes. Although, in both cases significant improvement is observed by data/knowledge sharing, the accuracy for sharing only new instances is lower. This is because newly found instance to a node say A might not be new to the neighboring node say B. Hence, the sharing of the entire training data is more significant. Table 3 and Table 4 presents average performance on Cambridge and UNIBS datasets, respectively for a single experiment. The summary for cumulative average performance after 100 experiments is presented in Table 5 and Table 6. The performance of the proposed system shows an improvement of about 4.5% over the centralised system.

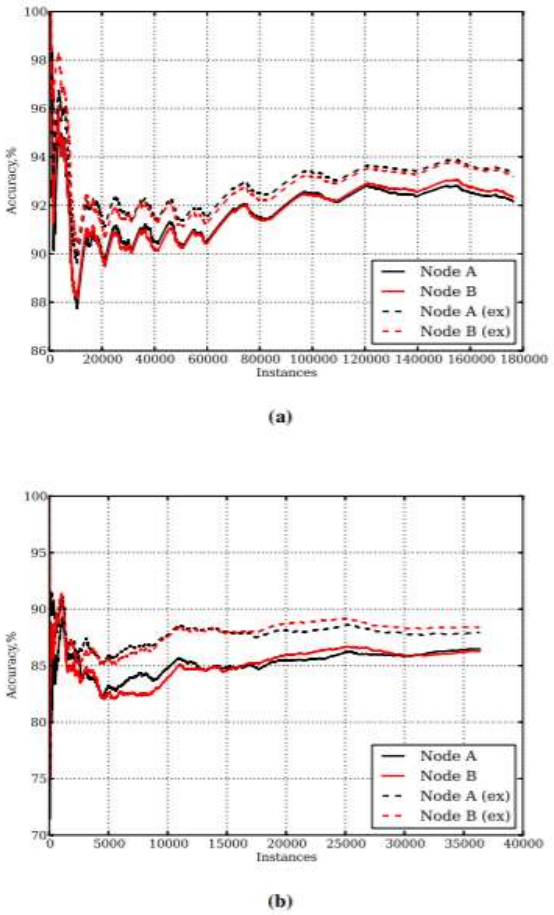


Fig. 9: Cumulative accuracy plots for cluster sharing (a) Cambridge dataset (b) UNIBS dataset

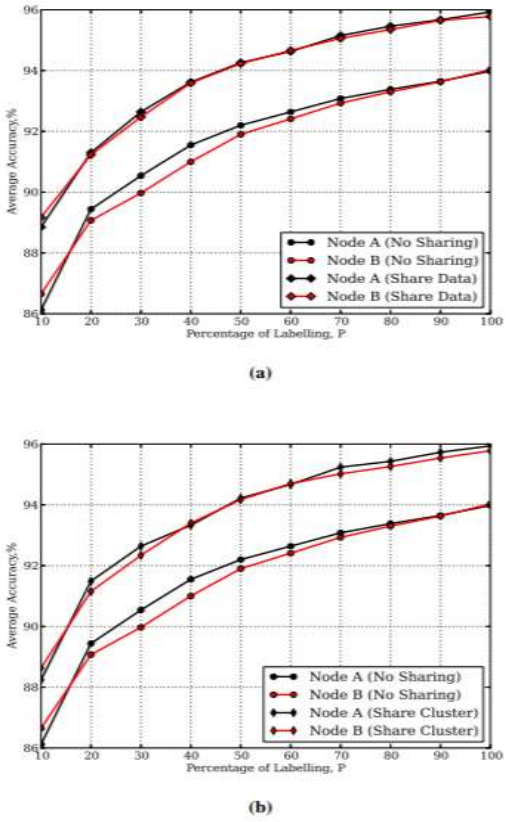


Fig. 10: The impact of percentage of flow instance labeling, P on accuracy on UNIBS (a) sharing data (b) sharing cluster

Table 3: Average performance results for Cambridge Dataset

Cambridge Dataset		
Nodes	Share Data Accuracy (%)	Share Cluster Accuracy (%)
A	93.3	93.29
B	93.39	93.18

Table 4 Average performance results for UNIBS Dataset

Cambridge Dataset		
Nodes	Share Data Accuracy (%)	Share Cluster Accuracy (%)
A	88.68	87.90
B	88.63	88.37

Table 5 Cumulative Average performance results for Cambridge Dataset

Experiment	Nodes without CL	Nodes with CL	
	Total Average Accuracy (%)	Total Average Accuracy (%)	Improvement (%)
Share Data	90.81	95.11	4.30
Share Cluster	90.81	94.98	4.17

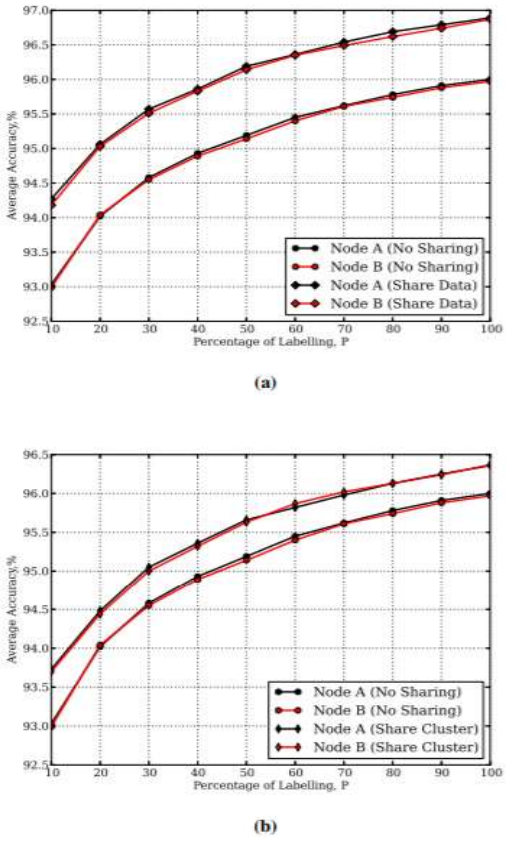


Fig. 11: The impact of percentage of flow instance labeling, P on accuracy on Cambridge (a) sharing data (b) sharing cluster

Table 6 Cumulative Average performance results for UNIBS Dataset

Experiment	Nodes without CL Total Average Accuracy (%)	Nodes with CL Total Average Accuracy (%)	Improvement (%)
Share Data	87.35	91.68	4.33
Share Cluster	87.35	91.59	4.24

CONCLUSION AND FUTURE RESEARCH

This paper proposes collaborative learning for autonomous distributed network monitoring based on online traffic classification with incremental learning capabilities. The system classifies network traffic using semi-supervised k-means clustering technique. In contrast to existing incremental k-means classification methods, our technique explores the capabilities of collaborative learning towards improving classification efficiency and accuracy among classification nodes. Furthermore, the goals for autonomous monitoring includes: reduction of human participation in management, reduction of overhead cost as a result of data storage, timely notification of system perturbation to avoid attacks or failure. Several open research issues remain regarding autonomous distributed network monitoring using collaborative learning. The advantage of k−mean technique has been demonstrated by improved performance, however, it remains to be seen if other existing or modified machine learning classification techniques, such as Support Vector Machines (SVM) or k−nearest neighbors, will lead to further increase in performance in autonomic classification domain. There is also a need to investigate further the believe factor for adopting new shared knowledge, in addition to the selecting of more efficient parameters for self-configuration by various autonomic network nodes. The performance of our technique was estimated with real network traces. Experimental results have demonstrated the increase in classification knowledge of nodes, improvement in classification accuracy of both network nodes participating in cooperative learning. The overall performance indicates that cooperative learning is promising to improve distributed network monitoring. Furthermore, we have showed that using cooperative learning in performance monitoring for distributed autonomic networks may provide new levels of functionality through proper cooperation and sharing of data/knowledge among autonomic nodes.

REFERENCES

Alalousi, A., Razif, R., AbuAlhaj, M., Anbar, M., Nizam, S. (2016). A preliminary performance evaluation of k-means, knn and em unsupervised machine learning methods

Bifet, A., Holmes, G., Kirkby, R., Pfahringer, B. (2010). Moa: Massive online analysis, The Journal of Machine Learning Research, 11, pp. 1601–1604.

Canzian, L., van der Schaar, M. (2014). A network of cooperative learners for data-driven stream mining'. In: Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference, pp. 2908–2912

Chen, C., Ma, J., Salomaa, J. (2008). Simulation study of cluster based data dissemination for wireless sensor networks with mobile sinks'. In: Advanced Communication Technology, ICACT 2008. 10th International Conference on. vol. 1, pp. 231–236.

Chi, C., Huang, D., Lee, D., Sun, X. (2007). Lazy flooding: a new technique for information dissemination in distributed network systems, IEEE/ACM Transactions on Networking (TON), 15, (1), pp. 80–92

for network flow classification', International Journal of Electrical and Computer Engineering (IJECE), 6, (2), pp. 778.

Deshpande, L. (2018). Concept drift identification using classifier ensemble approach, International Journal of Electrical and Computer Engineering (IJECE), 8(1):19-25, DOI: 10.11591/ ijece.v8i1.

Drabkin, V., Friedman, R., Kliot, G., Segal, M. (2007). Rapid: Reliable probabilistic dissemination in wireless ad-hoc networks'. Elwell, R., Polikar, R. (2011). Incremental learning of concept drift in nonstationary environments, Neural Networks, IEEE Transactions on, 22, (10), pp. 1517–1531.

Genc, Z., Ozkasap, O. 'Eramobile (2007). Epidemic-based reliable and adaptive multicast for manets'. In: Wireless Communications and Networking Conference, 2007. WCNC 2007. IEEE., pp. 4395–4400.

Guardalben, L., Gomes, T., Pinho, A., Salvador, P., Sargento, S. (2012). Nodes discovery in the in-network management

communication framework. In: Mobile Networks and Management. Springer, 2012. pp. 145–157

Guardalben, L., Sargento, S., Salvador, P., Mirones, V. (2010). A cooperative hide and seek discovery over in network management'. In: Network Operations and Management Symposium Workshops (NOMS Wksps), 2010 IEEE/IFIP., pp. 217–224

Jamil, H.A., Mohammed, A., Hamza, A., Nor, S.M., Marsono, M.N. (2014). Selection of on-line features for peer-to-peer network traffic classification. In: Recent Advances in Intelligent Informatics. Springer, 2014. pp. 379–390

Joseph, S.B., Loo, H.R., Ismail, I., Marsono, M.N. (2015). Cooperative learning for online in-network performance monitoring. In: 2015 EE 12th Malaysia International Conference on Communication, (MICC), 2015, pp. 213-218.

Kephart, J.O., Chess, D.M. (2003). The vision of autonomic computing, Computer, 2003, 36, (1), pp. 41–50.

Kim, H., Feamster, N. (2013). Improving network management with software defined networking, Communications Magazine, IEEE, 2013, 51, (2), pp. 114–119.

Lee, M., Ye, X., Marconett, D., Johnson, S., Vemuri, R., Ben.Yoo, S. (2008). Autonomous network management using cooperative learning for network-wide load balancing in heterogeneous networks. In: Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008, pp. 1–5

Liakopoulos, A., Zafeiropoulos, A. (2009). Autonomic monitoring and resource management using p2p techniques. In: Proceedings of Selected Papers, TERENA Networking Conference. Citeseer, 2009.

Loo, H.R., Marsono, M.N. (2015). Online data stream classification with incremental semi-supervised learning. In: Proceedings of the Second ACM IKDD Conference on Data Sciences. CoDS '15. New York, NY, USA: ACM, 2015. pp. 132–133. Available from: http://doi.acm.org/10.1145/2732587.2732614

Lu, X., Zhou, W., Song, J. (2010). Key issues of future network management'. In: Computer Application and System Modeling (ICCASM), 2010 International Conference on. vol. 11. (, 2010. pp. V11–649–V11–653.

Martínez.Rego, D., Pérez.Sánchez, B., Fontenla.Romero, O., Alonso.Betanzos, A. (2011). A robust incremental learning method for non-stationary environments', Neurocomputing, 74, (11), pp. 1800–1808.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J. (2008). Openflow: enabling innovation in campus networks, ACM SIGCOMM Computer Communication Review, 2008, 38, (2), pp. 69–74.

Modi, P.J., Shen, W.M. (2001). Collaborative multiagent learning for classification tasks'. In: Proceedings of the fifth international conference on Autonomous agents. ACM, pp. 37–38

Moore, A., Zuev, D., Crogan, M. (2005). Discriminators for use in flow-based classification. (Queen Mary and Westfield College, Department of Computer Science, 2005)

Olmezogullari, E., Ari, I. (2013). Online association rule mining over fast data. In: Big Data (BigData Congress), 2013 IEEE International Congress on Big Data, pp. 110–117

Parnes, P., Synnes, K., Schefstrom, D. (1999). A framewrk for management and control of distributed pplications using agents and ip-multicast. In: INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies Proceedings. IEEE. vol. 3, pp. 1445–1452

Pirhadi, M., Hemami, S.M.S., Khademzadeh, A., Chimeh, J.D. (2009). Mobility and qos control concern in next generation networks'. In: Advances in Communication and Networking. (Springer, 2009. pp. 89–104.

Prieto, A.G., Dudkowski, D., Meirosu, C., Mingardi, C., Nunzi, G., Brunner, M. (2009). 'Decentralized in-network management for the future internet'. In: Communications Workshops, 2009. ICC Workshops 2009. IEEE International Conference on, pp. 1–5

Prieto, A.G., Stadler, R. (2007). A-gap: An adaptive protocol for continuous network monitoring with accuracy objectives', Network and Service Management, IEEE Transactions on, 4, (1), pp. 2–12.

Raz, D., Stadler, R., Elster, C., Dam, M. (2010). In-network monitoring. In: Algorithms for Next Generation Networks. Springer, pp. 287–317

Schönwälder, J. (1996). Using multicast-snmp to coordinate distributed management agents'. In: Systems Management, 1996., Proceedings of Second IEEE International Workshop, pp. 136–141.

Shi Zhang, Y., pei Zhang, J., Yang, J., Yin, Z.W. (2009). 'Svms' cooperative learning strategy based on mas to data streams mining'. In: Internet Computing for Science and Engineering (ICICSE), 2009 Fourth International Conference, pp. 156–159.

Stadler, R., Dam, M., Gonzalez, A., Wuhib, F. (2008). Decentralized real-time monitoring of network-wide aggregates'. In: Proceedings of the 2nd Workshop on Large-Scale Distributed Systems and Middleware, ACM, 2008. pp. 7.

Tsagkaris, K., Logothetis, M., Foteinos, V., Poulios, G., Michaloliakos, M., Demestichas, P. (2015). Customizable autonomic network management: Integrating autonomic network management and software-defined networking, Vehicular Technology Magazine, IEEE, 2015, 10, (1), pp. 61–68.

Unibs: data sharing @ONLINE. 2009. http://www.ing.unibs.it/ntw/tools/traces/

Xia, Y., Ying, Y. (2010). A cooperative learning algorithm for multiclass classification'. In: Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference, vol. 3, pp. 223–226.