Extracting applicable features to troubleshoot computer networks for the first and second layers of the TCP/IP model using machine learning approaches

Jamal. Zarrinkamar [1], Seyed Mehdi. Hazrati Fard [2]
1- Pishtazan higher educational institute/ Computer Science and Engineering, Shiraz, Iran.
Email: pzp.zarrinkamar@gmail.com
2- Pishtazan higher educational institute/ Computer Science and Engineering, Shiraz, Iran.
Email: hazrati@cse.shirazu.ac.ir

ABSTRACT:
With the growth of services in the computer networks, expert operators are required to detect the active failures quickly and automatically. In the lack of expertise, using machine learning techniques is very applicable. Automatic failures detection needs a comprehensive dataset related to major network problems. In this way, data gathering and refinement are important to select and extract the best features.
In this paper, according to the first and second layers of the TCP/IP model, some useful and intelligently features are gathered in a unified dataset. Then, some classifiers such as decision tree, Support Vector Machine (SVM), k-Nearest Neighbors (kNN), Random Forest, Naive Bayes, and Logistic Regression are employed to test the applicability of the system. In fact, this machine is a model-based, supervised learning system. The results can better reveal the merits of the proposed approach to troubleshoot the network with minimum costs.

KEYWORDS: computer networks, machine learning, classification, data gathering, network troubleshooting.

1. INTRODUCTION

Nowadays, various services deploy on large computer networks with a wide range of areas, big variety and many types of equipment. So to deal with a problem, rapid troubleshooting is an important operation in the maintenance. Whereas a large network environment can include multi-vendor devices in different layers, in case of a problem, it would be very difficult to diagnose it [1].

In this realm, network operators have required a system which could detect the active failures quickly and automatically, especially when there is no access to an expert [2]. To learn this automatic system, the most important thing needs a comprehensive dataset of network features and related failures. In some points of views, preparing such a complete database is a very complicated process.

The first, there are too many concepts in each layer of the network, e.g., standards, protocols, and tasks related to the current problem of the network [3]. For example, depending on the type of encapsulation in the second layer or types of routing protocols in the network layer, there might be various problems.

The second, a network contains many types of equipment with several hardware and software resources that have been produced by various vendors [4]. For the sake of variety of protocols provided by a different vendor, we need to choose standard common protocols that are compatible with general devices and networks.

The main objective of this research is the automatic detection of active failures for physical and data link layers using machine learning techniques. But, to learn an intelligent algorithm, access to several annotated samples of available failures is essential [5]. Whereas such a complete dataset is not available, the first contribution of this work is to collect several pragmatic samples in various conditions and label them. The second contribution is to learn a powerful model to decide about the upcoming problems and suggest the type of failures.

In this research, some supervised learning approaches are used. So with a variety of failure classes, a generalized model can be obtained that can be extended the discerning to other items. Experimental results demonstrate the usefulness of the proposed approach to alleviating the costs of network troubleshooting. In this way, we did not use any tools such as Network Management Systems (NMSs) [6] or network log files, instead, some useful features based on computer network experiment in this realm are considered.

The rest of this paper is organized as follows. Section 2 summarizes some related work in this field. Section 3 investigates the proposed system in detail. In Section 4, several experiments conducting to evaluate the proposed system are discussed. Finally, the motivation and advantages of this paper are concluded in Section 5.

2. RELATED WORK

Splunk is a log analysis platform to collect, index and visualizes logs [7]. Another software is Logentries [8] that helps fast analysis of data from logs, process over log files that have been generated by Network Management Systems (NMSs) [9][10]. But, employing them needs great skills and domain knowledge to make efficient use of these products, such as alert rule finding and log format definition.

Yamanishi et al. [11] have proposed a technique to detect system failure from server system logs using a mixture of Hidden Markov Models (HMM). In another attempt, a log mining method was proposed using only frequency [12]. Zheng et al. [13] introduced a log preprocessing method of filtering important logs. Xu et al. [14] analyzed console logs of large-scale Hadoop systems and proposed a PCA-based anomaly detection method. An Intrusion Detection System (IDS) was proposed in [15] via frequent item-set mining approaches. Fu et al. [16] introduced an anomaly detection technique by learning normal system behavior with a finite state automaton. The aforementioned studies were focused on anomaly detection using unsupervised approaches.

SyslogDigest [17] targets the router system logs in a one-tier scale network; however, it just constructs digest information by grouping log messages within relevant routers. Kimura et al. [18] proposed a modeling and event extraction method of network log data using a tensor factorization approach; although, they did not focus on anomaly detection. Another method was proposed by Sipos et al. [19]. They presented a multiple-instance learning approach to predict equipment failures using system data log analysis. Tatsuaki Kimura et al [20], used the generation patterns of data log and automatic template extraction to solve a binary classification problem. They employed SVM with the Gaussian kernel [21] for their detection. Reidemeister et al. [22] also used these features and recurrent failure detections from the unstructured log in a decision tree classifier.

3. TECHNICAL WORK PREPARATION

This research proposes a method to find out the relationship between effective features and network failures. All of the activities are based on the TCP/IP model [3]. The project can consider in some steps:

3.1. Data Gathering

Most of the previous methods have used network log files and NMSs to make an automatic troubleshooting system, but here, knowledge of computer networks is employed to create a dataset.

The first step is data gathering by interviewing experienced experts and studying related books. This data includes all of the standards and protocols in the TCP/IP model [3].

Since the computer network field has several concepts, access to reliable data is the most important part of the work. In this phase, sometimes we need to ignore some data such as old standards or the special protocols which are not public. Here, an objective view for data gathering is considered.

Here, an object-oriented method [23] is required for data analysis and creating an optimum dataset. This method helps us to have an objective view of data. For example, the network can be an object that has some features. These features could be standards, protocols, and devices in each layer.

To prepare dataset we used some trouble tickets [24] which have been collected in the real computer network environment. The trouble ticket data describes the time and place of failure occurrence and the time of recovery [2].

3.2. Data Refinement

Data refinement is an important stage in order to maximize the benefit of big data assets. To be instrumentally useful, data should be considered as "answers" of questions to convert an abstract data model into an implementable data structure. Refining data include cleansing data, fixing or removing incorrect, incomplete, improperly formatted, or duplicated data. Then shape data structure by filtering, sorting, combining or removing features, and performing operations. In other words, data refinement eliminates redundant information and tuples with missed values [5].

For feature selection, the features which have more contribution in prediction of failure reasons in the network are picked. Existing irrelevant features can decrease the accuracy of the learned model. In other words, the selected attributes or a combination of them should have the most correlation with the label [5]. Here, correlation can be extracted from the covariance matrix and the dependency is found by semantic communication between each feature and label. Two types of correlations are considered: Pearson correlation and the Spearman correlation [25].

In the physical layer, there are some media, e.g. [4] fibers and devices, with different problems, so "Media" could be selected as a feature to detect the type of related physical problems. In the other hand, there are some related standards about every media that determine the media's attributes like their material, maximum distance, maximum bit rate, etc. So, some features can be selected such as “Standard”, “Media” as its material, “Maximum Distance”, “Maximum Rate” and also “Media Access Control Error” as a label. In this way, based on data refinement we could find out some semantic communications between features as a dependency. For example, reviewing the network failures in the physical layer indicates that just the values of the “Media” and “Maximum Distance” determine which failures could happen. So, we could remove “Standard” and “Maximum Rate” features from the dataset. So, some failures are common in all of the media and we can merge them with the special failures to reduce the failure classes. Finally, we select three features and "Location" as an identifier of the network to create the dataset.

In the Data Link layer we had thirteen features, “TT_Short_Descripton" as a short description of ticket, "TT_Impact_Assessment" as impact of the incident, "Affected_Service" as The service that was affected by the incident, "End_Line_Location_B" as the begin of the link, "End_Line_Location_B" as the end of the link, "Place_Of_Service" to determine situation of internal or external service, "Gateway_Statuse" to determine availability or unavailability of gateway, "Router_Redundancy" to determine using router redundancy, "Route" to determine using static or dynamic route, "Encapsulation" as a feature to determine the type of the encapsulation in the data link layer, "Ethernet-Channel" to determine using Link Aggregation Control Protocol [3] (LACP), "Port-Security" to determine running port-security over a port, and "STP" to determine using spanning tree protocol about loop avoiding and finally, "Logical Link Control Error" as a label. In addition, "Ethernet-Channel", "Port-Security" and "STP" features can use together in a link or only one of them is used.

Each of them is a special type of encapsulation that could be removed by merging them in the "Encapsulation" feature. By merging features, the classes should be merged too. Before merging, there were 24 classes of errors, but after that, they were reduced to 10 classes. Finally, we have eleven features in the data link layer.

Tables 1 and 2 show an overview of selected features and the correlation with their label according to data link and physical layers, respectively.

Table 1. Data Link Layer features

features	description	Pearson correlation	Spearman correlation
TT_Short_Descripton	Trouble Ticket Short Description	-0.09584	-0.10920
TT_Impact_Assessment	The impact of the incident	0.04076	0.03434
Affected_Service	The service that was affected by the incident	0.03707	0.04969
End_Line_Location_A	A-end of the link	0.39598	0.29283
End_Line_Location_B	B-end of the link	0.38530	0.27792
Place_Of_Service	There is internal or external service	-0.41167	-0.41596
Gateway_Statuse	Available or unavailable	0.50517	0.44328
Router_Redundancy	There is router redundancy	-0.10870	-0.10126
Encapsulation	Type of encapsulation in the data link layer	-0.07238	0.02986
Route	Static or dynamic	0.18220	0.30899
Logical Link Controller Error	Error in the data link layer	1	1

Table 2. Physical Layer features

features	description	Pearson correlation	Spearman correlation
Location	Location of network	0.31351	0.32412
Media	Media's material	0.20640	0.17756
Maximum Distance	Maximum distance of media support	-0.50703	-0.74498
Media Access Control Error	Failures in the physical layer	1	1

3.3. Data Statistics

There are 10 classes of failure in the Data Link layer of the TCP/IP model. These classes are shown in Table 3. According to the relation of errors, some of them are aggregated in a class. The last column shows the probability of errors in related class.

Table 3. Data Link Layer Classes

Class no.	description	Probability (%)
0	Port has been disabled by admin	50
	The incorrect IP address and subnet mask have been set on router's interfaces	50
1	Has been not set clock rate over DCE router's interface	35
	Has been not set encapsulation on router's interface	35
	Different encapsulation type have been used in link's interfaces or port has been disabled by admin	15
	Has been set an incorrect IP address or subnet mask on router's interfaces	15
2	Port has been not assigned to LACP	35
	Has been done incorrect port assignment in LACP	35
	There are two different protocol to make an Ethernet channel	5
	Port has been disabled by admin	5
	The incorrect IP address and subnet mask have been set on router's interfaces	20
3	Port has been disabled by admin	10
	The incorrect IP address and subnet mask have been set on switch's VLAN and router's interface	25
	.1q encapsulation has been not run over the trunk	25
	Has been not allowed to pass VLANs in the trunk	25
	A switch receives packets from two different interfaces with the same source MAC address.	15
4	There are two different encapsulations to make a trunk(Inter-Switch Link (ISL) and .1q)	5
	There are two different administrative modes to make a trunk (access and trunk)	5
	Port has been disabled by admin	5
	The incorrect IP address and subnet mask have been set on switch's VLAN	15
	.1q encapsulation has been not run over the trunk	10
	Has been not allowed to pass VLANs in the trunk	30
	A switch receives packets from two different interfaces with the same source MAC address	15
	There is a maximum number of MAC address limitation when is set "port-security"	15
5	Port has been not assigned to LACP	45
	Has been done incorrect port assignment in LACP	45
	There are two different protocol to make an Ethernet channel	10
6	Incorrect VPI/VCI	5
	Has been not made PPPOE connection	45
	Username-password is incorrect	45
7	Circuit error	100
8	Has been set the incorrect IP address	25
	incorrect subnet mask	25
	The incorrect default gateway on the PC	25
	Switch's port does not have been set port-fast	10
	VLAN does not exist in the database incorrect subnet mask	5
	PC is not connected to correct VLAN	5
9	Has been not set static MAC address	50
	There is a limitation for the maximum number of MAC addresses	50

Also, there are 4 classes of failure in the Physical layer that are shown in Table 4 with the pertained probability of errors.

Table 4. Physical Layer Classes

Class no.	description	Probability (%)
0	Optical media failed	50
	Distance is more than standard	50
1	Distance is more than standard	25
	connector is failed	75
2	STM 1 module failed	100
3	Copper media failed	100

4. EXPERIMENTS

In this research some supervised learning classification approaches are employed to detect active failures: Classification and Regression Trees (CART) [26], Support Vector Machine (SVM), k-Nearest Neighbors (kNN), Random Forest, Naive Bayes, Logistic Regression and Linear SVM [5].

This section discusses the experimental results of the proposed system. To create training and test data 421 trouble ticket in the data link layer and 421 in the physical layer was used.

The used models were trained on the extracted features from the prepared dataset and to test the system we used computer network troubleshooting scenario which has been designed and implemented in "GNS3" [27] and " Cisco Packet Tracer" [28] simulators by Cisco Training Services [29].

GNS3 that is shown in Fig.1 is a graphical network simulator that allows designing complex network topologies. You may run simulations or configure devices ranging from simple workstations to powerful Cisco routers [27]. Also, Cisco Packet Tracer that is shown in Fig.2 is an innovative and powerful network simulation tool used for practice, discovery and troubleshooting [28].

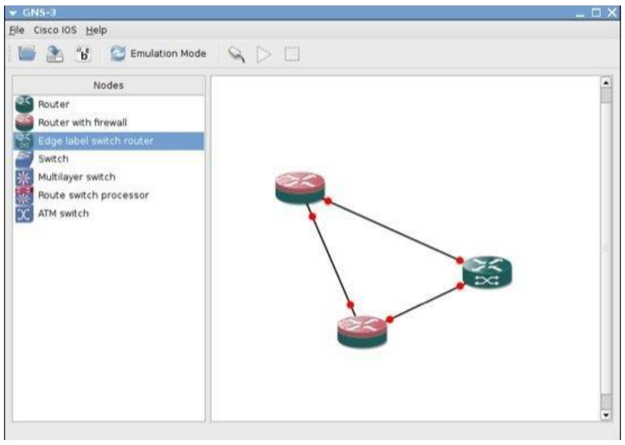

Fig.1. GNS3 simulator

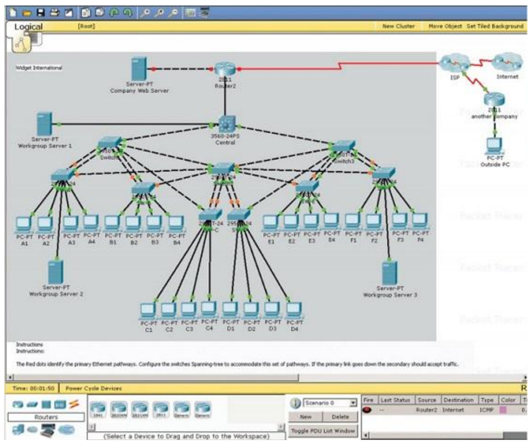

Fig.2. Cisco Packet Tracer simulator

To check the ability of collected dataset, the available functions in scikit-learn [30] are employed. Scikit-learn is a free machine learning library in Python include various classification, regression and clustering algorithms designed to interoperate with Python numerical and scientific libraries, i.e. NumPy [31] and SciPy [32].

To evaluate the results, accuracy is a common metric which needs some concepts:

True Positive (TP): positive samples which predicted correctly

True Negative (TN): negative samples which predicted correctly

False Positive (FP): negative samples which predicted incorrectly

False Negative (FN): positive samples which predicted incorrectly

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP} \quad (1)$$

Table 5 and 6 show the accuracy of the mentioned models that have been trained on the gathered dataset and tested on the generated samples of GNS3 and Cisco Packet Tracer.

Table 5. Classification results of Data Link Layer on the simulated network in GNS3.

Model	Accuracy (%)
Classification and Regression Trees	100.00
Random Forest	100.00
SVM	100.00
Naive Bayes	97.59
Logistic Regression	90.36
k-Nearest Neighbors	72.29

Table 6. Classification results of Physical layer on the simulated network in GNS3.

Model	Accuracy (%)
Classification and Regression Trees	100.00
Random Forest	100.00
Naive Bayes	100.00
SVM	98.61
Logistic Regression	98.8
k-Nearest Neighbors	91.88

5. CONCLUSION

This paper proposed an active failure detection system for the data link and physical layers. In this way, two separate datasets have been prepared by gathering and refining data to select the best features of each layer. Trouble tickets from some real network logs are employed for data gathering and make the dataset for the learning phase. Then, some machine learning algorithms are used to assess the system ability for error determination. To test the learned models some simulated networks by GNS3 simulator and Cisco packet tracer were employed.

Future work in this realm can be the investigation of active failures in upper layers of the TCP/IP model and trying to propose an automatic system to detect failures of such layers.

REFERENCES

[1] R. Mathonet, H. Van Cotthem, and L. Vanryckeghem, "DANTES: An Expert System for Real-time Network Troubleshooting," in Proceedings of the 10th International Joint Conference on Artificial Intelligence - Volume 1, 1987, pp. 527–530.
[2] N. Allen, Network maintenance and troubleshooting guide. Fluke, 2000.
[3] B. A. Forouzan and S. C. Fegan, TCP/IP protocol suite. McGraw-Hill Higher Education, 2002.
[4] Andrew S. Tanenbaum., Computer Networks. Pearson India, 2013.
[5] J. Han, M. Kamber, and J. Pei, Data mining : concepts and techniques. Elsevier Science, 2011.
[6] D. Comer, Automated network management systems : current and future capabilities. Pearson Prentice Hall, 2007.
[7] "SIEM, AIOps, Application Management, Log Management, Machine Learning, and Compliance | Splunk." [Online]. Available: https://www.splunk.com/. [Accessed: 18-Jan-2019].
[8] "Log Management &amp; Analysis Software Made Easy | Logentries." [Online]. Available: https://logentries.com/. [Accessed: 18-Jan-2019].
[9] "Network Fault Management, CA Spectrum® - CA Technologies." [Online]. Available: https://www.ca.com/us/products/ca-spectrum.html. [Accessed: 18-Jan-2019].
[10] "IBM - Compliance Software - Tivoli Compliance Insight Manager - Software." [Online]. Available: http://www-01.ibm.com/software/tivoli/products/compliance-insight-mgr/. [Accessed: 18-Jan-2019].
[11] K. Yamanishi and Y. Maruyama, "Dynamic syslog mining for network failure monitoring," in Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, 2005, pp. 499–508.
[12] C. Lim, N. Singh, and S. Yajnik, "A log mining approach to failure analysis of enterprise telephony systems," in Dependable Systems and Networks With FTCS and DCC, 2008. DSN 2008. IEEE International Conference on, 2008, pp. 398–403.
[13] Z. Zheng, Z. Lan, B. H. Park, and A. Geist, "System log pre-processing to improve failure prediction," in Dependable

Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on, 2009, pp. 572–577.

[14] W. Xu, L. Huang, A. Fox, D. Patterson, and M. I. Jordan, "Detecting large-scale system problems by mining console logs," in Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles, 2009, pp. 117–132.

[15] R. Vaarandi and K. Podi\cnš, "Network ids alert classification with frequent itemset mining and data clustering," in Network and Service Management (CNSM), 2010 International Conference on, 2010, pp. 451–456.

[16] Q. Fu, J.-G. Lou, Y. Wang, and J. Li, "Execution anomaly detection in distributed systems through unstructured log analysis," in Data Mining, 2009. ICDM '09. Ninth IEEE International Conference on, 2009, pp. 149–158.

[17] T. Qiu, Z. Ge, D. Pei, J. Wang, and J. Xu, "What happened in my network: mining network events from router syslogs," in Proceedings of the 10th ACM SIGCOMM conference on Internet measurement, 2010, pp. 472–484.

[18] T. Kimura et al., "Spatio-temporal factorization of log data for understanding network events," in INFOCOM, 2014 Proceedings IEEE, 2014, pp. 610–618.

[19] R. Sipos, D. Fradkin, F. Moerchen, and Z. Wang, "Log-based predictive maintenance," in Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining, 2014, pp. 1867–1876.

[20] T. Kimura, A. Watanabe, T. Toyono, and K. Ishibashi, "Proactive failure detection learning generation patterns of large-scale network logs," IEICE Trans. Commun., 2018.

[21] C. Cortes and V. Vapnik, "Support-vector networks," Mach. Learn., vol. 20, no. 3, pp. 273–297, 1995.

[22] T. Reidemeister, M. Jiang, and P. A. S. Ward, "Mining unstructured log files for recurrent fault diagnosis," in Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on, 2011, pp. 377–384.

[23] P. Beynon-Davies, "Object-Oriented Methods," in Information Systems Development, London: Macmillan Education UK, 1998, pp. 291–296.

[24] D. Zisiadis, S. Kopsidas, M. Tsavli, and G. Cessieux, Eds., "The Network Trouble Ticket Data Model (NTTDM)," Feb. 2011.

[25] "Pearson's and Spearman's Correlation," in An Introduction to Statistical Analysis in Research, Hoboken, NJ, USA: John Wiley & Sons, Inc., 2017, pp. 435–471.

[26] L. Breiman, J. H. (Jerome H. Friedman, R. A. Olshen, and C. J. Stone, Classification and regression trees.

[27] "GNS3 download | SourceForge.net." [Online]. Available: https://sourceforge.net/projects/gns-3/. [Accessed: 18-Jan-2019].

[28] "Download The Packet Tracer Simulator Tool &amp; Find Courses | Networking Academy." [Online]. Available: https://www.netacad.com/courses/packet-tracer. [Accessed: 02-Jun-2019].

[29] "Cisco Training Services - Cisco." [Online]. Available: https://www.cisco.com/c/en/us/training-events/training-certifications/training/training-services.html. [Accessed: 24-May-2019].

[30] "scikit-learn: machine learning in Python — scikit-learn 0.21.2 documentation." [Online]. Available: https://scikit-learn.org/stable/. [Accessed: 30-May-2019].

[31] "NumPy — NumPy." [Online]. Available: https://www.numpy.org/. [Accessed: 30-May-2019].

[32] "SciPy.org — SciPy.org." [Online]. Available: https://www.scipy.org/. [Accessed: 30-May-2019].