DOI: 10.2507/36th.daaam.proceedings.xxx

MODELS FOR EVALUATING EFFECTIVE IP TRAFFIC MANAGEMENT WITH APPLICATION OF ARTIFICIAL INTELLIGENCE

Irina Topalova* & Pavlinka Radoyska

This Publication has to be referred as: Topalova, I[rina] & Radoyska, P[avlinka] (2025). Title of Paper, Proceedings of the 36th DAAAM International Symposium, pp.xxxx-xxxx, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-xx-x, ISSN 1726-9679, Vienna, Austria
DOI: 10.2507/36th.daaam.proceedings.xxx

Abstract

The application of simulation models to assess the effective management of IP traffic requires the creation of appropriate network topologies, with supported functionality for building balanced and reliable traffic. The application of artificial intelligence methods for analysis and classification of statistical data determining effective management is emerging as an innovative modern approach to finding optimal solutions. But these systems do not provide a prescription for automated selection of changes in network topology/architecture to obtain better traffic efficiency. In this study, a method for automated prescription for improving the network structure is proposed, based on a database of current time parameters that determine the quality of remote connectivity. The method is developed with simulated network topologies, with the implementation of known protocols for alternative and fail-safe traffic. The measured time parameters, in different traffic regimes, serve as input training data for a proposed machine learning model. Depending on the results in the classification stage of newly received data, the model predicts the need for appropriate restructuring of the network topology and reconfiguration of the protocols. The obtained results are discussed in terms of accuracy and precision of the classification.

Keywords: IP traffic; IP network topologies; artificial Intelligence; machine learning.

1. Introduction

The most advanced network devices apply ML (Machine Learning) and AI (Artificial Intelligence) for threat detection, tracking and response, user behaviour analysis, network traffic analysis, automation in security operations and incident response. AI with the application of neural networks confidently enters the QoS mechanisms for prioritizing IP traffic to achieve better efficiency to reduce the effects of congestion [1]. In the training phase, the AI-based analytical module observes and learns the normal values of parameters. During the test phase, real-time traffic is intercepted and compared with normal behaviour. In search of plausibility in such systems, ML detects anomalies, identifies their common features and is trained on these data sets. They are analysed and classified according to the general ML model, enriching the information and creating a predictive model [2]. Various routine and repetitive tasks are intelligently designed and embedded in the modules of the software libraries. This modern and innovative functionality does not include analysis of

system behaviour in hybrid architectures, including the basic structures and protocols for balanced and fail-safe traffic, by providing alternative connections.

For these reasons, it would be appropriate to implement an adaptive training system with different parameters that determine effective traffic management. This would also allow for appropriate prescriptions for reconfiguring the system according to the results of additional training of the adaptive system for different network architectures.

State of the Art

Artificial intelligence (AI) methods, applied to adapt to changes in input data, dynamically updating the accumulated knowledge in order to obtain a quick and adequate response, are increasingly noticeable in many different tools in information technology. In this regard, it is reasonable, together with the various ML techniques developed in network devices, to integrate the analysis of a number of dynamically changing parameters of IP traffic such as the number of packets transmitted per second (Packets/sec), remote access time, etc. [3]. This would facilitate tracking the current state of the connection, classifying and reflecting various dynamically changing conditions.

There are a number of software platforms that offer similar functionality for monitoring the quality of connections to a device on the local network or to a remote server. For example, SolarWinds Ping Sweep [4] allows you to check the connectivity to a list or range of IP addresses. This allows you to assess the performance of both DHCP and DNS servers, as well as generate warning messages about a bad or missing connection. The software offers the ability to modify ping parameters such as ICMP Timeout, ICMP packet data portion, and Packet Time-To-Lives.

One of the features offered by Auvik's cloud console [5] is its Ping service, which checks for individual network endpoints from their location in the cloud infrastructure. The core network monitoring features of the Auvik system also provide continuous monitoring of SNMP-based devices. The Pingdom system [4] includes alerts that occur when ICMP response times drop. The system estimates response times for sites that are still accessible. This system also explores APIs for targeting alerts so that these notifications can be integrated into various custom applications.

An approach to proactive recommendation systems, proposed by authors in the field of road and manufacturing emergency prediction, is investigated in [6]. The information is collected from video clips, necessitating a hybrid approach that incorporates a neuro-fuzzy controller, a knowledge base, and deep learning. An algorithm for generating recommendations to prevent emergencies is proposed. This approach provides ideas for generating solutions in the design and management of communication networks based on the use of neural networks.

An approach for balancing the load of parallel production lines using neural networks is discussed in. The proposed solution accounts for system variations and trains the neural network to minimize order completion time. A simulator, based on previously collected information about the production process, is used [7]. The trained neural network recommends directing the production flow to one of the production lines. The proposed idea could be adapted for predicting and reorganizing traffic in communication networks.

At the same time, all these approaches do not offer automated classification of results to determine the quality of connectivity, nor do they implement options for dynamically adapting each possible classification to changes in network topology, to the assignment of different technologies and protocols, ensuring balancing and reliability of IP traffic.

In this study, a method for automated prescription for improving the network structure is proposed, based on a database of current time parameters that determine the quality of remote connectivity. The method is developed with simulated network topologies, with the implementation of known protocols, such as FHRP (First Hop Redundancy Protocol) and PVST (Per VLAN Spanning-tree) for alternative and fail-safe traffic. The measured time parameters, in different traffic regimes, serve as input training data for the proposed Random Forest machine learning model. Depending on the results in the classification stage of newly received data, the model predicts the need for appropriate restructuring of the network topology and reconfiguration of the protocols. The obtained results are discussed in terms of accuracy and precision of the classification. Also feature work is proposed.

2. Description of the proposed method

The main functional model in the study is presented in Figure 1. The input training data, after appropriate formatting, is fed to the input of a Random Forest model, including a different number of training trees. The training data represent the response times of ICMP (pings) from a specific device in the network to a remote destination, at different traffic loads

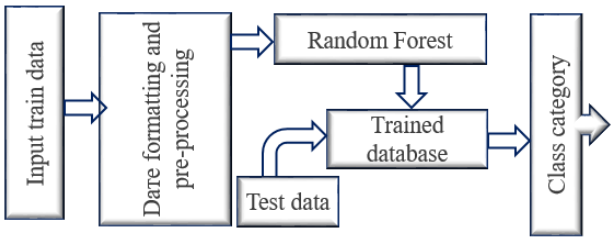
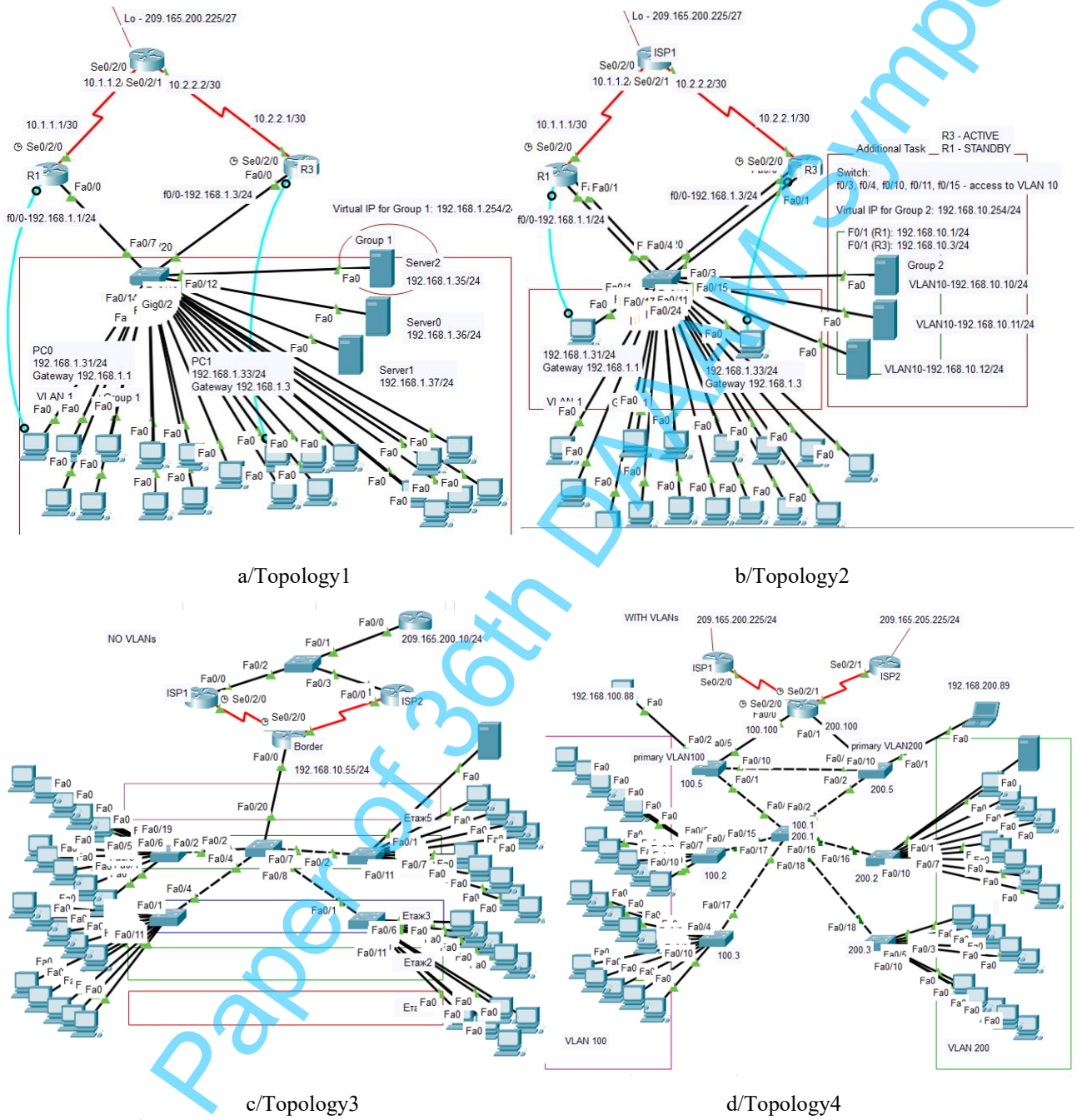


Fig. 1. Functional model flowchart

in the respective network. After receiving the trained data base, in the test phase, data from the response times in other situational moments are fed to it, in order to be classified by the trained Random Forest model.

2.1. Simulation of the tested network topologies

The simulated network topologies are presented in Figure 2 as follows. In Figure 2/a - Topology1 with FHRP protocol with one group, In Figure 2/b - Topology2 with FHRP protocol with two alternative groups, In Figure 3/c - Topology3 with one VLAN and in Figure 2/e - Topology4 with two VLANs (VLAN100 & VLAN200).



a/Topology1 b/Topology2

c/Topology3 d/Topology4

Fig. 2. Simulated network topologies: /a - Topology1 with FHRP protocol with one group, /b - Topology2 with FHRP protocol with two alternative groups, /c - Topology3 with one VLAN and /e - Topology4 with two VLANs.

The training data for the four topologies in Figure 2 is obtained and groped in *Class categories*, as follows:

- Average Ping response time for a single ping from the server in Topology 1 (Fig.2/a) to a remote address 209.165.200.225 and for a simultaneous, load-bearing, infinite broadcast ping *from 5* end devices to address 192.168.10.255 - *Class Category 1*;

- Average Ping response time for a single ping from the server in Topology 1 (Fig.2/a) to a remote address 209.165.200.225 and for a simultaneous, load-bearing, infinite broadcast ping *from 20* end devices to address 192.168.10.255 - *Class Category 2*;
- Average Ping response time for a single ping from the server in Topology 2 with two alternative FHRP groups and different roles of [(router R1 as active for Group1 - network 192.168.1.0/24 and router R3 as active for Group2-network 192.168.10.0/24) (Fig.2/b)] to a remote address 209.165.200.225 and for a simultaneous, load-bearing, infinite broadcast ping *from 5* end devices of Group1 to address 192.168.1.255 - *Class Category 3*;
- Average Ping response time for a single ping from the server in Topology 2 with two alternative FHRP groups and different roles of [(router R1 as active for Group1 - network 192.168.1.0/24 and router R3 as active for Group2-network 192.168.10.0/24) (Fig.2/b)] to a remote address 209.165.200.225 and for a simultaneous, load-bearing, infinite broadcast ping *from 20* end devices of Group1 to address 192.168.1.255 - *Class Category 4*;
- Average Ping response time for a single ping from the server in Topology 3 (Fig.2/c) to a remote address 209.165.200.10 and for a simultaneous, load-bearing, infinite broadcast ping *from 5* end devices in the only VLAN to address 192.168.10.255 - *Class Category 1-1*;
- Average Ping response time for a single ping from the server in Topology 3 (Fig.2/c) to a remote address 209.165.200.10 and for a simultaneous, load-bearing, infinite broadcast ping *from 20* end devices in the only VLAN to address 192.168.10.255 - *Class Category 2-1*;
- Average Ping response time for a single ping from the server in Topology 4 (Fig.2/d) to a remote address 209.165.200.10 and for a simultaneous, load-bearing, infinite broadcast ping *from 3* end devices in VLAN 100 to address 192.168.100.255 and 2 end devices in VLAN 200 to address 192.168.200.255 - *Class Category 3-1*;
- Average Ping response time for a single ping from the server in Topology 4 (Fig.2/d) to a remote address 209.165.200.10 and for a simultaneous, load-bearing, infinite broadcast ping *from 10* end devices in VLAN 100 to address 192.168.100.255 and 10 end devices in VLAN 200 to address 192.168.200.255 - *Class Category 4-1*;

For each of the categories defined in this way, 10 attempts/measurements of Average response time were made. To generate a large training set, additional values were added to these values in the range +-10% relative to the average value of the 10 measurements. In this way, for each class category, 30 combinations of 10 measurements were generated. That is, for *Class categories 1,2,3 and 4*, a total of 120 combinations and for *Class categories 1-1,2-1,3-1 and 4-1*, a total of 120 combinations of measured ping responses. Table 1 represents the first rows of 10 attempts for *Class category1*.

Attempt1	Attempt2	Attempt3	Attempt4	Attempt5	Attempt6	Attempt7	Attempt8	Attempt9	Attempt10	Class category
10	11	7	8	9	11	10	8	12	9	1
10.00	14.00	4.00	11.00	12.00	10.00	10.00	8.00	14.00	11.00	1
10	12	7	5	9	9	10	8	9	6	1
8	11	10	8	11	8	9	7	13	7	1
13	13	4	10	11	8	10	6	13	11	1
13	9	5	7	10	13	10	8	15	9	1
7	12	10	11	6	11	9	6	13	9	1
11	8	5	5	9	8	12	9	11	11	1
8	11	6	8	7	10	13	9	15	6	1
12	8	6	7	12	8	7	5	9	12	1
13	11	8	5	7	10	11	9	9	9	1
11	11	7	8	7	10	12	10	14	8	1
10	11	4	9	8	11	8	5	9	6	1

Table 1. Part of training data for *Class category1*

2.2. Training the Random forest machine learning model

To create the model, the *Colaboratory* cloud environment was used, with access to the necessary Python libraries [8]. For the purposes of the experiment, two Random Forest models were trained: *Random Forest1* for class categories 1,2,3,4 and *Random Forest2* for class categories 1-1,2-1,3-1,4-1.

	Random forest 1		Random forest 2	
Average response time when pinging from the server to a remote point [msec]	**Topology 1**	**Topology 2**	**Topology 3**	**Topology 4**
	FHRP-One group	**FHRP-Two groups**	**One VLAN**	**Two VLANs**
Infinite broadcast ping sent from 5 end devices	9,5 msec Class category1	1,83 msec Class category3	19 msec Class category1-1	11,37 msec Class category3-1
Infinite broadcast ping sent from 20 end devices	25,8 msec Class category2	15,4 msec Class category4	45,18 msec Class category2-1	2,5 msec Class category4-1

Table 2. Average response time when pinging from server to a remote point for all Class categories

To achieve and evaluate an effective ratio between model complexity, training phase duration and achieved accuracy in the validation/testing stage, each model was trained with 1 and 10 tree structures (estimators), respectively.

```
# Features and target variable
X = data [['Attempt1', 'Attempt2', 'Attempt3', 'Attempt4', 'Attempt5', 'Attempt6', 'Attempt7', 'Attempt8', 'Attempt9', 'Attempt10']]
y = data ['Class category']
# Split data into training (55% of all 120 combinations for Random Forest1 and Random Forest2) and 45% testing sets.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.45, random_state=42)
# Initialize RandomForestClassifier with n_estimators=10: This parameter specifies that the Random Forest will consist of 10 decision trees.
rf_classifier = RandomForestClassifier(n_estimators=10, random_state=42)
# Fit the classifier to the training data
rf_classifier.fit(X_train, y_train)
# Make predictions
y_pred = rf_classifier.predict(X_test)
# Calculate accuracy and classification report
accuracy = accuracy_score(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)
```

3. Experimental results

To determine whether test/validation data is effectively classified, according to the model's decision, we need to consider the classical estimation parameters of the model's accuracy. Namely, these are the parameters „Precision“, „Recall“ and „F1- score“. As F1-score takes into account both precision and recall and is based on a balance of the two, it is more appropriate to calculate this parameter according to the given equation 1 [10]:

$$F1-\text{score} = 2.\frac{\text{preciscion.recall}}{\text{preciscion+recall}} = \frac{\text{TP}}{\text{TP+1/2(FP+FN)}}, \quad \text{where} \tag{1}$$

True positives TP is an outcome where the model *correctly* predicts the positive *Class category*;
False positives FP is an outcome where the model *incorrectly* predicts the positive *Class category*;
False negatives FN is an outcome where the model incorrectly predicts the negative *Class category*.

Random forest/ Confusion Matrix	**Class category 1,2,3,4**					**Class category 1-1;2-1;3-1;4-1**					
1 decision tree/estimator	[[13 0 0 0] [0 16 0 0] [1 0 11 0] [4 0 0 9]]	class	precision	recall	f1-score	accuracy	[[13 0 0 0] [0 16 0 0] [3 0 9 0] [4 0 0 9]]	class	precision	recall	f1-score	accuracy
		1	0.72	1.00	0.84			1	0.65	1.00	0.79	
		2	1.00	1.00	1.00	0.91		2	1.00	1.00	1.00	0.87
		3	1.00	0.92	0.96			3	1.00	0.75	0.86	
		4	1.00	0.69	0.82			4	1.00	0.69	0.82	
10 decision trees/estimators	[[13 0 0 0] [0 16 0 0] [1 0 11 0] [3 0 0 10]]	class	precision	recall	f1-score	accuracy	[[13 0 0 0] [0 16 0 0] [1 0 11 0] [2 0 0 11]]	class	precision	recall	f1-score	accuracy
		1	0.76	1.00	0.87			1	0.81	1.00	0.90	
		2	1.00	1.00	1.00	0.93		2	1.00	1.00	1.00	0.94
		3	1.00	0.92	0.96			3	1.00	0.92	0.96	
		4	1.00	0.77	0.87			4	1.00	0.85	0.92	

Table 3. Confusion Matrix and accuracy estimated parameters for all Class categories

The obtained high classification accuracy for all tested categories, including when the number of trees is only = 1, shows that at the lowest possible complexity of the model, an optimal ratio between the evaluation parameters for the effectiveness of the model is achieved. As 45 % of all 120 combinations are applied to test the trained model, it means that 54 combinations of Response time are randomly tested. The confusion matrix shows that only 5, 4, 7 and 3 combinations are wrong classified, respectively for Class category 1,2,3,4 (1 decision estimator), for Class category 1,2,3,4 (10 decision estimators), Class category 1-1,2-1,3-1,4-1 (1 decision estimator), Class category 1-1,2-1,3-1,4-1 (10 decision estimators). But to determine the most appropriate number of trees in the model, it is necessary to take into account the significance of each of the ten response time values/parameters in the training sample. From the graphs in Figure 3, it is clear that some parameters have almost zero significance with 1 tree in the model, while with 10 trees in

the model, their significance increases. This leads to the conclusion that it is better to finally choose the model with 10 trees. All network topology simulations are held with simulator Packet Tracer [9].

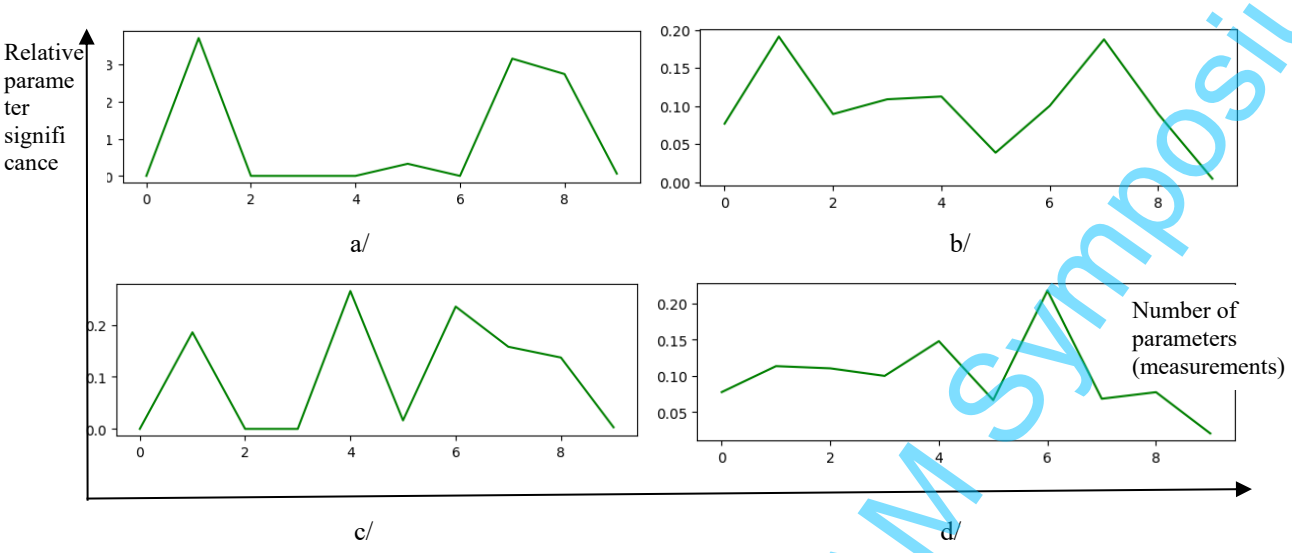


Fig. 3. Parameter significance-a/ for *Class category 1,2,3,4* with 1 tree, /b for *Class category 1,2,3,4* with 10 trees c/ for *Class category 1-1,2-1,3-1,4-1* with 1 tree, d/ for *Class category 1-1,2-1,3-1,4-1* with 10 trees

Thus, depending on the recognized *Class category*, i.e., the solution provided by the model, administrators and managers of the network infrastructure may consider it necessary to convert an existing topology from type Topology1 to type Topology 2 (with more FHRP groups), which will improve performance and lead to lower response time values, or to switch from Topology 3 to Topology 4 (with more VLANs and Per VLAN STP), which will lead to the same positive effect.

4. Comparison and interpretation of results

The two studies, the present one and the one mentioned in [3], both from the same research domain, use different machine learning models and data sets to classify ICMP connection time.

The first paper, "Models for Evaluating Effective IP Traffic Management with Application of Artificial Intelligence" (Random Forest model), focuses on a network restructuring prescription based on pre-trained models., while the second, "Classification of ICMP connection time with a multi-layered neural network" (MLP model) [3], focuses on connection quality assessment and stability.

Feature	Random Forest (RF) Model	Multi-Layered Neural Network (MLP) Model [3]
Machine Learning Model	Random Forest	Multi-Layered Neural Network (MLP) / Deep Learning
Primary Research Goal	Automated prescription for improving the network structure (reconfiguring protocols and restructuring topology).	Automated classification of time variability for connection quality with focus on easy adaptation.
Input Data Features	10 measured ICMP response times (pings) from 10 attempts.	2 parameters: Local Average (LA) time and Remote Average (RA) time [ms].
Output Classes	8 Class Categories corresponding to 4 specific network topologies (FHRP, PVST) under 2 different load conditions (e.g., 5 devices vs. 20 devices).	3 Classes based on fixed connection time thresholds: Class 1 (appropriate quality), Class 0 (bad quality), and Class 2 (no connection).
Data Source	Data generated from simulated network topologies (implementing FHRP and PVST). The training set was expanded using artificial variability (±10% of the average value).	Data collected from IoT collection devices on a 2.4 GHz wireless network. Uses a larger dataset of 30,000 samples.
Classification Results	Achieved high classification accuracy even with low model complexity (1 tree). The chosen 10-tree model had a very low number of wrong classifications (e.g., 4 to 7 out of 54 tested combinations).	Achieved high training and validation accuracy (upwards of 92-93% in the best cases). The MNN structure showed stable work and absence of overfitting with a minimum number of elements.

Table 4. Comparison of results, methodology and goals of the two researches

Table 4 presents a comparison of their results, methodology, and goals. Both models achieved high accuracy, supporting the use of machine learning for ICMP classification. The MLP model was explicitly designed to achieve high accuracy with a minimum number of elements in the dense layers, which is a good prerequisite for fast performance speed when integrated into a real-time system. The Random Forest model also demonstrated high accuracy with low complexity (1 tree) but ultimately chose a 10-tree structure for greater significance of all input features.

The MLP model addresses a fundamental, diagnostic problem by classifying general connection quality based on simple LA/RA thresholds and its strength lies in its ability to show easy adaptation to different threshold requirements (using a New Data Set). While the Random Forest model in this research addresses a more complex, prescriptive problem by evaluating the ICMP response times to the physical and logical configuration of the network (topology and protocol groups). The successful classification suggests the model can serve as a decision-support tool for network restructuring.

5. Positive, negative outcomes and challenges

The core benefit is the ability to receive an automated prescription for improving the network structure. This means the network could dynamically adapt based on its performance, with the model predicting the need for appropriate restructuring of the network topology and reconfiguration of the protocols.

Network managers would move away from reactive or purely manual configuration, instead relying on the high accuracy of the trained model to classify the quality of connectivity. This provides a stronger, data-backed foundation for system reconfiguration.

The model could be implemented and integrated into real-time traffic monitoring systems, allowing for a continuous cycle of data collection, classification, and prescription for newly received data.

The system is inherently designed to be adaptive, allowing for additional training of the system for different network architectures and a corresponding prescription for reconfiguring the system according to the results.

The implementation also presents potential challenges and negative consequences: if the simulated network does not accurately reflect the complexities, traffic patterns, and real-world latency of a live production network, the model's predictions and prescriptions for the real network may be suboptimal.

Also there will occur a risk of faulty automation. If a new data point is misclassified (a false positive or false negative, as indicated by the wrong classifications in the confusion matrix), the system could issue a prescription for unnecessary or counterproductive changes, potentially disrupting network services or introducing new vulnerabilities.

Over-reliance on artificial data generation can be pointed out as the main challenge: to generate a large training set, the we manually add values in the range of +-10% relative to the average of 10 measurements. This technique, while generating a larger set of 30 combinations per class category, introduces artificial variability rather than capturing true, organically observed network fluctuations, which could lead to a selection bias in the training.

6. Some limitations of the research and approach

The verification and testing are confined to a small set of protocols and topologies: FHRP and PVST across four specific simulated topologies. While we suggest the model is applicable to other protocols, its performance on more complex architectures, a larger number of FHRP groups, or more VLANs is proposed only as future work, not a current achievement.

The input data is solely based on ICMP (ping) response times from a single device to a remote destination under two load conditions (5 and 20 end devices for Topology 1 and 2, and 5 and 20 devices split for Topology 3 and 4). This may be an insufficient representation of effective IP traffic management, which can be affected by numerous other dynamic parameters like packets/sec, jitter, packet loss, and application-layer metrics.

7. Conclusion and future work

The proposed method leverages measured time parameters (ICMP response times) from different traffic regimes as input training data for a Random Forest machine learning model. This provides a data-driven basis for management decisions rather than relying solely on manual assessment.

The presented and optimized random forest model for training and classification of remote access time parameters in different network topologies shows high accuracy with +–10% variations in the input data.

The model has been verified for two of the most commonly used protocols for ensuring alternative and fault-free network communication. However, it can also be applied to other network topologies, when working with other protocols.

The model can be implemented and integrated into real-time traffic monitoring systems, with a dynamically created and updated database of remote access times and classification of newly received data.

As a future development of the method, verification and testing of the same protocols, but with a larger number of FHRP groups, as well as with more VLANs in a more complex topology with more switches and different roles of the switches for individual VLANs, can be added.

The model could be implemented and integrated into real-time traffic monitoring systems 8, allowing for a continuous cycle of data collection, classification, and prescription for newly received data. The system is inherently designed to be

adaptive, allowing for additional training of the system for different network architectures and a corresponding prescription for reconfiguring the system according to the results

8. Acknowledgments

Acknowledgements for the support of the Research Project № ЗО-337/10.04.2025, of the *University of Telecommunications and Post* titled "Test models for evaluating effective IP traffic management with the application of artificial intelligence".

9. References

[1] Toplalova, I. & Radoyska, P. (2018). Adjustment of the QoS Parameters on Routers with Neural Network Implementation“, International Journal on Advances in Networks and Services, ISSN:1942-2644 vol. 11, no. 3 & 4, pp. 143 – 151, 2018.
[2] Hillstone Networks, (2021). AI-driven Security Solutions by Hillstone Networks”, 2021, © 2021 Hillstone Networks, https://www.hillstonenet.com/.
[3] Topalova, I., (2023). Classification of ICMP connection time with a multi-layered neural network, 2022 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom) | 978-1-6654-9749 7/22/$31.00 ©2022 IEEE | DOI: 10.1109/BLACKSEACOM54372.2022.9858304.
[4] Topalova, I., Radoyska, P. & Sokolov, S. (2020). Neural Network Implementation for Detection of Denial of Service Attacks Journal of Engineering Science and Technology Reviewthis link is disabled, 2020, (Special Issue), pp. 98–102.
[5] Radoyska, P. & Atanasova, M. (2020). Free tools for testing the security of web services in the utp network, Fifth international scientific conference “Telecommunications, informatics, energy and management”, October 3-4, 2020, Sofia, Bulgaria, pp.133-137
[6] Fedorov, A. & Shkodyrev, V. (2018). Recommendation System based on Neural Network for Prediction of Emergency Situations, Proceedings of the 29th DAAAM International Symposium, pp.0133-0136, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-20-4, ISSN 1726-9679, Vienna, Austria DOI: 10.2507/29th.daaam.proceedings.018
[7] Gaku, R. (2024). Scheduling Optimization of Manufacturing Process Flows using Agent-Based Simulation Modeling and Neural Networks, Chapter 14 in DAAAM International Scientific Book 2024, pp.171-180, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-42-6, ISSN 1726-9687, Vienna, Austria DOI: 10.2507/daaam.scibook.2024.14
[8] Radovanovic, I, (2024). Google Colab – A Step-by-step Guide
[9] Cisco.com (2025), Cisco netacad.com, Simulator Packet tracer 8.3.
[10] Capa Learning, (2025). F1 Score in Machine Learning: Formula, Precision and Recall/ Artificial Intelligence