WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Detection of Anomalous TCP/IP Network Requests

Km Anjali Srivastava[1], Aman Agrawal[2], Ms. Isha Gupta[3]

[1 2] Raj Kumar Goel Institute of Technology Ghaziabad

[3] Assistant Professor, Raj Kumar Goel Institute of Technology Ghaziabad

ABSTRACT

This paper has proposed an AI approach for the malware classification system and organization-based inconsistency identification utilizing Autoencoder (AE). Contrasted with the past work, our proposed model has used a remarkable preparing stage and topography. We have shown how a unique model with a comparable trained model and the topography can be more productive for malware classification and anomaly detection in network requests. This will be useful with regards to developing a standard inserted security system device for different frameworks.

For better framework proficiency, 9 more algorithmic rules were utilized for anomaly detection in network request. As a result, it was deduced that our proposed model has better efficiency contrasted with these 9 algorithmic rules.

For detecting network-based anomalous, 3 hidden layers neural network have been utilized as the Autoencoder. First, the conveyance of calculated error is plotted for the training dataset and afterwards used to distinguish an appropriate threshold value for recognizing an anomaly. The set threshold is kept over the "noise" level.

Introduction

The number of devices connected to the Internet has increased from approximately 7 billion in 2018 to around 26.66 billion in the year 2019 and an estimation of 31 billion devices is made for the year 2020. With the growth of the Internet of Things (IoT), artificial intelligence (AI), cryptocurrencies and other financial technologies along with the increased dependency of businesses, government and common people on the Internet and terabytes of confidential data being stored and circulated around the network, it becomes very important to ensure the security of the network and provide the required immunity to the users of the Internet. With the increasing computational power and availability of computers, the threat of cyber-attacks is alarming. Amongst the various strategies to ensure network security and reliability, is the ability to detect the arrival of anomalous network requests in order to take action and defend against them. Some of the common network attacks that are encountered include Distributed Denial of Service (DDoS), Remote to Local (R2L), Probe and U2R.

The project involves building a complete pipeline that receives, processes, stores and displays real-time network data. The real-time data is received by a Python process that analyses the input to check if it is an anomalous network request or a regular non-harmful network request. The process stores this data in a data store and through a frontend interface provides the user with the statistics of the network requests. The machine learning model is trained on a labelled dataset and is tested for different algorithms on both the supervised and unsupervised learning paradigms. Based on the results obtained for each of the learning algorithms, the best model shall be selected.

The proposed system addresses the objective of detecting anomalous network requests and classifying it to the corresponding anomaly type. The use of autoencoders along with the ANN classifier is able to effectively determine the type of the network request with a significant amount of accuracy. This ensures reliability on the outcomes provided by the system and to allow necessary actions to be taken to ensure network safety. The instances of feature bunch in domain which involves detection of anomaly in network, incorporate organization stream, IP address of source and respective port, objective IP as well as conventions.

The report has been prepared and organized in such a way that it is easy to understand the entire flow of the project. This synopsis will give an overview of the project, introducing to readers the problem statement, and a brief description of the same. The following sections give deeper insight into the Anomalous detection application including design of the project, Objective of the project, Methodology of the project, its expected outcome and its Software and Hardware requirements. Each component of the project and how it is implemented has been clearly explained.

Literature Survey

Anomaly Detection in TCP/IP network connection has been researched for quite a time, due to the increasing number of devices that are always connected to the Internet, and most of the traffic is on TCP/IP protocols. It becomes necessary to effectively monitor the network logs to detect any anomaly in the network. A new fast and efficient approach using Neural networks is needed for analysing the increasing amount of data, and detecting anomalies in it. Some of the papers discussing such methods are included in this section.

[1] goes in detail about the various machine learning models which can be used for anomaly detection using machine learning. It discusses in detail the use of Gaussian multivariate, K means, and LSTM models for anomaly detection. However, it does not talk about the best model which can be used for the majority of the cases.

[2] has listed Autoencoders as a suitable method for anomaly detection problems. It gives an architecture of the autoencoder model which can be used. We take the Autoencoder architecture from this paper. The architecture seems a bit complex here, and no discussion on reconstruction error has been made in this paper.

[3] is an exploratory paper about the NSL KDD dataset and is very useful for understanding the dataset and getting the most important parameters of it to be used in our model and thereby reducing the overall complexity of the model. The paper was used as a reference to extract features and confirm them with the results we achieve after Principal Component Analysis.

[4] leverages a streaming architecture based on ELK, Spark, and Hadoop in order to collect, store, and analyse network connection logs in real-time. The paper highlights the most important features and parameters to be displayed in a graphical manner which gives the most useful analysis of the real-time network connection logs.

OBJECTIVES OF THE PROPOSED WORK

The proposed system addresses the objective of detecting anomalous network requests and classifying it to the corresponding anomaly type. The use of autoencoders along with the ANN classifier is able to effectively determine the type of the network request with a significant amount of accuracy. This ensures reliability on the outcomes provided by the system and to allow necessary actions to be taken to ensure network safety. The instances of feature bunch in domain which involves detection of anomaly in network, incorporate organization stream, IP address of source and respective port, objective IP as well as conventions. For cyber-attack investigation, entropy of the binary file and may be quantity of bytes, along with system calls other than activity code for gathering records is being ordinarily utilized. The AE will take in ideal area including first capabilities for accomplishing both of those errands. Furthermore, benefit in our project plot will be dimensionality decrease. As far as the manageability of a model, a few classifiers require the perception of uncorrelated highlights. The most ordinarily utilized factual procedures to give such highlights is PCA said as Principal Component Analysis. The Autoencoders can lessen dimensionality for the highlights, in this manner assisting with decreasing the memory expected to register the covariance network.

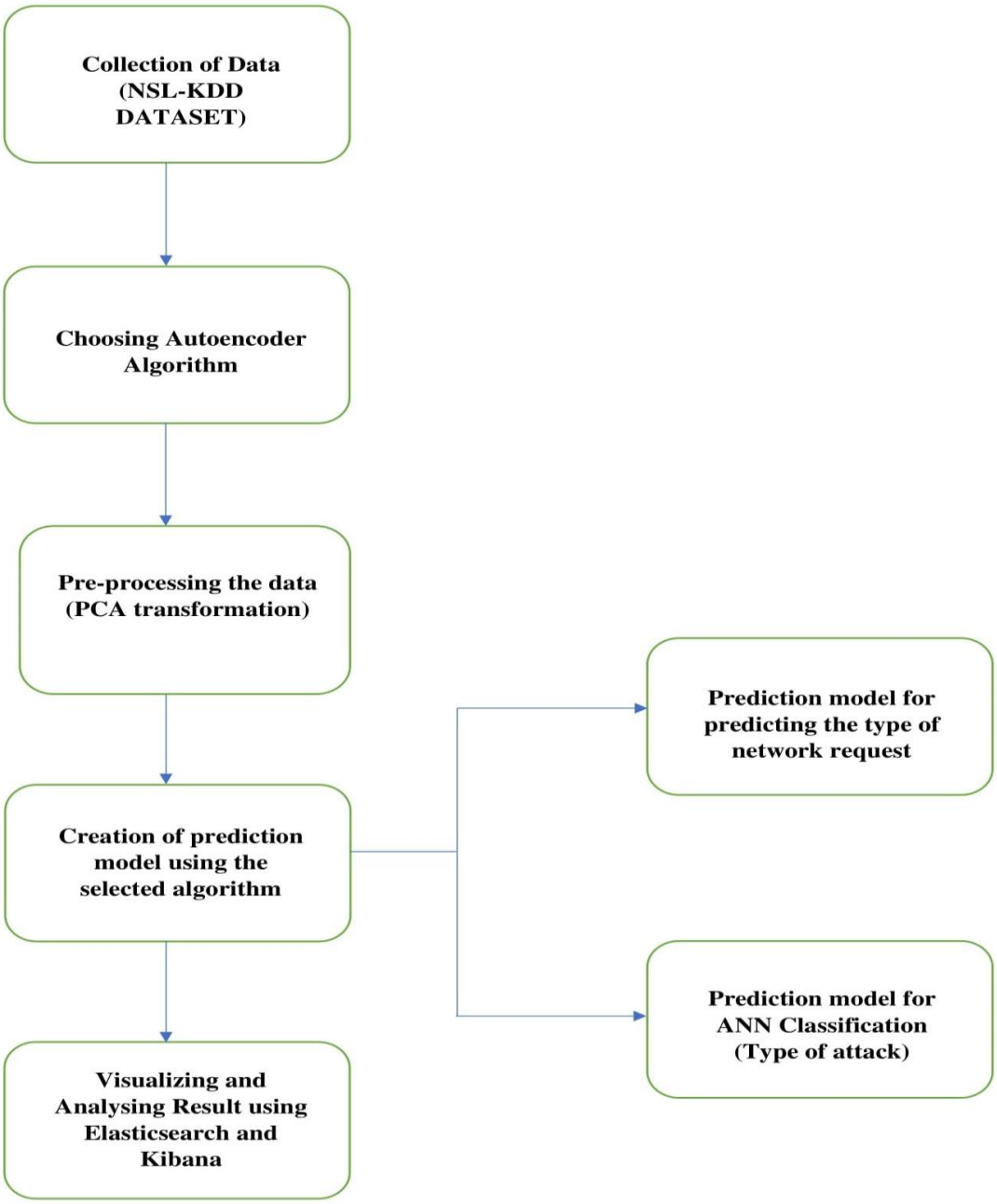

Fig: Basic overview of the idea

The main objective of the project is to detect anomalous requests and classify them into the type of attack using the most accurate and efficient machine learning algorithm of the best performance. So comparative study is performed to decide the best training method for the model. The machine learning model is trained on a labelled dataset and is tested for 10 different algorithms on both the supervised and unsupervised learning paradigms.
Based on the results obtained for each of the learning algorithms, the best model shall be selected to fulfil the objective of the project. The process stores this data in a data store and through a frontend interface provides the user with the statistics of the network requests.

Methodology

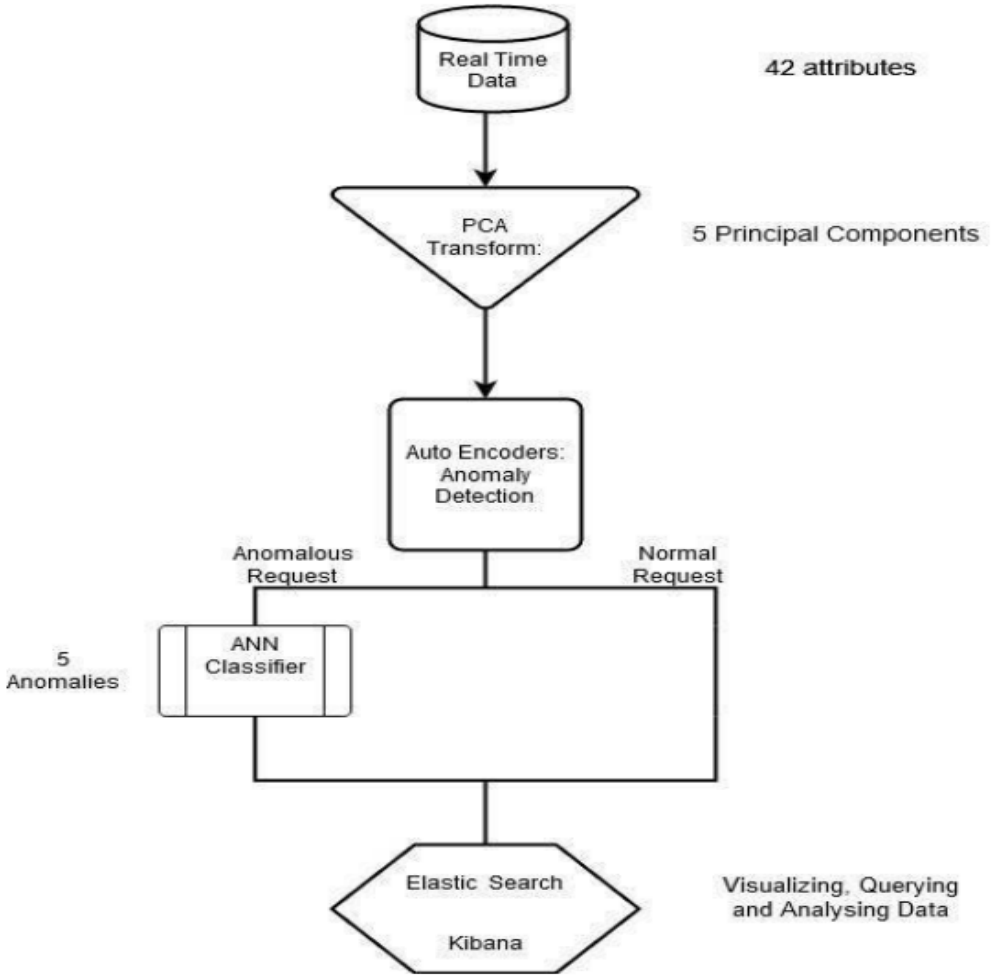

Fig: High Level System Design

Step 1: Pre-process the dataset.
Step 2: Train, develop and select an appropriate model.
Step 3: Configure the storage and data pipeline.
Step 4: Process the real-time data and store the same with the results.
Step 5: Integrate the frontend visualization interface.
Step 6: Build visualizations to obtain statistics of the data.

TECHNICAL PROPOSTION

Pre-processing of data set

KDD dataset requires some processing before feature selection. For feature selection all attribute will be manipulated but some attribute has their values in text not in numbers so first of all we have to replace those values with mathematical numbers and for that we are going to use one hot encoding. After that we have to normalise the attribute values as well.

Feature Selection

Principal component analysis is a technique for feature extraction — so it combines our input variables in a specific way, then we can drop the "least important" variables while still retaining the most valuable parts of all of the variables! As an added benefit, each of the "new" variables after PCA are

all independent of one another. This is a benefit because the assumptions of a linear model require our independent variables to be independent of one another. If we decide to fit a linear regression model with these "new" variables this assumption will necessarily be satisfied.

Implementation of classifier

Autoencoders

Autoencoders are a family of neural networks that essentially try to recreate the input. In other words, autoencoders accept the input with all of its core features and attempt to learn the pattern of data by learning an identity function $h(x) \approx x$. There is some loss (reconstruction error) that is involved in this process. But the intuition is that the encoder will be to accurately recreate the input corresponding to the normal network requests, and will incur a large loss in reconstructing the anomalous network requests. To learn this identity function, an autoencoder uses two main components – an encoder and a decoder. As the names suggest, the encoder accepts the input and compresses it in a spatial latent visualization, whereas decoder part aims reconstruction of input from this spatial latent visualisation.

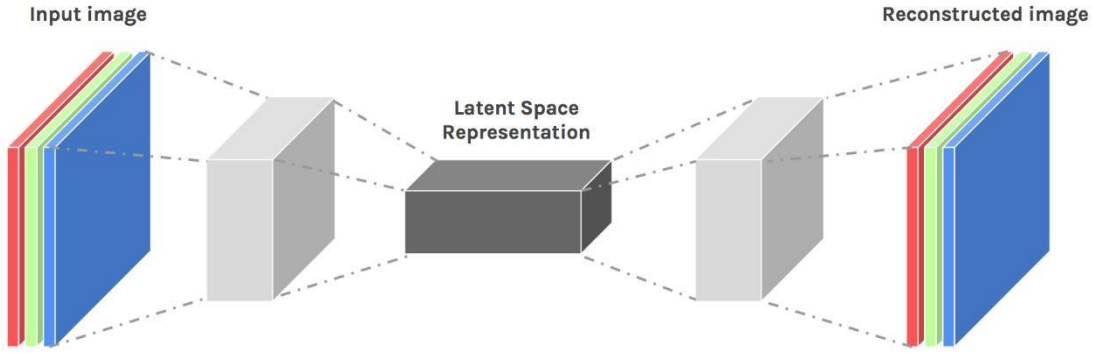

Fig : Analysis of processes involved in Autoencoder

An ideal autoencoder should be able to reconstruct the exact same input which was initially passed to it as input, but in the real world there is a finite "reconstruction error" corresponding to each input tuple. This reconstruction error can be utilized as a statistical metric to draw inferences. Comparing the observed "reconstruction error" with a threshold error (calculating during training of the model) can distinguish between the two types of requests.

ANN classifier

Artificial neural networks (ANNs), usually simply called neural networks (NNs) or, more simply yet, neural nets, are computing systems inspired by the biological neural networks that constitute animal brains.

An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal to other neurons. An artificial neuron receives signals then processes them and can signal neurons connected to it. The "signal" at a connection is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs. The connections are called edges. Neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Neurons may have a threshold such that a signal is sent only if the aggregate signal crosses that threshold. Typically, neurons are aggregated into layers. Different layers may perform different transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times.

Datastore and Visualisation

Elasticsearch

Raw data flows into Elasticsearch from a variety of sources, including logs, system metrics, and web applications. Data ingestion is the process by which this raw data is parsed, normalized, and enriched before it is indexed in Elasticsearch. Once indexed in Elasticsearch, users can run complex queries against their data and use aggregations to retrieve complex summaries of their data. From Kibana, users can create powerful visualizations of their data, share dashboards, and manage the Elastic Stack.

Searching and visualizing data in kibana

Kibana enables the visual analysis of data from an Elasticsearch index or multiple indices. Indices are created when Logstash (a largescale ingested) or Beats (a collection of single purpose data shippers) ingests unstructured data from log files and other sources and converts it into a structured format for Elasticsearch storage and search functionalities.

Kibana's interface allows users to query data in Elasticsearch indices and then visualize the results through standard chart options or built-in apps like Lens, Canvas, and Maps. Users can choose between different chart types, change the aggregations of numbers, and filter to specific segments of data.

RESULT ANALYSIS

Our project proposes to construct a complete pipeline which receives, processes, stores as well as displays real-time network data. The real-time data is received by a Python process that analyses the input to check if it is an anomalous network request or a regular non-harmful network request. The process stores this data in a data store and through a frontend interface provides the user with the statistics of the network requests. Our project also proposes to consider both malware classification and detection of any malware.

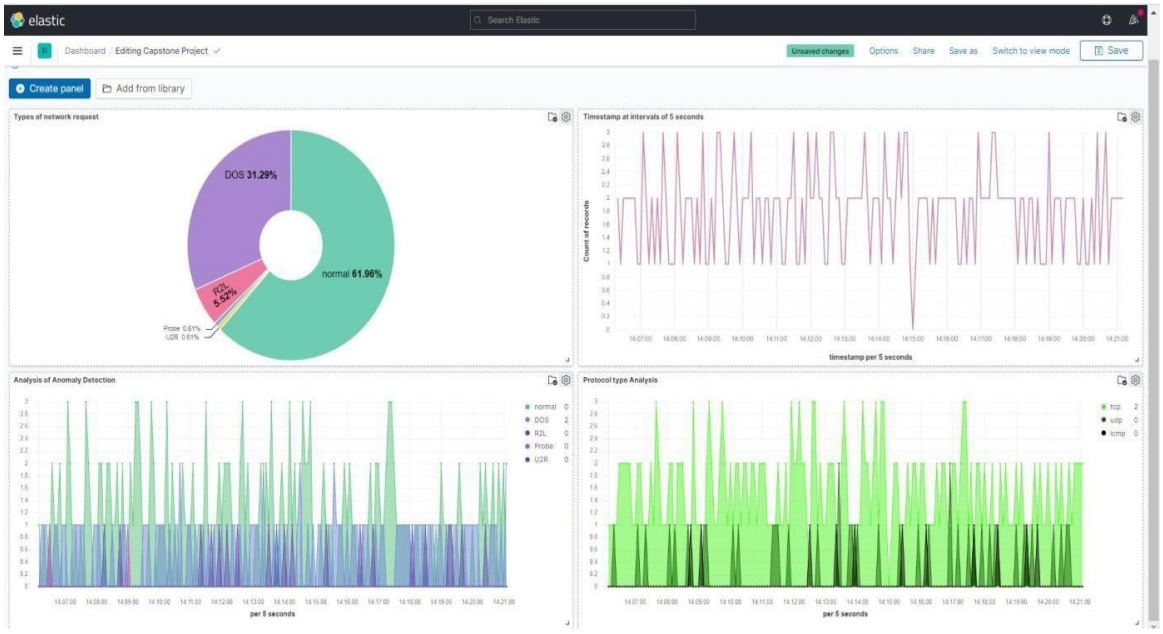

Fig: Graphical representation of the result on the real time data

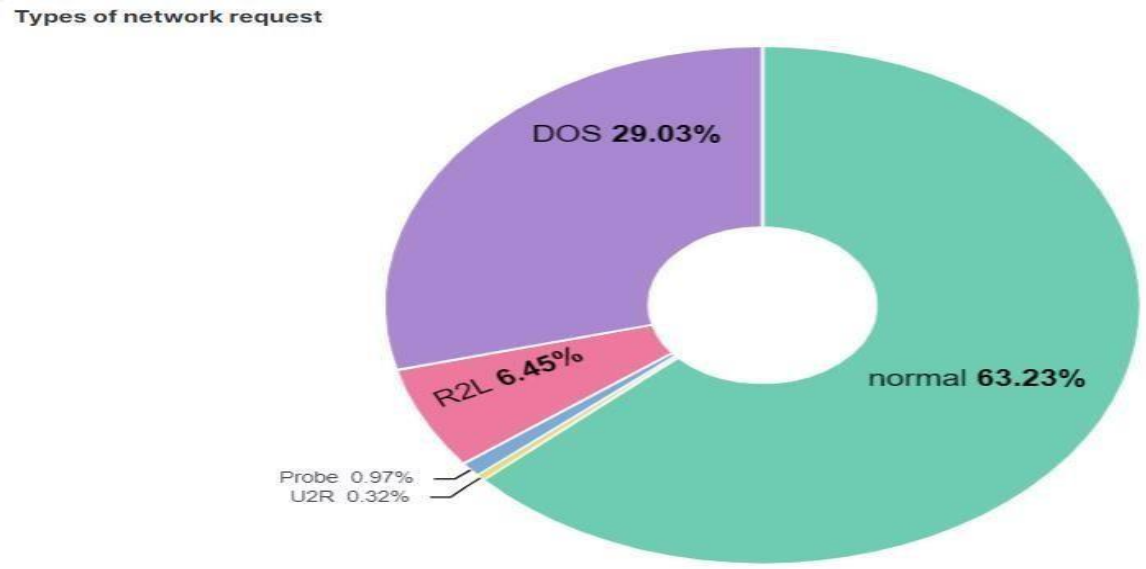

Fig: Donut visualization of the network requests

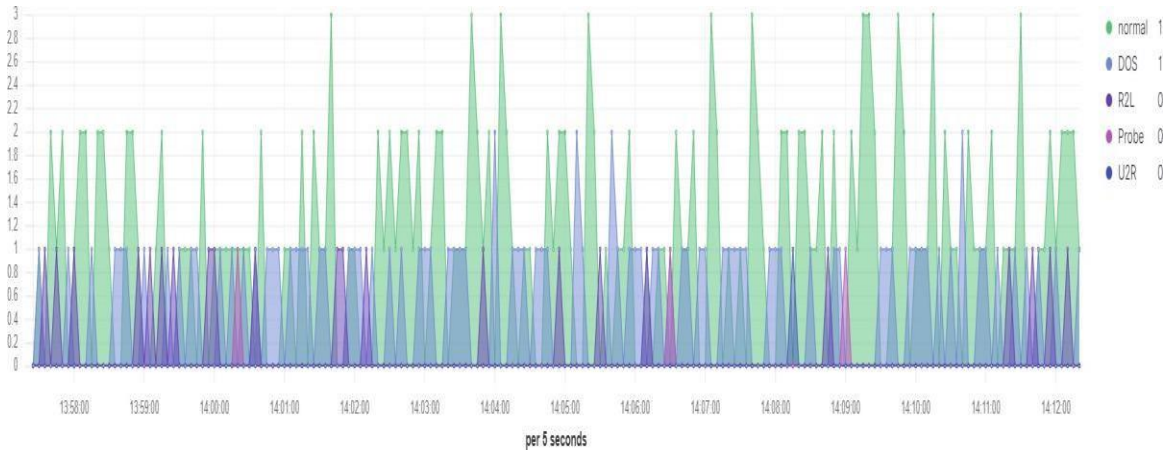

Fig: Time graph representation of network request

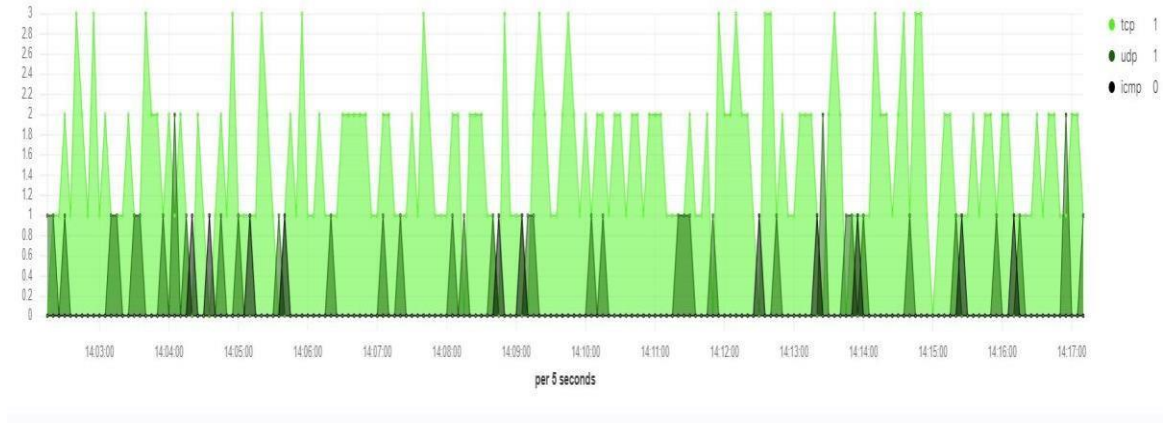

Fig: Time graph representation of protocol type received

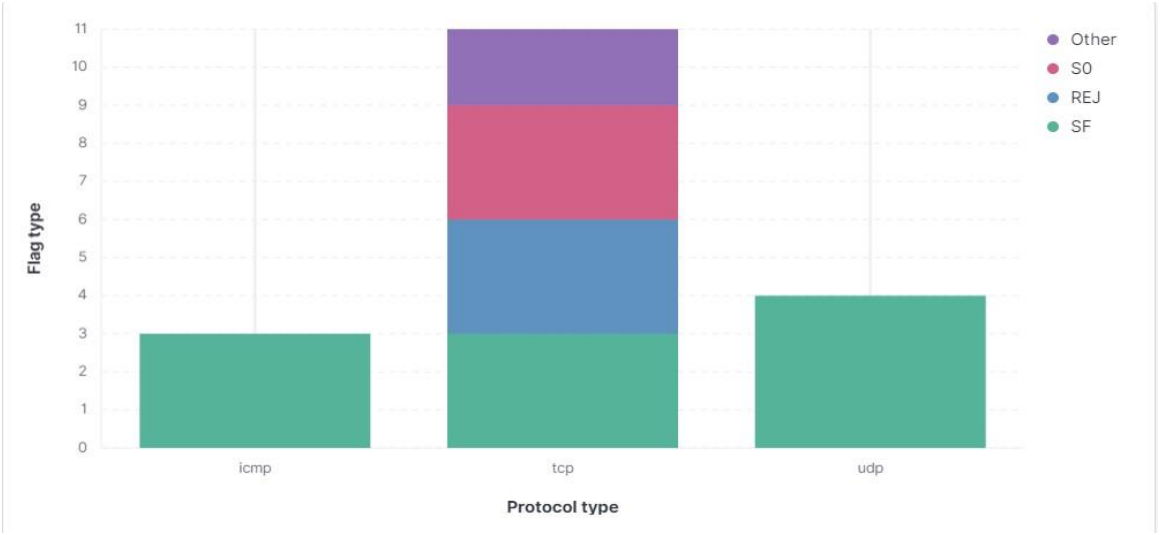

Fig: Bar graph representation of flag vs different protocol

CONCLUSION

The proposed system addresses the objective of detecting anomalous network request and classifying it to the corresponding anomaly type. The use of autoencoders along with the ANN classifier is able to effectively determine the type of the network request with a combined accuracy of 84 %. This ensures reliability on the outcomes provided by the system and to allow necessary actions to be taken to ensure network safety. The architecture discussed could be used on other network request datasets with reduced or varied features to obtain similar results. The development of a real-time network data pipeline with Elasticsearch and Kibana allows the end user to effectively create inferences from the data and the statistics obtained from it. The system may be integrated with existing network safety management software for improved performance in network anomaly detection.

FUTURE SCOPE

This system gives an analysis of the anomalies which can occur and the accuracy. Further improvements which can be made in this project are-

- Accuracy of the model can be improved.
- We can increase the dataset to incorporate more anomalies in the project.
- Better graphical representation of the data and anomalies can be made for real time data.

REFERENCES

1. Yousefi-Azar, Mahmood, Vijay Varadharajan, Len Hamey, and Uday Tupakula. "Autoencoder-based feature learning for cyber security applications." In 2017 International joint conference on neural networks (IJCNN), pp. 3854-3861. IEEE, 2017.
2. Chakraborty, Niloy & Nair, Roshan & Kasula, Chaithanya Pramodh & Vankayala, Sravanthi. (2019). IP Network Anomaly Detection using Machine Learning. 10.1109/I2CT45611.2019.9033545.
3. M. Sakurada and T. Yairi, “Anomaly Detection Using Autoencoders with Nonlinear
4. Dimensionality Reduction,” presented at the MLSDA 2014 2nd Workshop, 2014, doi: 10.1145/2689746.2689747.
5. Chae, Hee-su, Byung-oh Jo, Sang-Hyun Choi and Twae-Kyung Park. “Feature Selection for Intrusion Detection using NSL-KDD.” (2013).
6. Mehta, Swapneel, et al. ‘Anomaly Detection for Network Connection Logs’. ArXiv:1812.01941 [Cs, Stat], Nov. 2018. arXiv.org, http://arxiv.org/abs/1812.01941.
7. Yehezkel, Aviv, Eyal Elyashiv, and Or Soffer. "Network anomaly detection using transfer learning based on auto-encoders loss normalization." Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security. 2021.
8. Mishin, Mikhail. "Anomaly Detection Algorithms and Techniques for Network Intrusion Detection Systems." (2020).
9. Thakkar, Ankit, and Ritika Lohiya. "Attack classification using feature selection techniques: a comparative study." Journal of Ambient Intelligence and Humanized Computing 12.1 (2021): 1249-1266.
10. M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, “A Detailed Analysis of the KDD CUP 99 Data Set,” Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), 2009.
11. https://en.wikipedia.org/wiki/Autoencoder
12. https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f
13. https://www.sartorius.com/en/knowledge/science-snippets/what-is-principal-component-analysis-pca-and-how-it-is-used
14. https://www.elastic.co/
15. https://www.elastic.co/kibana/