ÇUKUROVA UNIVERSITY
INSTITUTE OF NATURAL AND APPLIED SCIENCES

MSc THESIS

Roaa Rafih MOHAMMED

ANOMALY DETECTION IN NETWORK TRAFFIC USING MACHINE LEARNING

DEPARTMENT OF COMPUTER ENGINEERING

ADANA-2022

ABSTRACT

MSc THESIS

ANOMALY DETECTION IN NETWORK TRAFFIC USING MACHINE LEARNING

Roaa Rafih MOHAMMED

CUKUROVA UNIVERSITY
INSTITUTE OF NATURAL AND APPLIED SCIENCES
DEPARTMENT OF COMPUTER ENGINEERING

Supervisor : Prof. Dr. M. Fatih AKAY
Year: 2022, Pages 73
Jury : Prof. Dr. M. Fatih AKAY
: Asst. Prof. Mehmet Uğraş CUMA
: Asst. Prof. Dr. Yasin KAYA

A primary thematic of this study is centered on detecting anomalies and measuring the device health for Central Processing Unit (CPU), memory utilization, and allocation; for Key Performance Indicator (KPI) dataset which assembled throw twenty-one-day, by improving models using machine learning (ML) methods; namely, Convolutional Neural Network (CNN), and Long Short-Term Memory (LSTM), with Auto Encoders (AE), One-Class Support Vector Machine (Oc-SVM), also k-Nearest Neighbors (k-NN). The accuracy of all methods was measured by using a confusion matrix. According to the observed results, the deep learning methods yield great performance results compared to classification methods for all models. In general, CNN/AE and LSTM/AE models show higher accuracy than the other methods. The ranking of models from best to worst based on accuracy in the confusion matrix are; CNN/AE, LSTM/AE, as for the deep learning models, while for classification models the favorable order for the methods are; k-NN, and Oc-SVM.

Keywords: Deep Learning, Anomaly Detection.

ÖZ

YÜKSEK LİSANS TEZİ

MAKİNE ÖĞRENMEYİ KULLANARAK AĞ TRAFİĞİNDE ANOMALİ TESPİTİ

Roaa Rafih MOHAMMED

ÇUKUROVA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

Danışman : Prof. Dr. M. Fatih AKAY
Yıl: 2022, Sayfa 73
Jüri : Prof. Dr. M. Fatih AKAY
: Dr.Öğr.Üyesi Yasin KAYA
: Assoc. Prof. Mehmet Uğraş CUMA

Bu çalışmanın ana temalarından biri, Merkezi İşlem Birimi (CPU), bellek kullanımı ve bellek tahsis için anormallikleri tespit etmeye ve cihaz sağlığını ölçmeye odaklanmıştır; Makine öğrenmesi (ML) yöntemlerini kullanarak modelleri geliştirerek yirmi bir günlük süreyi bir araya getiren Anahtar Performans Göstergesi (KPI) veri seti için. Evrişimli Sinir Ağı (CNN) ve Uzun Kısa Vadeli Bellek (LSTM) ile Otomatik Kodlayıcılar (AE), Tek Sınıf Destek Vektör Makinesi (Oc-SVM), ayrıca k-En Yakın Komşular (k-NN). Tüm yöntemlerin doğruluğu bir karışıklık matrisi kullanılarak ölçülmüştür. Gözlemlenen sonuçlara göre, derin öğrenme yöntemleri, tüm modeller için sınıflandırma yöntemlerine kıyasla daha iyi performans sonuçları vermektedir. Genel olarak CNN/AE ve LSTM/AE modelleri diğer yöntemlere göre daha yüksek doğruluk göstermektedir. Karışıklık matrisinde doğruluğa göre modellerin en iyiden en kötüye doğru sıralanması; Derin öğrenme modellerinde CNN/AE, LSTM/AE, sınıflandırma modellerinde ise yöntemler için uygun sıralama; k-NN ve Oc-SVM.

Anahtar Kelimeler: Derin Öğrenme, Anomali Tespiti.

EXTENDED ABSTRACT

Anomaly detection is mostly grasped as the identification of points, or wrongs that distort the data. The distortions are translated into three kinds of problems which are; frauds, defects, or errors.

In particular, the contexts in network traffic, the anomaly in traffic is like an explosion that happens without warning. So, this pattern needs a full readiness in order to detect these states and to protect the network early.

From this standpoint, numerous methods have been developed over decades, such as; (RNN), (CNN), (SVM), (OPTICS), (DBSCAN), (Ward hierarchical clustering), and (Mean-shift), and it has become possible to classify these methods into two denominations which are; supervised, and unsupervised, or a mixture of them called semi-supervised all of which is called, in short, anomaly detection techniques.

To put it simply, labeling and classifying datasets as normal and abnormal are a requirement of supervised techniques. In addition, learning by iterative training is used to make predictions and detect anomalies. While unsupervised algorithms are used with unlabeled test datasets, by assuming a part of the datasets is normal, unnormal data are instances that do not seem to fit with the remainder of the datasets. Semi-supervised methods construct a model appearing a normal attitude from a normal training dataset, and then test the likelihood of a test pattern generated by the utilized model.

To study behaviors of the information in networks and to provide the necessary protection by detection of novelty is considered as one of the biggest challenges today. Thus, this thesis aims to detect outlier data in network traffic, where some of the machines learning techniques are applied for this aim. Models which have been utilized are CNN/AE, LSTM/AE, Oc-SVM, and k-NN.

The methods are; the model of Convolutional Neural Network with Auto Encoder (CNN/AE). CNN; is a kind of Artificial Neural Networks (ANN) in deep learning, commonly used for analysis, classification, recognition, segmentation. Explained briefly, convolutional networks which are inspired by the connectivity pattern between neurons, are considered as versions of multilayer perception. As a major advantage, CNN usually uses few processers that are done with human intervention to learn through Machine training, whereas in other classification methods the filters are hand-engineered.

The second model is a Long Short-Term Memory (LSTM / AE). LSTM is considered as an improved architecture of artificial recurrent neural network (RNN), which has feedback connections. Likewise, it is utilized in many fields such as recognition and detection novelty in network traffic. LSTM has high ability and performance to process different data types like images, video, and speech.

So as to detect an anomaly, we need to use Convolutional Neural Networks, and Long Short-Term Memory with Auto Encoder, which is also known as a type of artificial neural network, which is utilized to label and signal effectively for unlabeled data (unsupervised learning).

One-Class Support Vector Machine (Oc-SVM), in simple words, we can define the Oc-SVM as an unsupervised method, which has the ability to learn by training to detect an anomaly in network traffic by categorizing new data to identical or dissimilar in the training set. For this reason, Oc-SVM is one of the most popular prediction methods.

k-Nearest Neighbors (k-NN), is a non-parametric supervised learning method, which depends on distance for classification the input data, and the output depends on whether k-NN is used for classification or regression to find groups in a class membership. k-NN; is a popular methods used for the assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones.

Detection strategies which are used in this study, are classification strategies, thus the dataset was split by hold-out, and k-fold techniques into (70 %) training data and (30 %) testing data with hold-out technique, and (10 n_sample) with k-fold techinque.

In the all experiments for CPU, Memory, also for Disk; 70% of dataset content has been utilized for training while the rest of the dataset has been utilized for testing. Additionally, 10 of n_samples for k-fold are used to training and testing divisions have been utilized as second partitioning range for datasets. For the detection paradigm, we have used grid-search to tuning different hyperparameters for the dataset in order to develop classification and deep learning models.

The performance of all models was evaluated depending on Accuracy calculated by using the confusion matrix. 24 different models have been developed in total, including 12 models for deep learning strategies and 12 models for classification strategies.

Observed results lead to the deep learning models produced better results compared to the classification models of all methods. In general, CNN/AE and LSTM/AE models show better performance than other methods.

The order of used methods for the prediction model in terms of accuracy-based performance, from best to worst, is CNN/AE, and LSTM/AE, while for the classification model, the order of favor of these methods is shown as k-NN and Oc-SVM.

GENİŞLETİLMİŞ ÖZET

Anormallik tespiti, çoğunlukla verileri bozan noktaların veya yanlışların tespiti olarak kavranır. Çeşit çeşit problemlere çevrilen çarpıtmalar; sahtekarlıklar, kusurlar veya hatalar.

Özellikle ağ trafiğindeki bağlamlar, trafikteki anormallik, uyarı vermeden gerçekleşen bir patlama gibidir. Bu nedenle, bu modelin bu durumları algılamak ve ağı erkenden korumak için tam hazır olması gerekir.

Bu açıdan bakıldığında, onlarca yıl içinde; (RNN), (CNN), (SVM), (OPTICS), (DBSCAN), (Ward hiyerarşik kümeleme) ve (Mean-shift) ve bu yöntemleri iki kupürde sınıflandırmak mümkün hale geldi; denetimli ve denetimsiz veya bunların karışımına yarı denetimli adı verilen bu kategorilerin tümüne kısaca anomali tespit teknikleri denir.

Basitçe söylemek gerekirse, veri kümelerini normal ve normal olmayan olarak etiketlemek ve sınıflandırmak, denetimli tekniklerin bir gereğidir. Ek olarak, yinelemeli eğitim yoluyla öğrenme, tahminlerde bulunmak ve anormallikleri tespit etmek için kullanılır. Denetimsiz algoritmalar, etiketlenmemiş test veri kümeleriyle kullanılırken, veri kümelerinin bir bölümünün normal olduğu varsayılarak ve normal olmayan veriler, veri kümelerinin geri kalanına uymayan örneklerdir. Yarı denetimli yöntemler, normal bir eğitim veri kümesinden normal bir tutum gibi görünen bir model oluşturur ve ardından kullanılan model tarafından oluşturulan bir test modelinin olasılığını test eder.

Ağlardaki bilginin davranışlarını incelemek ve yeniliği tespit ederek gerekli korumayı sağlamak günümüzün en büyük zorluklarından biri olarak görülmektedir. Bu nedenle, bu tezdeki hedef, bu amaçla bazı makine öğrenme tekniklerinin uygulandığı ağ trafiğindeki aykırı değerlerin tespit edilmesidir; kullanılan modeller CNN/AE, LSTM/AE, Oc-SVM, ve k-NN'dir.

Yöntemler; Otomatik Kodlayıcılı (CNN/AE) Evrişimli Sinir Ağı modeli. CNN; Derin öğrenmede yaygın olarak analiz, sınıflandırma, tanıma, segmentasyon

için kullanılan bir tür Yapay Sinir Ağlarıdır (ANN). Kısaca, evrişimli ağlar, çok katmanlı algının versiyonları olarak kabul edilen nöronlar arasındaki bağlantı modelinden ilham alır. Büyük bir avantaj olarak, CNN genellikle Makine eğitimi yoluyla öğrenmek için insan müdahalesi ile yapılan birkaç işlemci kullanır, oysa diğer sınıflandırma yöntemlerinde filtreler elle tasarlanmıştır.

İkinci model ise Uzun Kısa Süreli Bellektir (LSTM/AE). LSTM, yapay tekrarlayan sinir ağının (RNN) geliştirilmiş bir mimarisi olarak kabul edilir. Geri bildirim bağlantılarınız olsun. Aynı şekilde ağ trafiğinde tanıma ve algılama yeniliği gibi birçok alanda kullanılmaktadır. LSTM'nin görüntüler, video ve konuşma gibi farklı veri türlerini işlemek için yüksek yetenek ve performansa sahip olduğu yerler.

Bir anomaliyi tespit etmek için bir model olarak, Evrişimli Sinir Ağlarını ve Otomatik Kodlayıcı ile Uzun Kısa Süreli Belleği kullanmamız gerekiyor. Bu, bir tür yapay sinir ağı olarak da bilinebilir. Etiketlenmemiş veriler için etkili bir şekilde etiketlemek ve sinyal vermek için kullanıldığında (denetimsiz öğrenme).

Tek Sınıf Destek Vektör Makinesi (Oc-SVM), basit bir deyişle Oc-SVM'yi denetimsiz bir yöntem olarak tanımlayabileceğimiz, yeni verileri aynı veya farklı olarak kategorize ederek ağ trafiğindeki bir anormalliği tespit etmeyi eğitim yoluyla öğrenme yeteneğine sahiptir. eğitim seti. Bu nedenle Oc-SVM en popüler tahmin yöntemlerinden biridir.

k-En Yakın Komşular (k-NN), giriş verilerinin sınıflandırılması için mesafeye bağlı olan ve çıktı, k-NN'nin sınıflandırma için mi yoksa bir sınıftaki grupları bulmak için regresyon için mi kullanıldığına bağlı olan, parametrik olmayan denetimli bir öğrenme yöntemidir. üyelik. k-NN; komşuların katkılarına ağırlık atamak için kullanılan popüler bir yöntemdir, böylece yakın komşular uzak olanlardan daha fazla ortalamaya katkıda bulunur.

Bu çalışmada kullanılan tespit stratejileri sınıflandırma stratejileridir, dolayısıyla veri seti hold-out ve k-fold teknikleri ile bölünmüştür. Hold-out tekniği ile (%70) eğitim verisine ve (%30) test verisine, ve (10 n_sample) k-katlama tekniği ile.

CPU, Memory, Disk için yapılan tüm deneylerde; Veri seti içeriğinin %70'i eğitim için, geri kalanı ise test için kullanılmıştır. Ek olarak, k-fold için n_samples'ın 10'u eğitim için kullanılmış ve bölümlerin test edilmesi, veri kümeleri için ikinci bölümleme aralığı olarak kullanılmıştır. Algılama paradigması için, sınıflandırma ve derin öğrenme modelleri geliştirmek amacıyla veri kümesi için farklı hiperparametreleri ayarlamak için ızgara aramayı kullandık.

Tüm modellerin performansı, karışıklık matrisi kullanılarak hesaplanan Doğruluğa bağlı olarak değerlendirilmiştir. Derin öğrenme stratejileri için 12 model ve Sınıflandırma stratejileri için 12 model olmak üzere toplam 24 farklı model geliştirilmiştir.

Gözlemlenen sonuçlar; Derin öğrenme modelinin tüm yöntemlerin sınıflandırma modellerine göre daha iyi sonuçlar vermesine yol açar. Genel olarak CNN/AE ve LSTM/AE modelleri diğer yöntemlere göre daha iyi performans göstermektedir.

Tahmin modeli için kullanılan yöntemlerin doğruluk temelli performans açısından en iyiden en kötüye doğru sıralaması CNN/AE, ve LSTM/AE şeklinde iken, sınıflandırma modeli için bu yöntemlerin tercih sırası şu şekildedir: k-NN ve Oc-SVM olarak gösterilir.

ACKNOWLEDGMENTS

I would like to thank my supervisor, Prof. Dr. Mehmet Fatih AKAY, for his guidance, support, encouragement, and generosity throughout all phases of this thesis and my graduate studies. His presence and optimism have invaluable influenced my career and outlook for the future.

I would also like to thank, Dr. Yasin KAYA, and Dr. Mehmaet Uğraş CUMA, for their valuable time and constructive comments with regard to this thesis. Also, like to thank all my professors at Çukurova University-Adana and all my friends.

I dedicate this thesis to my father and my mother. Finally, I am most grateful to my family, especially my parents, and my husband, for their continued love and support; and for all, they have done for me throughout my entire life. Words cannot express my appreciation for them.

A special thanks to my son Oğuzhan ….

CONTENTS **PAGE**

ABSTRACT.....I
ÖZ II
EXTENDED ABSTRACT III
GENİŞLETİLMİŞ ÖZET VII
ACKNOWLEDGMENTS XI
CONTENTS..... XII
LIST OF TABLES..... XIV
LIST OF FIGURES XVI
LIST OF ABBREVIATIONS..... XVIII
1. INTRODUCTION 1
1.1. Overview of Anomaly Detection in Network Traffic 1
1.2. Review of the Literature Studies..... 4
1.3. Overview of The Datasets..... 10
1.4. Motivation, Purpose and Contributions of This Thesis 11
2. METHODS SUMMARY..... 15
2.1. One Class Support Vector Machine (Oc-SVM) 15
2.2. k-Nearest Neighbors (k-NN) 17
2.3. Auto Encoders(AE) 19
2.4. Convolutional Neural Network /Auto Encoder (CNN/ AE)..... 22
2.5. Long Short-Term Memory/ Auto Encoder (LSTM/ AE) 24
3. DETAILS OF METHODS 27
3.1. One Class Support Vector Machine Method (Oc-SVM)..... 27
3.2. k- Nearest Neighbors Method(k-NN)..... 27
3.3. Long Short-Term Memory/ Auto Encoder Model (LSTM/AE)..... 28
3.4. Convolutional Neural Network /Auto Encoder Model (CNN/AE) 28
3.5. Performance Metrics 29
4. RESULTS, AND DISCUSSIONS..... 33

4.1. Results and Discussions of Detection Anomaly 33
4.2. General Discussion of the Results 52
4.3. Discussion of Classification Models 54
4.4. Discussion on Deep Learning Models 57
4.5. Discussion of the Comparison with Previous Works 61
5. CONCLUSION 63
REFERENCES 65
CURRICULUM VITAE 72

LIST OF TABLES **PAGE**

Table 1.1. Summary of studies about anomaly detection. 8
Table 1.2. Overview of the datasets. 11
Table 3.1. The intervals for the parameters of the Oc-SVM methods. 27
Table 3.2. The intervals for the parameters of the k-NN method. 28
Table 3.3. The intervals for the parameters of the LSTM/AE models. 29
Table 3.4. The intervals for the parameters of the CNN/AE models. 29
Table 4.1. CPU Hold-out, split range (70 % TRAIN -30% TEST) 34
Table 4.2. CPU KFold, split range (10) 35
Table 4.3. MEMORY Hold-out, split range (70 % TRAIN -30% TEST) 35
Table 4.4. MEMORY KFold, split range (10) 36
Table 4.5. DISK Hold-out, split range (70 %-30%) 36
Table 4.6. .DISK KFold, split range (10) 37
Table 4.7. Comparing the results of accuracy for various techniques. 61

LIST OF FIGURES **PAGE**

Figure 1.1. KPI datasets. 11
Figure 2.1. The geometry interpretation of the Oc-SVM classifier 15
Figure 2.2. k-NN 18
Figure 2.3. Auto Encoder Diagram 20
Figure 2.4. The components of a typical CNN layers 23
Figure 2.5. Anomaly detection using LSTM with Auto Encoder 25
Figure 3.1. Confusion Matrix 30
Figure 4.1. Learning Curve of Oc-SVM, Hold-Out of CPU. 38
Figure 4.2. Learning Curve of k-NN, Hold-Out of CPU. 38
Figure 4.3. Learning Curve of CNN/AE, Hold-Out of CPU. 39
Figure 4.4. Learning Curve of LSTM/AE, Hold-Out of CPU 39
Figure 4.5. Learning Curve of Oc-SVM, K-Fold of CPU 40
Figure 4.6. Learning Curve of k-NN, K-Fold of CPU. 40
Figure 4.7. Learning Curve of CNN/AE, K-Fold of CPU 41
Figure 4.8. Learning Curve of LSTM/AE, K-Fold of CPU 41
Figure 4.9. Learning Curve of Oc-SVM, Hold-Out of Memory. 42
Figure 4.10. Learning Curve of k-NN, Hold-Out of Memory. 42
Figure 4.11. Learning Curve of CNN/AE, Hold-Out of Memory 43
Figure 4.12. Learning Curve of LSTM/AE, Hold-Out of Memory 43
Figure 4.13. Learning Curve of Oc-SVM, K-Fold of Memory. 44
Figure 4.14. Learning Curve of k-NN, K-Fold of Memory. 44
Figure 4.15. Learning Curve of CNN/AE, K-Fold of Memory 45
Figure 4.16. Learning Curve of LSTM/AE, K-Fold of Memory 45
Figure 4.17. Learning Curve of Oc-SVM, Hold-Out of Disk. 46
Figure 4.18. Learning Curve of k-NN, Hold-Out of Disk. 46
Figure 4.19. Learning Curve of CNN/AE, Hold-Out of Disk 47
Figure 4.20. Learning Curve of LSTM /AE, Hold-Out of Disk 47

Figure 4.21 Learning Curve of Oc-SVM, K-Fold of Disk..... 48
Figure 4.22. Learning Curve of k-NN, K-Fold of Disk. 48
Figure 4.23. Learning Curve of CNN /AE, K-Fold of Disk 49
Figure 4.24. Learning Curve of LSTM /AE, K-Fold of Disk 49
Figure 4.25. Accuracy of CPU, split range (70%), k-fold (10)..... 50
Figure 4.26. Accuracy of Memory, split range (70%), k-fold (10)..... 51
Figure 4.27. Accuracy of Disk, split range (70%), k-fold (10)..... 51

LIST OF ABBREVIATIONS

AD : Anomaly Detection

AE : Auto Encoders

CNN : Convolutional Neural Network

CM : Confusion Matrix

IDS : Intrusion Detection Systems

k-NN : k-Nearest Neighbors

LSTM : Long Short-Term Memory

Oc-SVM : One Class Support Vector Machine

1. INTRODUCTION

1.1. Overview of Anomaly Detection in Network Traffic

As applications over the Internet become increasingly complex, accurate description information and rapid monitoring of network traffic have become challenging tasks. So, on this section, we will touch some details of Intrusion Detection Systems (IDS), network traffic anomalies, machine, and deep learning.

The intrusion detection system is software that can identify and prevent network breaches. It can also alert users about suspicious activities before they become a full-fledged attack. IDS can enhance the security of network devices and valuable network data by identifying suspicious network traffic and drawing attention to it. Network needs strong security to protect information and internal and external network data transmissions. The network generates a lot of information every day through regular operations; the intrusion detection system helps to distinguish between necessary activity and less important information, by helping to decide which data to pay attention to.

Getting a detailed and accurate view of network activity through (IDS) can help demonstrate compliance, as intrusion prevention systems are designed to detect, organize, and alert incoming and outgoing network traffic, and identify the most important information, by filtering through network traffic. It can also give Intrusion Detection System a step when it comes to determining the compliance of the network and its devices, as IDS is designed to improve intrusion detection and prevention by filtering through the flow of traffic, this can save time, energy and resources while detecting suspicious activity before turning into a complete threat. Predict IDS also provides increased visibility into network traffic, which can help prevent and detect malicious activity, determine compliance status, and improve overall network performance, due to the fact that the more a private (IDS) system detects malicious activity on a network, the more it can adapt to attacks.

Two main types of IDS are Network Intrusion Detection System (NIDS), and Host Dependent Intrusion Detection System (HIDS). These systems are used to monitor network traffic and match traffic patterns to known attacks from collected data.

Through this, an intrusion prevention system can determine the unusual activities and define it as a cyber attack. In addition, it can also send an alert to selected administrators or technicians when it identifies suspicious activities. This allows them to quickly identify the root causes of the problem and prevent further attacks. All kinds of Networks are often exposed to anomalous behavior. For example, attacks or large data transfers in IP networks, the presence of intruders in surveillance systems, and sudden congestion in the network, so one of the biggest challenges that both network administrators and researchers face is detecting anomalies in network traffic, where the rapid accurate detection of anomalies is important to prevent many serious problems.

Intrusion detection systems primarily use two main methods which are; expect-based, and anomaly-based intrusion detection. Expect-based intrusion detection is designed to detect potential threats by comparing network traffic and log data with existing attack patterns; and anomaly-based intrusion detection is designed to identify unknown attacks, Additionally, a popular requirement analyzing at analyze real-world datasets is to define which state plainly swerve from majority data statse which are named anomalies also named outlier detection or novelty detection. The aim of detection anomalies is to determine all such instances at data which is probably caused by data errors, or at times point out the presence of a new, formerly unbeknown underlying process such as hackers (Guansong Pang et.al., 2020).

Because of the growing demand, anomaly detection requires a buoyant research area in wide scope for several decades, within early time exploration dating back as far as to 1960s. Some of these scopes are critical safety. Moreover, they are commercially or scientifically significant real-world applications such as fraud

detection (Charu C. Aggarwal, 2017), harsh climate event detection, mechanical mistake detection, defect detection, also risk management, compliance, security, financial surveillance, health and medical risk, and artificial intelligent safety. Anomaly detection plays increasingly significant role, highlighted in varied domains including data mining, machine learning, computer vision, and statistics.

Machine Learning (ML) techniques and Deep Learning (DL) techniques enable the development of algorithms to detect anomalies that allow the Intrusion Detection System (IDS) to establish the trust models whereas, Hybrid Intrusion Detection Systems use expect-based, and anomaly-based intrusion detection to increase the scope of the intrusion prevention system, which enables the identification of as many threats as possible, and the comprehensive intrusion detection system (IDS) can understand the evasion techniques used by cybercriminals to deceive the intrusion prevention system to believe that an attack has not occurred. These methods can include but not limited to hashing, low bandwidth attacks, pattern change evasion, spoofing of addresses or proxies.

Recently, deep neural networks have proliferated remarkably in outstanding performance in different fields. Deep learning is an offshoot of machine learning, which obtains high performance, and elasticity by means of learning for representing the data such as the nested hierarchy of notions in neural network layers. Deep learning outperforms classic machine learning with the scale-growing data (Raghavendra Chalapathy et. al., 2019). Deep learning has shown great capabilities in learning the expression and representation of complex data such as high-dimensional data, temporal data, spatial data, and graphic data, breaking through the boundaries of different learning tasks. Deep learning is used for detecting anomalies, referred to as deep anomaly detection. For anomaly detection, feature representations or anomaly scores are learned through neural networks. Deep learning introduces a large number of deep anomaly detection methods, which significantly outperform traditional anomaly detection in solving challenging detection problems in various practical applications (Guansong Pang et. Al., 2020). However, popular deep

learning techniques are not suitable for anomaly detection due to some unique characteristics of anomalies, such as rarity, heterogeneity, infinity, and the prohibitive cost of collecting large-scale anomaly data. Therefore, a lot of research has been devoted to designing deep learning techniques specifically for anomaly detection (Longbing Cao et. al., 2021).

Based on what we have explained above, the purpose of this study concentrates on developing classification-based models of machine learning techniques that can detect an anomaly in network traffic by using a Key Performance Indicator (KPI) dataset, and finding out which models have high performance and better accuracy based on the Confusion Matrix (CM).

1.2. Review of the Literature Studies

In literature, machine learning and deep learning techniques became one of the most common methods to discover anomaly, so many researchers have used this field so as to cluster and classify an anomaly. Below, some of the important studies are explained and showed in Table 1.1.

Taeshik Shon et. al., 2005, used genetic algorithm (GA) for feature selection and support vector machine (SVM) for packet classification to detecting attacks from internet anomalies also demonstrated that the proposed framework performed employed real-world MDS.

Dewan Md. Farid et. al., 2010, presented a new learning algorithm for adaptive network intrusion detection using naive bayesian classifier and decision tree which performs balance detections and keeps false positives at acceptable level for different types of network attacks and eliminates redundant attributes as well as contradictory examples from training data that make the detection model complex.

LI Han, 2010, improved the detection rate to decrease the false alarm rate by clustering analysis research method, intrusion detected based on data mining.

Reyadh Shaker Naoum et. al., 2013, designed an intrusion detection systems multilayer perception as trained using an enhanced resilient back propagation training algorithm for intrusion detection.

Reda M.Elbasiony et. al., 2013, designed a hybrid detection framework that depended on data mining classification and clustering techniques which were proposed by used random forests classification algorithm to build intrusion patterns automatically from a training dataset and used the k-means clustering algorithm for anomaly detection by clustering the network connections data to collect the most intrusions together in one or more clusters.

Gideon Creech et.al., 2014, introduced a new host-based anomaly intrusion detection methodology using discontinuous system call patterns, in an attempt to increase detection rates whilst reducing false alarm rates.

Ralf C. Staudemeyer, 2015, tested the performance on trained long short-term memory (LSTM) recurrent neural networks with the training data provided by the DARPA / KDD cup 99 challenge; where they tested all features and on extracted minimal feature sets respectively.

Naila Belhadj Aissa et. al., 2015, proposed a clustering based detection technique using a genetic algorithm named genetic clustering for anomaly based detection (GC-AD), GC-AD using a dissimilarity measure to form k clusters.

Shailendra Sahu et. al., 2015, used a new labeled network dataset, called Kyoto 2006+ dataset. In Kyoto 2006+ dataset, every instant is labeled as normal (no attack), attack (known attack) and unknown attack. Also, they used the Decision Tree (J48) algorithm to classify the network packet that can be used for NIDS for detecting the intrusion in networks.

Mohammed A. Ambusaidi, 2016, proposed a mutual information-based algorithm that analytically selects the optimal feature for classification. This mutual information-based feature selection algorithm can handle linearly and nonlinearly dependent data features, effectiveness was evaluated in the cases of network intrusion detection by an Intrusion Detection System (IDS), named Least Square

Support Vector Machine based IDS (LSSVM-IDS), which is built using the features selected by our proposed feature selection algorithm.

Xinlong Zhao et. al., 2016, proposed a new intrusion detection method based on improving K-means, which is designed to fit the characteristics and security requirements of cloud computing. The new method can find out the known attack as well as anomaly attack in the environment of cloud computing

Chuanlong Yin et. al., 2017, explored how to model an intrusion detection system based on deep learning, and they proposed a deep learning approach for intrusion detection using recurrent neural networks (RNN-IDS). Moreover, they studied the performance of the model in binary classification and multiclass classification, and the number of neurons and different learning rate and impacts on the performance of the proposed model.

Jonathan Goh et. al., 2017, presentsed a novel unsupervised approach to detect cyber-attacks in Cyber-Physical Systems (CPS), and they described an unsupervised learning approach using a Recurrent Neural network which was a time series predictor as their model.

Jun Inoue, Yoriyuki Yamagata, et. al., 2017, proposed and evaluated the application of unsupervised machine learning to anomaly detection for a Cyber-Physical System (CPS). they compared two methods: Deep Neural Networks (DNN) adapted to time series data generated by a CPS, and one-class Support Vector Machines (SVM).

Muhammad Hilmi Kamarudin, Tim Watson et. al., 2017, proposed an anomaly-based intrusion detection system using an ensemble classification approach to detect unknown attacks on web servers. The process involved removing irrelevant and redundant features utilising a filter and wrapper selection procedure.

Kai Peng, Victor C. M. Leung et. al., 2018, proposed an IDS system based on a decision tree and proposed preprocessing algorithm for the string number in the dataset to ensure the quality of the input data so as to improve the detection efficiency. They also used the decision tree method of their IDS system and then

compared this method with the NAIVE Bayesian method as well as the k-NN method.

Shijoe Jose, D. Malathi et. al., 2018, tried to provide a structured and comprehensive overview of the research on anomaly detection by the existing anomaly detection techniques and notice the strengths and weaknesses.

Alia Abu Ghazleh, Basel Magableh, et.al., 2019, presented an intelligent intrusion detection system based on radial basis function capable to handle all types of attacks and intrusions with high detection accuracy and precision through addressing the intrusion detection problem in the framework of interpolation and adaptive network theories.

Chafiq Titouna et. al., 2019, have proposed a model known as Distributed Outlier Detectionuujj Scheme (DODS) which detects anomalies from fully distributed way of operation and therek is no involvement of exchange of the message in neighbors. The effectiveness of this method is evaluated by many performance metrics and found outperforming.

Efrat Vilenski et. al., 2019, proposed an outlier detection method which allows users to detect faulty sensors creating anomalous data and find out quality sensors in largest sensors network. They used concept of pipelines which help agronomic operations and tasks of conserving the quality of information for automatic irrigation by finding malfunction dendrogram. Advanced visual analytic system combines automated multivariate statistical method with visualizations and helps interactive visualizations which provides the means for effective decision making based on large and complex datasets.

Zornitza Genova Prodanoff et. al., 2020, used modeling techniques such as Knuth’s Rule or Bayesian Blocks since the purposed of real-time traffic characterization in RFID networks has already been proposed. This study examined the applicability of using Voronoi polygon maps or alternatively, DBSCAN clustering, as initial density estimation techniques for computing 2-Dimentional Bayesian Blocks models of RFID traffic.

Guanglu Wei et. al., 2020, In order to study the application of deep learning in designing a network traffic anomaly detection device, the aim was to solve two common problems in the field of network error detection: distinct dependence and high false positive rate. They combined a convolutional neural network (CNN) with a neural network. Recurrent (RNN) A method for detecting anomalies in the network was proposed based on hierarchical learning of temporal-spatiotemporal features (HAST-NAD) on the basis of deep learning.

Anastasia Khudoyarova et. al., 2021, suggested a set of the most effective anomaly detection methods as well as recommendations on the underlying system architecture and they studied the application of machine learning methods, as well as spectral and statistical methods for real time network traffic anomaly detection.

Annie Gilda Roselin, et. al., 2021, proposed intelligent anomaly detection for extensive network traffic analysis with an Optimized Deep Clustering (ODC) algorithm.

Table 1.1 shows a number of previous studies about the detection of the anomaly with different methods of machine learning for each study.

Table 1.1. Summary of studies about anomaly detection.

Studies	Methods	PerformanceMetric	Values
Taeshik Shon, et. al., [2005]	GA, SVM	CR	87.74 %
Nouria Harbi, et. al.,[2010]	NBC, DT	DR FP	99.78 % 0.055 %
Li Han, [2010]	CA	DR FAR	97.70 % 2.15 %
Re. Sh. Naoum, et. at., [2013]	ERBP	DR	94.70 %
Reda M. Elbasiony, et. al., [2013]	RF, k-Means	DR FPR	96.38 % 7.35 %
Gideon Creech, et. al., [2014]	ELM	DR FAR	100 % 0.60 %
Ralf C. Staudemeyer, [2015]	LSTM	MSE, CM, ROC ACC Cost	- 93.82 % 0.2213

Shailendra Sahu, et. al., [2015]	DT(J48)	CM, ACC	97.23 %
Naila B. Aissa, and M. Guerroumi, [2015]	GC-AD	DR FPR	75-98 % 1.3-0.12 %
Xinlong Zhao, and W. Zhang, [2016]	k-Means	FPR FNR DR	3.56 % 7.65 % 32.4 %
Mohammed A. Ambusaidi, et. al., [2016]	LSSVM-IDS	ACC	97.33 %
Chuanilong Yin, et. al., [2017]	RNN-IDS	ACC	97.09 %
Jonathan Goh, et. al., [2017]	RNN-LSTM	CSM	90.00 %
Jon Inoue, et. al., [2017]	DNN Oc-SVM	F-measure	80.30 % 79.60 %
Tim Watson, et. al., [2017]	Logitboost-RF	DR ACC	99.10 % 99.45 %
Kai Peng, et. al., [2018]	DT	F-score	87.70 %
Shijoe Jose, et.al., [2018]	HIDS vs.NIDS	-	-
Chafiq Titouna, et. al., [2019]	DODS	C_{MAP}	AA
Muder Almiani, et. al., [2019]	SOM/k-Means	DR	96.66 %
Efrat Vilenski, et. al., [2019]	AMS	-	-
Zornitza Genova Prodanoff et. al., [2020]	DBSCAN	-	-
Guanglu Wei et.al., [2021]	HAST-NAD	ACC	99.91 %
Anastasia Khudoyarova, et.al., [2021]	Improve the Classical Kalman Filter	-	-
Annie Gilda Roselin, et.al., [2021]	AE/k-Means ODC	NMI ACC NMI ACC	75 % 82.93 % 95.70% 98.30 %

GA, Genetic Algorithm; **SVM**, Support Vector Machine; **CR**, Correction Rate; **NBC**, Naive Bayesian Classifier; **DT**, Decision Tree; **DR**, Detection Rate; **FP**, False Positive; **CA**, Clustering Analysis; **FAR**, False Alarm Rate; **ERBP**, Enhanced

Resilient Back Propagation Algorithm; **RF**, Random Forests; **FPR**, False Positive Rate; **ELM**, Extreme Learning Machine; **LSTM**, Long Short Term Memory; **MSE**, Mean Squared Error; **CM**, Confusion Matrix; **ROC**, Receiver Operating Characteristic; **ACC**, Accuracy; **GC-AD**, Genetic Clustering Anomaly Detection; **FNR**, False Negative Rate; **LSSVM-IDS**, Least Square Support Vector Machine-Intrusion Detection System; **RNN-IDS**, Recurrent Neural Networks; **CSM**, Cumulative Sum method; **DNN**, Deep Neural Networks; **Oc-SVM**, One Class Support Vector Machine; **RF**, Random Forests; **HIDS**, Host Intrusion Detection System; **NIDS**, Network Intrusion Detection System; **DODS**, Distributed Outlier Detection Scheme; $\mathbf{C_{MAP}}$, Maximum A Posteriori Concept; **SOM**, Self Organize Map; **AMS**, Automated Multivariate Statistical; **HAST-NAD**, Hierarchical Learning of Temporal-Spatiotemporal Features; **AE**, Auto Encoder; **ODC**, Optimized Deep Clustering; **NMI**, Normalized Mutual Information.

1.3. Overview of the Datasets

To ensure to detect anomalies network traffic, we need to keep an eye on various KPIs for the utilization, and allocation of devices, to obtain steady and credible services.

KPIs are time-series data that consist of a set of values ordinarily created by continuous mensuration over time, like network throughput, serving latency, page views, and the number of online users. However, due to the mystery of anomalies, the lack of diversity of KPIs (periodic, stable, and fluctuating), anomaly detection for various KPIs has been a great challenge.

In this thesis, the dataset was prepared by using key performance indicators (KPI datasets) to measure device health through the use of the data of CPU and memory utilization besides memory allocation. Datasets were aggregated every ten minutes in a period of 24 hours for twenty-one days. The range index are (6300 entries, from 0 to 6299), and the data columns are (4 columns as total), as shown below in table 1.2., and figure 1.2.

Table 1.2. Overview of the datasets.

#	Column	Non-Null Count	Data type
0	Timestamp	6300 non-null	object
1	kpi_name	6300 non-null	object
2	kpi_value	6300 non-null	int64
3	anomaly_label	6300 non-null	int64

Figure 1.1. KPI datasets.

1.4. Motivation, Purpose and Contributions of This Thesis

In the previous works, we noticed that there were several academic studies, proven worthily in developing Intrusion Detection Systems (IDS), and prediction of network traffic. Moreover, the attempts were limited to developing the detection models by utilizing Outlier Detection Datasets (ODDS) as; KDDCUP99, DARPA, and CAIDA, by depending on machine learning techniques, or deep learning techniques and doing comparisons, such as; Long Short Term Memory (LSTM), Desition Tree (DT), Support Vector Machine (SVM), Recurrent Neural Network (RNN), Random Forests (RF), Optimized Deep Clustering (ODC), AutoEncoder

(AE), Convolutional Neural Network (CNN), and Long Short Term Memory / Auto Encoder (LSTM/AE), Convolutional Neural Network / Auto Encoder (CNN/AE) models, in addition to One Class Support Vector Machine (Oc-SVM), k-Means, and Dbscan methods. However, to the best of our knowledge, there are not enough works applied to recognize the health of Devices, by utilizing KPI dataset of CPU, DISK, and MEMORY usages. So, this study aims at applying deep learning-based and classification-based methods to detect an anomaly and recognize the health of device based on Key Performance Indicator (KPI) datasets, and choosing the best method gives the high detection accuracy, taking into consideration the studies in the literature. Proposed techniques are; CNN/AE, LSTM/AE, Oc-SVM, and k-NN, which are common machine learning methods.

The main elements which has been utilized in this study, and made it possible to apply a method containing more features together in the literature, can be listed as follows:

1. Dataset was gathered; Key Performance Indicator (KPI) for daily CPU usage, as well as DISK, and MEMORY usages in network traffic.
2. Two different approaches to detection were suggested; two models of deep learning-based methods are; CNN/AE, and LSTM/AE.On the other hand; two models of classification-based methods are; Oc-SVM, and k-NN.
3. Defined the part of the dataset which will be used for train/test the models by two techniques, the 1st technique is hold-out; (70 ratio) for train, and (30 ratio) for test, and the 2nd technique is k-fold; (10 n_sample) for train, and test.
4. Applied different values of parameters for detection methods to obtain the best hyperparameters depend on accuracy by use grid search technique.

5. High Accuracy (ACC) rates results were compared; based on Confusion Matrix (CM) for each model.
6. The model that has a high accuracy was compared; with previous works to find the performance and better accuracy (ACC) to detect anomalous and recognizing the health of devices.

2. METHODS SUMMARY

In this section, basic concepts of each method used in this study are introduced.

2.1. One Class Support Vector Machine (Oc-SVM)

One Class SVM (Oc-SVM) was usually defined as a popular unsupervised method to disclose anomalies (Raghavendra Chalapathy, 2019), that have built smooth boundaries around the majority of an eventuality data mass (Bernhard Schölkopf, Smola et. al. 2001). Unsupervised detecting anomalies aimed to separate between normal and abnormal data by finding out rules; we have two main steps to understand how Oc-SVM works, first step is training, where training dataset is used to generate the detector of Oc-SVM, and the second step is to detect anomalies with the trained detector in the data as explained in figure 2.1.

Figure 2.1. The geometry interpretation of the Oc-SVM classifier (Larry M, 2001).

More precisely, a training dataset without any information, $x_i \in R^n$, i=1,2,..., l , where (i) is denote to the number of data points in train set, (R^n) is the input, (n) is the dimension of the input, Φ(x) is a function to convert x from the input to the feature (F). A hyper-plane function f(x) in the feature F is constructed in the (2.1) equation.

$$f(x) = w^T \Phi(x) - \rho \tag{2.1}$$

In 2.1. $\{ \Phi(x_i), i = 1,2,\ldots,l \}$ is used to split as many as potentially of the mapped vectors from the origin, *w* is the norm perpendicular to the hyper-plane and ρ is the bias of the hyper-plane. In order to solve w and ρ, we need to solve the follow optimization problem.

$$\min \frac{1}{2} w^T w + \frac{1}{\nu l} \sum_{i=1}^{l} \xi_i - \rho \tag{2.2}$$
$$\text{subject to } w^T \Phi(x) \geq -\rho - \xi_i, \xi_i \geq 0, i = 1,2,\ldots, l$$

where ξ_i are slack variables, and ν ∈(0,1) which is the parameter that controls a trade off between maximizing the distance of the hyper-plane from the origin and the number of data points contained by the hyper-plane. Thus, to solve (2.2) equation, multiplier α_i is introduced to each x_i. therefore, a dual problem can obtain, and (2.3) equation leads to solving it.

$$w = \sum_{i=1}^{l} \alpha_i \phi(x_i) \tag{2.3}$$

where $0 \leq \alpha_i \leq \frac{1}{\nu l}$.

For that reason, the decision function *f(x)* becomes a nonlinear function as in shown in equation (2.4).

$$f(x) = \sum_{i=1}^{l} \alpha_i K(x_i, x) - \rho \tag{2.4}$$

In equation (2.4), where $K(x_i,x) = \phi(x_i)^T \phi(x)$, *K* is defined as a kernel function in the input space (V.N. Vapnik, 1999). There are many admissible choices for kernel function, Radial Basic Function (RBF), is the most widely used kernel in SVM (S.S. Keerthi, et.al. 2003); in equation (2.5) RBF was explained.

$$K(x_i, x) = e^{(-\|x_1 - x_2\|^2 / 2\sigma^2)} \tag{2.5}$$

2.2. k-Nearest Neighbors

k-NN, which is a well known and used classification algorithm and consider one of the simplest supervised ML methods, proposed by Evelyn Fix and Joseph Hodges in 1951 (Evelyn Fix et. al., 1951), it used widely for classification and also occasionally using for regression problems , So;It is well used to detecet the anomolious and building recommender systems etc.

The main work of this algorithm is that depends on observations that are close and similar to each other are considered normal and which are distant from the cluster of similar observations and which be single are considered abnormal observations as figure 2.2 shows below.

Figure 2.2. k-Nearest Neighbors.

So further, when the k-NN comes to detect the outlier it work as an unsupervised approach due there isnt genuine “learning” concerned in the operation also there isnt pre-determined labeling of “abnormal” or “normal” in dataset but depend on threshold values instead that, *k* is deem an most important parameter in k-NN so different values for *k* lead to a difference in the performance result (Salima Omar et. al., October 2013). It utilizing for calculates the approximate distances between points which is be unlike upon the input vectors, after that can be determine unlabeled point the class of its k-nearest neighbors.

The most commonly useable distance function in k-NN is Euclidean function which is used to compute the approximate distances between points which is be unlike upon the input vectors, after that can be determine unlabeled point the class of its k-nearest neighbors,as explained in equation (2.6).

$$\text{Euclidean } \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \qquad (2.6)$$

In equation (2.6) ,there are two input vectors x indicate to the distance between a point x $(x_1, x_2,...,x_n)$, y indicate to the distance between a point y $(y_1,y_2,....,y_n)$.

$$Manhattan\ \sum_{i=1}^{k} |x_{i-}\ y_i| \tag{2.7}$$

In (2.7) the Manhattan distance equation. In other words, It applying the distance function between two points (x_1, y_1) and x_2 ,y_2 are : $|x_{i-}\ y_i|$

$$Minkowski = \ (\sum_{i=0}^{k}(|x_{i-}\ y_i|)^q)^{\frac{1}{q}} \tag{2.8}$$

In (2.8) the Minkowski equation, The distance of order p (where p is an integer) applied between two points: x(x_1, x_2,…,x_n), and y (y_1,y_2,….,y_n), (Amit V Kachavimath et. al., 2020).

2.3. Auto Encoders(AE)

Auto Encoder (AE) is a traineeship for rejuvenating input vectors as outputs such as in the neural network. (I. Goodfellow, et. al., 2016). An AE is made of three main layers which correspond to (i) the input layer to take in the features, (ii) the latent representation layer of the features and (iii) the output layer which is the reconstruction of the features. (Quoc Phong Nguyen, et. al., 2019); input and output layer alike consists of *N* nodes, whilst hidden layer contains *K* nodes; AE was named under completed when $K < N$.

Hidden layer of AE is familiar by means of different names in literature including bottleneck, discriminative, coding or abstraction layer. We will enjoin to the name ‘‘bottleneck’’ and use these names mutually if it has desired (SHERAZ NASEER, et. al., 2018).

The AE consists of two parts called encoder and decoder respectively as explained in figüre (2.3). The encoder maps the input into its latent representation while the decoder attempts to reconstruct the features back from the latent representation. The encoder may be deep in the sense that information from the input

is passed through several in a supervised deep learning model; likewise for the decoder (Quoc Phong Nguyen, et. al., 2019).

Figure 2.3. Auto Encoder Diagram

Learning mission in AE is under-complete, so; it needs to take and grasp the most worthy merit of train data at the bottleneck step, likewise, input is recreated at output layers. This can be achieved out of minifying loss function L(x, g(f (x))) variation of g(f (x)) foras train data (x), (SHERAZ NASEER,et. al., 2018).

$$z^{l+1} = w^{l}.\alpha^{l} + b^{l} \tag{2.9}$$

In equation (2.9), w^{l} and b^{l} indicates the parameter of each coat where $\boldsymbol{W}$ and $\boldsymbol{b}$ are ***weights*** and ***bias***; related to the connection between coat l and coat $l+1$, Autoencoder encoding step in accumulated was defined as an operating of encoding step for each layer as forwarding order. $\alpha^{l} = f(z^{l})$ where $a, f(.), and\ z$ denoting to activation outputs, activation function, and weighted totality inputs in coat l, sequentially.

Decoding is executed by overrunning the decoding stack in inverse order $\alpha^{n+1} = f(z^{l})$ as shown up below in equation (2.10).

$$z^{n+l+1} = w^{n-l}.\alpha^{n+l} + b^{n-l} \tag{2.10}$$

where α^{n} denotes activations of bottleneck layer.

The distinctive of deep AEs is numerous. Some of these merits stem from a general approximator concept (Kurt Hornik, 1990). That means a feed-forward neural network with fully one hidden layer at least can elect an approximation from any function to arbitrary accuracy. In other words, when deep AEs have more than an encoder layer that can affect on exponential reduction in the computation and the amount of train data needed for symbolizing numerous function (I. Goodfellow, et. al., 2016).

- Scattered Auto Encoders

Define autoencoder as scattered autoencoder when traineeship criterion has to rebuild error and sparsity at the same time $\Omega(h)$ at bottleneck coat, below, (2.11) is an equation to represent the Sparse AE.

$$\mathrm{L}(\mathrm{x}, \mathrm{g}(\mathrm{f}(\mathrm{x}))) + \Omega(h) \tag{2.11}$$

Sparsity assists to handicap overfitting of an AE which would do it another way labor as an identity function for train data (SHERAZ NASEER, et. al., 2018).

- Denoising Auto Encoders

Denoising Auto Encoders (DAE) is defined as training foretelling original undamaged data versions of comparable as output by receiving damaged data as input, (Pascal Vincent, et. al., 2010). $\mathrm{L}(\mathrm{x}, \mathrm{g}(\mathrm{f}(\bar{x})))$ DAE minimizes objective

function, where $\bar{x}$ symbolized a corrupte version of x using some noise to reduce over fit.

- Contractive Auto Encoder

Contractive Auto Encoders (ContAEs) are defined as learning representations that are sturdy in respect of small changes in training data (Salah Rifai, et. al., 2011).

- Convolutional Auto Encoders

Convolutional Auto Encoders (ConvAE) are learning features utilizing simple stochastic gradient descent (SGD), they also find out good initialization for (CNNs), which avert the distinguished minimalist non-cambered objective functions in diverse deep learning problems (Jonathan Masci, et. al., 2011).

2.4. Convolutional Neural Network /Auto Encoder (CNN/ AE)

Convolutional Neural Network (CNN), is one of the most famous neural networks after the best results were shared on the ImageNet Large Scale Visual Recognition Competition (ILSVRC), (Alex Krizhevsky, et. al., 2017), (Olga Russakovsky, et. al., 2015). The CNN method was inspired by the visual cortex for the eye, that is responsible for detecting light in small parts of the visual partions (Weibo Liu, et. al., 2017); the visual cortex consists of, two kinds of layers are convolution and sampling layers used to extract, and map features sequentially as in CNN. The input assumed is a two-dimensional image with a set of pixels. For this reason, CNN has been basically used to classify images, and network security (Taejoon Kim, et. al., 2018). The sparse interactions, equivariant representations, and parameter sharing of convolution can be used to improve machine learning fields. The layers of convolution have three important points explained in figure (2.4), they are; the layer performs several convolutions in parallel to produce a set of linear activations, each linear activation is run through a nonlinear activation

function, and a pooling function is used to modify the output of the layer further (Gonzalo Marin, et.al., 2019).

Figure 2.4. The components of a typical CNN layers.

We can summarize the CNN categories as follows; 1D-CNN, 2D-CNN, and 3D-CNN where 1D- is often used with sequential data or languages, 2D-is oftentimes used with data like images, and 3D-is perfect for data like video (Ren-Hung Hwang, et.al., 2020). The most appropriate usage of CNN is data frames of one or multiple dimension arrays that can appear on features (Zhitang Chen, et.al., 2017).

Furthermore, for network traffic analysis, we can use 1D-CNN, for expounding xi is a s of bytes beginning from i^{th} byte of input, as shown *i* below in equation (2.17).

$$i = 1, \ldots, \mathrm{n} \times \mathrm{l} - \mathrm{s} + 1. \tag{2.17}$$

In equation (2.18,) the convolution operations were explained, where m is the number of filters. xi the applied input to produce a new feature. As a result, a feature h_i^k from the kth filter to the window xi is generated.

$$h_i^k = \mathrm{f}\,(w^k\ \mathrm{xi} + b_k\,) \tag{2.18}$$

Where; f is a ReLU function, W^k is the weight vector of the kth filter, and b_k is a bias term or the offset of the kth filter. The filter is applied to the possible window of *s* bytes in the input to produce a feature map (Wei Wang, et. al., 2017).

$$\mathrm{h} = [\mathrm{h1}, \mathrm{h2}, \ldots, h_{\mathrm{nl}} - \mathrm{s} + 1] \tag{2.19}$$

In Equation (2.19), the aggregation process is applied and the maximum value (h) is taken as a feature of the next layer.

Multiple wrapping layers and stacking layers to extract high-level features are used; then it is passed to a completely continuous dense layer which results in the probability distribution of the type of benign fluxes (Ren-Hung Hwang, et.al., 2020). There are two different types of pooling layers: average pooling layer and max pooling layer, denoted as denoted in equations (2.20, and 2.21).

- Average pooling;

$$\mathrm{favg(x)} = \frac{1}{N} \sum_{\mathrm{i}=1}^{N} \mathrm{xi} \tag{2.20}$$

- Max pooling;

$$\mathrm{fmax(x)} == \mathrm{max(xi)} \tag{2.21}$$

where x denotes a vector of input data with activation values and N indicates a local pooling region (Ritesh K. Malaiya, et. al., 2018)

Among them, the max pooling layer is adopted because it is used as the default pooling layer more frequently than the average pooling layer in the CNN model (Chen-Yu Lee, et. al., 2016).

2.5. Long Short-Term Memory/ Auto Encoder (LSTM/ AE)

Long Short-Term Memory (LSTM), is an optimization method used to find artificial neural networks weights to avoid long-term dependency problems (Sara A.

Althubiti, et. al., 2018). this method was explicitly proposed as a solution to avoid the vanishing gradient problem in recurrent neural networks (RNN) (Felix A. Gers, et. al., 2000).

In briefly the vanishing gradient problem, Known in recurrent neural network (RNN) can be defined as the update of the weight values in the learned model. However, in case the gradient is very small, the model can not learn efficiently. Thus, layers that get a small gradient update in RNN will stop learning, and usually, this issue happens in earlier layers. So, The primary goal of LSTM is to achieve disappearing gradient descent. in other words, LSTM networks are a type of recurrent neural network (RNN) with special units called 'memory cells' (Sepp Hochreiter and Jurgen Schmidhuber, 1997),(Sepp Hochreiter, 1998).

LSTM uses the mechanism of control gates to regulate the flow of information, these gates are called input, output, and forget; The forget gate keeps a fraction of previous information, while the output gate is responsible for choosing how much information will output, and the input gate is responsible for getting new information (Mahmoud Said Elsayed, et. al., 2020), as shown below in figure(2.5).

Figure 2.5. Anomaly detection using LSTM with Auto Encoder

An RNN is a connectivity pattern that perform computations on a sequence of vectors $x1,\cdots,xn$ using a recurrence formula of the form $ht = f\theta\ (ht-1, xt)$,

where f , at any timestamp, an activate function and θ, and a parameter, are utilized to procedure series with qualitative lengths. h_t is known as a hidden vector, used to supply an implementation synopsis for each vector x until updated time-step xn. RNNs are usage longitudinal combinations of xt, $ht - 1$, have weakly conjugation between inputs and hidden cases (Yuhuai Wu, et. al., 2016), LSTM form is shown below in equation (2.21).

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} sigm \\ sigm \\ sigm \\ tanh \end{pmatrix} W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix} \tag{2.21}$$

In equation above, *sigmoid* and *tan*^{h} professions are applied. A variable W have distances $[4H * (D + H)]$, $i, f, o \in R^H$ it look a like a binary ports that domination of memory cell, and $g \in R^H$ are permit gradients to be divided evenly over backpropagation (SHERAZ NASEER,et. al., 2018).

3. DETAILS OF METHODS

3.1. One Class Support Vector Machine Method (Oc-SVM)

The accuracy of the Oc-SVM model depends on the hyperparameter which is used to control the sensitivity of the support vectors which separate all the data points from the origins "in feature space F ", and maximizes the distance from this hyperplane to the origin.

In the Oc-SVM model, it trained only on the ‘normal’ data to learn the boundaries of these points. Then, it is able to classify any points that lie outside the boundary i.e. outliers. The intervals for the parameter values of the Oc-SVM methods are given as specificied in Table 3.1.

Table 3.1. The intervals for the parameters and values of the Oc-SVM methods.

Parameters	Values
Hyper parameter (nu)	0.5
Gamma	auto
Kernel	Rbf

3.2. k-Nearest Neighbors Method

The accuracy of the the k-Nearest Neighbors method (k-NN) depends on the distance when its utilized for the classification, wherein An object is classify through disclose a plurality of its neighbors, with the object being allocate to the category most common amongst its k nearest neighbors , In general; k is consider a positive integer. The intervals for the parameters and values of the k-NN method are shown in Table 3.2.

Table 3.2. The intervals for the parameters and values of the k-NN method.

Parameters	Values
n-neighbors	5
weights	uniform

3.3. Long Short-Term Memory/ Auto Encoder Model (LSTM/AE)

The success of an LSTM is related to neuron numbers (hidden units) in the hidden layer and the number of epochs.

The number of neurons affects the learning ability of the network. Generally, more extra neurons would be capable to learn structure of the problem more at the cost of longer training time. More learning capacity further creates the problem of potentially over fitting the training data. The number of epochs is the number of times the whole training set pass through the network. The number of epochs should increase until a small gap between the test error and the training error is seen. The intervals for the parameters values of the LSTM models are shown in Table 3.3.

Table 3.3. The intervals for the parameters and values of the LSTM/AE models.

Parameters	Values
Mean	None
Sd	None
Units	64-32
Rate	0.5-0.2
Return_sequences	True

3.4. Convolutional Neural Network /Auto Encoder Model (CNN/AE)

The accuracy of the CNN model depends on number of neurons and the activation function. Having a small number of neurons may attend to underfitting while having more number of neurons are normally not harmful.

In the CNN models, the Rectified Linear Units (ReLU) activation function has been used. It has become the default activation function for several types of neural networks because a model that utilizes it is easier to train and usually achieves better performance. Moreover, the flatten layer is used between the convolutional

layers and the dense layer to reduce the feature maps to a single one dimensional vector. The intervals for the parameters values of the CNN/AE models are shown in Table 3.4.

Table 3.4. The intervals for the parameters and values of the CNN/AE models.

Parameters	Values
Mean	None
Sd	None
Kernel_size	1
Padding	'Valid'
Strides	1
Activation	'relu'
Input_dim	3
Activation	'sigmoid'

3.6. Performance Metrics

As a standard evaluation metric for İntrusion Detection Systems (IDSs), accuracy is considered as a metric that estimates the overall percentages of detection and false alarms an IDS model produces, which reflects the overall success rate of any IDS. The accuracy is evaluated by the confusion matrix, which have some of the terms such as; the True-Positive (TP), and True-Negative (TN) that denote correctly predicted conditions, whilst the False-Positive (FP), and False-Negative (FN) misclassified ones.

In other words, TPs and TNs refer to correctly classified attack and normal records, respectively, and conversely, FPs and FNs refer to misclassified normal and attack records, respectively as shown in Figure (3.1).

Figure 3.1. Confusion Matrix

These four terms are used to calculate Accuracy, Precision, Recall, and Area Under Curve (AUC) as elucidated in formulas from (3.1) to (3.7). The performance will be evaluated by the capability of a system to handle network traffic with high accuracy and low loss while running in real-time.

$$\text{Accuracy} = \frac{\Sigma TN+TP}{\Sigma TP+FP+TN+FN} \tag{3.1}$$

$$\text{Precision} = \frac{\Sigma TP}{\Sigma TP+FP} \tag{3.2}$$

$$\text{Recall} = \frac{\Sigma TP}{\Sigma TP+FN} \tag{3.3}$$

$$\text{True Positive Rate (TPR)} = \frac{\Sigma TP}{\Sigma TP+FN} \tag{3.4}$$

$$\text{False Positive Rate (FPR)} = \frac{\Sigma FP}{\Sigma FP + TN} \tag{3.5}$$

$$\text{True Negative Rate (TNR)} = \frac{\Sigma TN}{\Sigma TN + FP} \tag{3.6}$$

$$\text{False Negative Rate (FNR)} = \frac{\Sigma FN}{\Sigma FN + TP} \tag{3.7}$$

4. RESULTS, AND DISCUSSIONS

4.1. Results and Discussions of Detection Anomaly

In this dissertation, four different methods including Oc-SVM, k-NN, CNN/AE, and LSTM/AE were used to classify any kind of anomalies in network traffic, and find out which method has higher accuracy to ensure the health of the device.

To adopt classification models, and deep learning models; two techniques were applied to splits the dataset for each method. the 1st technique; one combination of the hold-out split was used. The split ambit is 70% for training and 30% for testing, respectively. While the 2nd technique; 10 of k-fold splits have been used to build machine learning models. Taking into consideration that these techniques were applied with grid search to tunning hyperparameters to get the best parameter for each method.

Tables 4.1, to 4.6 show the detection models' results of three experiments for CPU, Memory, and Disk are; key performance indicator datasets. We have developed machine learning models using KPI dataset that is labeled as normal and abnormal. The observations that have a reconstruction error greater than the threshold will be classified as freeze, whereas the ones with a reconstruction error less than the threshold as non-freeze, traffic data. We have illustrated the efficacy of the proposed approach by reporting different hold-out and k-fold values and representing their impact on Accuracy, Area Under Curve (AUC), Precision, and Recall. Tables 4.1, to 4.6 summarize the performance of different values in terms of evaluation metrics.

In this section, three experiments to detect the anomaly, and ensure the health of the device are displayed. the model response performance of each method is presented, where the model response is conducted to find the best accuracy rates for CPU, Memory, and Disk as shown below, the results for each method are displayed separately in tables, and plots.

The response of the model, which uses the KPI dataset as an input for classification and deep learning methods is collected as explained in the previous section. classification and deep learning methods were run on these datasets to detect anomalies where the hold-out split range is 70% for training, and 30% for testing, also for k-fold split is10, and tuning hyperparameter depending on the grid search to get the best accuracy, respectively for all methods.

Table 4.1. CPU Hold-out, split range (70 % TRAIN -30% TEST)

Model Name	Accuracy	Auc	Precision	Recall	Best Model Parameter
Oc-SVM	0.620	0.390	0.600	0.610	Kernel = 'rbf', Gamma = 'auto', nu = 0.1
k-NN	0.900	0.900	0.990	0.990	N_neighbors = 3
CNN/AE	0.996	0.998	0.990	0.998	Epoch = 50, batch size = 20
LSTM/AE	0.989	0.996	0.971	0.996	Epoch = 100, batch size = 60

Table 4.2. CPU KFold, split range (10)

Model Name	Accuracy	Auc	Precision	Recall	Best Model Parameter
Oc-SVM	0.430	0.560	0.440	0.440	Kernel = 'rbf', Gamma = 'auto', nu = 0.3
k-NN	0.980	0.990	0.990	0.980	N_neighbors = 3
CNN/AE	0.997	0.997	0.979	0.999	Epoch =50, batch size = 20
LSTM/AE	0.992	0.999	0.992	0.998	Epoch = 90, batch size = 30

Table 4.3. MEMORY Hold-out, split range (70 % TRAIN -30% TEST)

Model Name	Accuracy	Auc	Precision	Recall	Best Model Parameter
Oc-SVM	0.320	0.660	0.350	0.340	Kernel = 'rbf', Gamma = 'auto', nu = 0.6
k-NN	0.900	0.990	0.990	0.990	N_neighbors = 2
CNN/AE	0.996	0.999	0.989	1.000	Epoch = 30, batch size = 20
LSTM/AE	0.994	0.999	0.983	0.996	Epoch = 60, batch size = 40

Table 4.4. MEMORY KFold, split range (10)

Model Name	Accuracy	Auc	Precision	Recall	Best Model Parameter
Oc-SVM	0.590	0.430	0.560	0.570	Kernel = 'rbf', Gamma = 'auto', nu = 0.1
k-NN	0.970	0.999	1.000	1.000	N_neighbors = 2
CNN/AE	0.995	0.999	0.986	1.000	Epoch =70, batch size = 40
LSTM/AE	0.995	0.999	0.986	0.998	Epoch = 90, batch size = 30

Table 4.5. DISK Hold-out, split range (70 %-30%)

Model Name	Accuracy	Auc	Precision	Recall	Best Model Parameter
Oc-SVM	0.570	0.430	0.560	0.570	Kernel = 'rbf', Gamma = 'auto', nu = 0.1
k-NN	0.900	0.990	1.000	1.000	N_neighbors = 3
CNN/AE	0.989	0.997	0.968	1.000	Epoch = 100, batch size = 30
LSTM/AE	0.976	0.996	0.933	0.994	Epoch = 100, batch size = 60

Table 4.6. .DISK KFold, split range (10)

Model Name	Accuracy	Auc	Precision	Recall	Best Model Parameter
Oc-SVM	0.470	0.510	0.500	0.490	Kernel = 'rbf', Gamma = 'auto', nu = 0.1
k-NN	0.960	0.999	1.000	1.000	N_neighbors = 2
CNN/AE	0.998	0.999	0.987	1.000	Epoch = 60, batch size = 16
LSTM/AE	0.995	0.999	0.996	1.000	Epoch = 100, batch size = 16

Figure 4.1 to Figure 4.24 illustrate the learning curve of all methods of prediction models separately, where the split range is (70%-30%) for hold-out, (10) for k-fold, for all the observed results above of detection models separately for dataset.

Figure 4.1. Learning Curve of *Oc-SVM*, Hold-Out of CPU.

Figure 4.2**.** Learning Curve of *k-NN*, Hold-Out of CPU.

Figure 4.3. Learning Curve of *CNN/AE*, Hold-Out of CPU. (50 Epoch, 20 Batch size)

Figure 4.4. Learning Curve of *LSTM/AE*, Hold-Out of CPU. (100 Epoch, 60 Batch size)

Figure 4.5. Learning Curve of *Oc-SVM*, K-Fold of CPU.

Figure 4.6**.** Learning Curve of *k-NN*, K-Fold of CPU.

Figure 4.7**.** Learning Curve of *CNN/AE,* K-Fold of CPU. (50 Epoch, 20 Batch size)

Figure 4.8**.** Learning Curve of *LSTM/AE,* K-Fold of CPU. (90 Epoch, 30 Batch size)

Figure 4.9**.** Learning Curve of *Oc-SVM*, Hold-Out of Memory.

Figure 4.10**.** Learning Curve of *k-NN*, Hold-Out of Memory.

Figure 4.11**.** Learning Curve of *CNN/AE*, Hold-Out of Memory. (30 Epoch, 20 Batch size)

Figure 4.12**.** Learning Curve of *LSTM/AE*, Hold-Out of Memory. (60 Epoch, 40 Batch size)

Figure 4.13. Learning Curve of *Oc-SVM*, K-Fold of Memory.

Figure 4.14**.** Learning Curve of *k-NN*, K-Fold of Memory.

Figure 4.15. Learning Curve of *CNN/AE*, K-Fold of Memory. (70 Epoch, 40 Batch size)

Figure 4.16**.** Learning Curve of *LSTM/AE*, K-Fold of Memory. (90 Epoch, 30 Batch size)

Figure 4.17. Learning Curve of *Oc-SVM*, Hold-Out of Disk.

Figure 4.18. Learning Curve of *k-NN*, Hold-Out of Disk.

Figure 4.19. Learning Curve of *CNN/AE*, Hold-Out of Disk. (100 Epoch, 30 Batch size)

Figure 4.20. Learning Curve of *LSTM /AE*, Hold-Out of Disk. (100 Epoch, 60 Batch size)

Figure 4.21 Learning Curve of *Oc-SVM*, K-Fold of Disk

Figure 4.22. Learning Curve of *k-NN*, K-Fold of Disk.

Figure 4.23. Learning Curve of *CNN /AE*, K-Fold of Disk. (60 Epoch, 16 Batch size)

Figure 4.24. Learning Curve of *LSTM /AE*, K-Fold of Disk. (100 Epoch, 16 Batch size)

Figure 4.25 to Figure 4.27 illustrate the average Accuracy of the *Oc-SVM*, *k-NN*, *LSTM/AE*, and *CNN/AE* methods of anomaly detection models separately. The accuracy rates between hold-out and k-fold in the whole dataset on the average, and the other paradigms; while the Figures show the average accuracy of all methods for anomaly detecting models.

Figure 4.25. Accuracy of CPU, split range (70%), k-fold (10).

Figure 4.26. Accuracy of Memory, split range (70%), k-fold (10).

Figure 4.267. Accuracy of Disk, split range (70%), k-fold (10).

4.2. General Discussion of the Results

In the first aspect; according to the observed results of the completed study, it can be seen that the deep learning models of detecting anomaly yield lower error rates as opposed to classification models for all methods. In novelty detection models, for deep learning methods, the favorable order for the methods is *CNN/AE*, and *LSTM/AE*, the mean of the Accuracy values being 0.998, and 0.995, respectively. As for classification methods, the favorable order for the methods appears as *k-NN*, and *Oc-SVM*, the mean of the Accuracy values being 0.980, and 0.620 respectively.

For all data split (70% train- 30% test) in hold-out techniques, and (10) in k-fold techniques of detection methods with best hyperparameters found by grid search, the inclusive rating of the models in terms of prediction execution based on the average Accurancy are; *CNN/AE*, *LSTM/AE*, *k-NN*, and *Oc-SVM* for 70% split, where the average Accurancy are; 0.993, 0.986, 0.900 and 0.503, respectively; on the other hand *CNN/AE*, *LSTM/AE*, *k-NN* and *Oc-SVM* for the 10 k-fold, where the average Accurancy are; 0.996, 0.994, 0.970 and 0.496; respectively.

Of all the models of deep learning in this thesis, six picked numbers of an epoch (30, 50, 60, 70, 90, and 100), and batch size (16, 20, 30, 40, 60, and 70) have been used for tuning the hyperparameter of models. With regard to deep learning models, for split 70%, in general, the 1st pick-CPU dataset is 50, 100 for epochs; 20, 60 batch sizes, yield the best values for accuracy; in addition, the 2nd pick-memory dataset are 30, 60 for epochs; 20, 40 batch sizes, yield the best values for accuracy; as well, in the 3rd pick-disk dataset is 100, 100 for epochs; 30, 60 batch sizes, yield the best values for accuracy in *CNN/AE*, *LSTM/AE*; respectively. generally, the 1st, and 2nd pick is (0.996 > 0.989) for accuracy in *CNN/AE*, *LSTM/AE* thus these yields a higher band than the 3rd picks. In the other side; for 10 k-fold, in general, the 1st pick-CPU dataset is 50, 90 for epochs; 20, 30 batch sizes, yield the best values for accuracy; in addition, the 2nd pick-memory dataset are 70, 90 for epochs; 40, 30 batch sizes, yield the best values for accuracy; as well, in the 3rd pick-disk dataset is 60, 100 for epochs; 16, 16 batch sizes, yield the best values for accuracy in *CNN/AE*,

LSTM/AE; respectively. generally, the 1st, and 3rd pick is (0.998 > 0.997) for accuracy in *CNN/AE* thus this yields a higher band than the 2nd picks; the 2nd, and 3rd pick is (0.995) for accuracy in *LSTM/AE* thus these yields a higher band than the 1st pick.

As for classification models, nine picked numbers of *n_neighbore* parameter (1, 2, 3, 4, 5, 6, 7, 8 and 9) for *k-NN* model, and (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9) for *nu* parameter of *Oc-SVM* model have been used for tuning the hyperparameter of models. With regard to classification models, for split 70%, for the kpi dataset, in general, the 1st pick-CPU is 3 for *n_neighbore*; 0.1 *nu*, yield the best values for accuracy in *k-NN, Oc-SVM*; as well, the 2nd pick-memory is 2 for *n_neighbore*; 0.6 *nu*, yield the best values for accuracy in *k-NN, Oc-SVM*; in addition, and the 3rd pick-disk is 3 for *n_neighbore*; 0.1 *nu*, yield the best values for accuracy in *k-NN, Oc-SVM*, generally, the 1st pick is (0.90 > 0.62) for accuracy in *k-NN*, and *Oc-SVM* thus this yields a higher band than the 2nd and 3rd picks. In the other ways; for 10 k-fold, in general, the 1st pick-CPU is 3 for *n_neighbore*; 0.3 *nu*, yield the best values for accuracy in *k-NN, Oc-SVM*; as well, the 2nd pick-memory is 2 for *n_neighbore*; 0.1 *nu*, yield the best values for accuracy in *k-NN, Oc-SVM*; in addition, and the 3rd pick-disk is 2 for *n_neighbore*; 0.1 *nu*, yield the best values for accuracy in *k-NN, Oc-SVM*, generally, the 1st, and 2nd picks are (0.98 > 0.97) for accuracy in *k-NN*, thus these yields a higher band than the 3rd picks. the 2nd pick are (0.59) for accuracy in *Oc-SVM*, thus these yields a higher band than the 1st, and 3rd picks.

It is also worth noting that in this study, the effects of the dataset on the results were analyzed. According to the results gathered by using KPI dataset which contains three sets are; CPU, Memory, and Disk usage, (6300 rows) in total, observed over this study that one of the most important aspects of achieving an accurate detection, is taking into consideration the size of the dataset as big as possible, especially with a classification model. It has been realized that it is needed to employ a dataset that includes at least bigger than several hundred rows to get the best classification performance of accuracy. With regard to all classification models

and denoting to detect, the best average of CM values were obtained by the ten minutes of twenty-four-hour for twenty-one days dataset (300/6300 rows), For all datasets, the best results were achieved in the ranking model, as shown in Table 4.6, using *CNN/AE*, also in Table 4.4, using *LSTM/AE*, which obtained on the (6300 rows) of the dataset, by the usage of 10 k-fold data split, when (60 epoch, 16 batch size), for *CNN/AE*, giving (0.998) Accuracy rate. On the other hand by the usage of 10 k-fold data split, when (90-30), epoch, batch size, for LSTM/AE, giving (0.995) as Accuracy.

A comparison notes among the methods with hold-out split (train/test ratio), k-fold are provided with tuning hyperparameter through grid search, with each section for classification models, and deep learning models, in connection with the average Accuracy of built models to detect an anomaly, was expounded in the following sections.

4.3. Discussion of Classification Models, Split (Train % - Test %), K-Fold (K) for KPI-Dataset

As for classification methods for (KPI dataset) which is contain three main columns include CPU, Memory, and Disk values of device, the inclusive rating of the models in terms of their performance in classification execution are based on the accuracy, AUC, precision, and recall of Confusion Matrix (CM), with two diffirent split techniquse are; hold-out, and k-fold, are *Oc-SVM*, and *k-NN* methods which are explained below.

- **Hold-out (70% Train - 30% Test)**

In this section, when the values of parameters *n_neighbore* (1, 2, 3, 4, 5, 6, 7, 8 and 9), and *weights* ('uniform', 'distance') for *k-NN* model, and (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9) for *nu*, ('auto', 'scale') for *gamma*, ('rbf', 'linear', 'poly', 'sigmoid') for *kernel*, parameters of *Oc-SVM* model have been used for tuning the hyperparameter of methods by use grid search.

In Table 4.1 of the *Oc-SVM* method, noticed the best parameters values are; (0.1) for *nu*, ('auto') for *gamma*, ('rbf' ') for *kernel*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.620, 0.390, 0.600, and 0.610, respectively. On the other hand; noticed the best parameters values of the *k-NN* method, are; (3) for *n_neighbore*, and ('uniform') for *weights*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.900, 0.900, 0.990, and 0.990, respectively. All of explained results depended on grid search to classify the CPU data.

In Table 4.3 of the *Oc-SVM* method, noticed the best parameters values are; (0.6) for *nu*, ('auto') for *gamma*, ('rbf' ') for *kernel*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.320, 0.660, 0.350, and 0.340, respectively. On the other hand; noticed the best parameters values of the *k-NN* method, are; (2) for *n_neighbore*, and ('uniform') for *weights*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.900, 0.990, 0.990, and 0.990, respectively. All of explained results depended on grid search to classify the Memory data.

In Table 4.5 of the *Oc-SVM* method, noticed the best parameters values are; (0.1) for *nu*, ('auto') for *gamma*, ('rbf' ') for *kernel*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.570, 0.430, 0.560, and 0.570, respectively. On the other hand; noticed the best parameters values of the *k-NN* method, are; (3) for *n_neighbore*, and ('uniform') for *weights*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.900, 0.990, 1.000, and 1.000, respectively. All of explained results depended on grid search to classify the Disk data.

Furthermore, the best results for accuracy are; 0.900, when *n_neighbore* is (3), for *k-NN* method. Also; 0.62 when *nu* is (0.1), for *Oc-SVM* method. The worst results for accuracy is; 0.320 when *nu* is (0.6), for *Oc-SVM* method.

- **K-Fold (10)**

In this section, when the values of parameters *n_neighbore* (1, 2, 3, 4, 5, 6, 7, 8 and 9), and *weights* ('uniform', 'distance') for *k-NN* model, and (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9) for *nu*, ('auto', 'scale') for *gamma*, ('rbf', 'linear', 'poly', 'sigmoid') for *kernel*, parameters of *Oc-SVM* model have been used for tuning the hyperparameter of methods by use grid search.

In Table 4.2 of the *Oc-SVM* method, noticed the best parameters values are; (0.3) for *nu*, ('auto') for *gamma*, ('rbf' ') for *kernel*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.430, 0.560, 0.440, and 0.440, respectively. On the other hand; noticed the best parameters values of the *k-NN* method, are; (3) for *n_neighbore*, and ('uniform') for *weights*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.980, 0.990, 0.990, and 0.980, respectively. All of explained results depended on grid search to classify the CPU data.

In Table 4.4 of the *Oc-SVM* method, noticed the best parameters values are; (0.1) for *nu*, ('auto') for *gamma*, ('rbf' ') for *kernel*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.590, 0.430, 0.560, and 0.570, respectively. On the other hand; noticed the best parameters values of the *k-NN* method, are; (2) for *n_neighbore*, and ('uniform') for *weights*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.970, 0.999, 1.000, and 1.000, respectively. All of explained results depended on grid search to classify the Memory data.

In Table 4.6 of the *Oc-SVM* method, noticed the best parameters values are; (0.1) for *nu*, ('auto') for *gamma*, ('rbf' ') for *kernel*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.470, 0.510, 0.500, and 0.490, respectively. On the other hand; noticed the best parameters values of the *k-NN* method, are; (2) for *n_neighbore*, and ('uniform') for *weights*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.960, 0.999, 1.000, and 1.000, respectively. All of explained results depended on grid search to classify the Disk data.

Furthermore, the best results for accuracy are; 0.980, when *n_neighbore* is (3), for *k-NN* method. Also; 0.59 when *nu* is (0.1), for *Oc-SVM* method. The worst results for accuracy are; 0.960, when *n_neighbore* is (2), for *k-NN* method. Also; 0.430 when *nu* is (0.3), for *Oc-SVM* method.

4.4. Discussion on Deep Learning Models, Split (Train % - Test %), k-fold (K) for KPI-Dataset

As for Deep Learning methods for (KPI dataset) which is contain three main columns include CPU, Memory, and Disk values of device, the inclusive rating of the models in terms of their performance in classification execution are based on the accuracy, AUC, precision, and recall of Confusion Matrix (CM), with two diffirent split techniquse are; hold-out, and k-fold, are CNN/AE, and LSTM/AE methods which are explained below.

- **Hold-out (70% Train - 30% Test)**

In this section, when the values of parameters *epoch* (30, 50, 60, 70, 90, and 100), and *batch size* (16, 20, 30, 40, 60, and 70) for *CNN/AE*, and *LSTM/AE* models, have been used for tuning the hyperparameter of methods by use grid search. In Table 4.1 of the *CNN/AE* method, noticed the best parameters values are; (50) for *epoch*, (20) for *batch size*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.996, 0.998, 0.990, and 0.998, respectively. Moreover; noticed the best parameters values of the *LSTM/AE* method, are; (100) for *epoch*, and (60) for *batch size*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.989, 0.996, 0.971, and 0.996, respectively. All of explained results depended on grid search to classify the CPU data.

In Table 4.3 of the *CNN/AE* method, noticed the best parameters values are; (30) for *epoch*, (20) for *batch size*, while the accuracy, AUC, precision, and recall

values of the confusion matrix for the best parameters are; 0.996, 0.999, 0.989, and 1.000, respectively. Moreover; noticed the best parameters values of the *LSTM/AE* method, are; (60) for *epoch*, and (40) for *batch size*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.994, 0.999, 0.983, and 0.996, respectively. All of explained results depended on grid search to classify the Memory data.

In Table 4.5 of the *CNN/AE* method, noticed the best parameters values are; (100) for *epoch*, (30) for *batch size*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.989, 0.997, 0.968, and 1.000, respectively. Moreover; noticed the best parameters values of the *LSTM/AE* method, are; (100) for *epoch*, and (60) for *batch size*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.976, 0.996, 0.933, and 0.994, respectively. Last but not least, all of explained results depended on grid search to classify the Disk data.

Furthermore, the best results for accuracy are; 0.996, when *epoch*, *batch size* are (50, 20), (30, 20) for *CNN/AE* method. Also; 0.994 when *epoch*, *batch size* are (60, 40), for *LSTM/AE* method. The worst results for accuracy are; 0.989, when *epoch*, *batch size* are (100, 30), for *CNN/AE* method. Also; 0.976 when *epoch*, *batch size* are (100, 60), for *LSTM/AE* method.

- **K-Fold (10)**

In this section, when the values of parameters *epoch* are; (30, 50, 60, 70, 90, and 100), and *batch size* are; (16, 20, 30, 40, 60, and 70) for *CNN/AE*, and *LSTM/AE* methods, have been used for tuning the hyperparameter of methods by use grid search.

In Table 4.2 of the *CNN/AE* method, noticed the best parameters values are; (50) for *epoch*, (20) for *batch size*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.997, 0.997, 0.979, and 0.999, respectively. While; noticed the best parameters values of the *LSTM/AE*

method, are; (90) for *epoch*, and (30) for *batch size*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.992, 0.999, 0.992, and 0.998, respectively. Last but not least, all of explained results depended on grid search to classify the CPU data.

In Table 4.4 of the *CNN/AE* method, noticed the best parameters values are; (70) for *epoch*, (40) for *batch size*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.995, 0.999, 0.986, and 1.000, respectively. Moreover; noticed the best parameters values of the *LSTM/AE* method, are; (90) for *epoch*, and (30) for *batch size*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.995, 0.999, 0.986, and 0.998, respectively. Last but not least, all of explained results depended on grid search to classify the Memory data.

In Table 4.6 of the *CNN/AE* method, noticed the best parameters values are; (60) for *epoch*, (16) for *batch size*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.998, 0.999, 0.987, and 1.000, respectively. Moreover; noticed the best parameters values of the *LSTM/AE* method, are; (100) for *epoch*, and (16) for *batch size*, while the accuracy, AUC, precision, and recall values of the confusion matrix for the best parameters are; 0.995, 0.999, 0.996, and 1.000, respectively. Last but not least, all of explained results depended on grid search to classify the Disk data.

Furthermore, the best results for accuracy are; 0.998, when *epoch*, *batch size* are (60, 16), for *CNN/AE* method. Also; 0.995 when *epoch*, *batch size* are (90, 30), (100, 16), for *LSTM/AE* method. The worst results for accuracy are; 0.995, when *epoch*, *batch size* are (70, 40), for *CNN/AE* method. Also; 0.992 when *epoch*, *batch size* are (90, 30), for *LSTM/AE* method.

Of all 24 fields of 6 tables for four detection models, results show that the *CNN/AE* method is the better choice overall, yielding better results in 5 fields out of the total amount of 6 for test sets, while the *LSTM/AE* method yields more reliable results for 1 field out of the total amount of 6 for test sets, as a deep learning models.

On the other hand, the *k-NN* method, yielding better results in 6 fields out of the total amount of 6 for test sets, while the *Oc-SVM* method was not the best choice for any of the tables in anomaly detection models, out of the total amount of 6 for test sets, as a classification models.

Last but not least, through the confusion matrix readings for all methods we conclude that the confusion matrix is effective and fruitful with the classification, and deep learning models. So, to validate obtained results of models which are used in this study, and adjusting optimization to improve convergence incrementally over time, and determine the amount of data used for training, also choosing model parameters during design; A learning curve (machine learning curve or training curve) plots, were applied for classification, and deep learning models. Briefly, learning curve; is a tool to find out how much a machine model benefits from adding more training data and whether the estimator suffers more from a variance error or a bias error. In the machine learning domain, there are two implications of learning curves differing in the x-axis of the curves, with experience of the model graphed either as the number of training examples used for learning as used in this study with classification models and the number of iterations as used in this study with deep learning models; to training the model. Figures (4.1-4.24); explain the difference between the readings of *CNN/AE*, *LSTM/AE*, and *Oc-SVM*, *k-NN* accuracy rates, wherein, Figures (4.1 - 4.24); explain the number of training examples used for learning the classification models whereas the averages of accuracy are 0.998, 0.995, 0.990, and 0.220, respectively with regard to training, and testing rates the uppermost ability order for the methods, appears as *CNN /AE*, *LSTM/AE*, *k-NN*, and *Oc-SVM*, for difference epochs and patch size in deep learning models, and dataset size in classification models. These techniques helped in testing used models in our study and made area to increase the possibilities of obtaining higher accuracy.

4.5. Discussion of the Comparison with Previous Works

In this section, obtained results of developed models were discussed and compared with previous works to make sure that classification models, and deep learning models have parallel performance with previous works. The comparison was summarized in Table 4.7.

Table 4.7**.** Comparing the results of accuracy for various techniques.

Ser.	Methods	Performance Metric	Values
1#	**LSTM**, Ralf C. Staudemeyer, [2015]	ACC	93.82 %
2#	**DT(J48)**, Shailendra Sahu, et. al., [2015]	ACC	97.23 %
3#	**LSSVM-IDS**, Mohammed A. Ambusaidi, et. al., [2016]	ACC	97.33 %
4#	**RNN-IDS**, Chuanilong Yin, et. al., [2017]	ACC	97.09 %
5#	**Logitboost-RF**,TimWatson, et. al., [2017]	ACC	99.45 %
6#	**ODC**, Annie Gilda Roselin, et. al., [2021]	ACC	98.30 %
7#	**AE/ k-Means**, Annie Gilda Roselin,et. al., [2021]	ACC	82.93 %
8#	**HAST-NAD**, Guanglu Wei et.al.,[2021]	ACC	99.91 %
9#	**CNN/AE**	ACC	**99.98 %**
10#	**LSTM/AE**	ACC	**99.97 %**
11#	**k-NN**	ACC	**95.00 %**

As shown in the table, the ratio of the accuracy for Convolutional Neural Network with Auto Encoder **(CNN/AE)**, Long Short-Term Memory with Auto Encoder **(LSTM/AE)** models, are higher than; Long Short-Term Memory **(LSTM)**, Ralf C. Staudemeyer, [2015]; Desition Tree **(DT(J48))**, Shailendra Sahu, et. al., [2015]; Least Square Support Vector Machine-Intrusion Detection System

(LSSVM-IDS), Mohammed A. Ambusaidi, et. al., [2016]; Recurrent Neural Network- Intrusion Detection System **(RNN-IDS)**, Chuanilong Yin, et. al., [2017]; Logitboost- Random Forest **(Logitboost-RF)**, TimWatson, et. al., [2017]; Optimized Deep Clustering **(ODC)**, and Auto Encoder **(AE)** with Kmeans, Annie Gilda Roselin, [2021]; Hierarchical Spatiotemporal Feature Learning **(HAST-NAD)**, Guanglu Wei, [2021]; wherein the values of Accuracy **(ACC)** are; 99.98 %, 99.97 %, 99.70 %, 93.82 %, 97.23 %, 97.33 %, 97.09 %, 99.45 %, 98.30 %, 82.93 %, and 99.91%, respectively; which is consistent with the results of Guanglu Wei, et al. [2021]; indicating the models learn more features, and improve the accuracy rate.

The results confirm that the *CNN/AE*, and *LSTM/AE*, classier provide superior achievement to detect an anomaly and recognize the health of devices, when compared to results previously published results of strong static classiers in terms of Accuracy.

5. CONCLUSION

The main objective of this thesis is to employ accurate models for detecting anomaly and recognize the health of devices, by CPU, MEMORY, and DISK rates using Machine learning methods including LSTM/AE, CNN/AE, Oc-SVM, and k-NN to test KPI daily datasets. Two different detection approaches are created by the development of environment, which are the classification models, and deep learning models.

For the detection model (train-test%), and (k)fold that used CPU, Memory, and Disk datasets have been partitioned into two different ranges. In the first division, 70% of dataset content has been utilized for training while the rest of the dataset has been utilized for testing. Additionally, 10 of k-fold also used to train and test divisions have been utilized as second partitioning ranges for datasets.

The performance of all models were evaluated by calculating the value of the confusion matrix. A total of 24 different models were developed, including 12 classification models and 12 deep learning models.

According to the obtained results, the deep learning model produced better results compared to the classification models of all methods. In general, CNN/AE models show better performance than models developed by other methods. The order of the methods for the prediction model in terms of accuracy-based performance, from best to worst, is CNN/AE, LSTM/AE, k-NN, and Oc-SVM, while for the classification model, the order of favor of these methods is shown as k-NN , and Oc-SVM.

In general, when recognizing the health of the device, we deducting that the device is good during all the execution time, and also the memory has some of bozenıing. the order in terms of speed of the methods producing detection models is based on execution time, from fast to slow, CNN/AE, LSTM/AE, k-NN, and Oc-SVM.

As a suggestion for future work, a similar study can be conducted in several different areas by employing a hyperdeep learning neural networks methods or models to detect online attacks early with high accuracy.

REFERENCES

Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton (2017)." ImageNet Classification with Deep Convolutional Neural Networks", communıcatıons of the acm, june, vol. 60, no. 6.

Amit V Kachavimath , Shubhangeni Vijay Nazare , Sheetal S Akki (2020) “Distributed Denial of Service Attack Detection using Naïve Bayes and K-Nearest Neighbor for Network Forensics “ , IEEE Xplore Part Number: CFP20K58-ART; ISBN: 978-1-7281-4167-1.

Anastasia Khudoyarova et.al., **2021** " Using Machine Learning to Analyze Network Traffic Anomalies", 978-1-6654-0476-1/21/$31.00 c2021 IEEE.

Annie Gilda Roselin et.al., **2021** "Intelligent Anomaly Detection for Large Network Traffic With Optimized Deep Clustering (ODC) Algorithm", 10.1109/ACCESS.2021.3068172

Asri Ngadi, Hamid H. Jebur, Salima Omar(2013) .” Machine Learning Techniques for Anomaly Detection: An Overview”, International Journal of Computer Applications (0975 – 8887), Volume 79 – No.2, October 2013.

Bernhard Schölkopf, John C Platt, John C Shawe-Taylor, Alex J Smola, and Robert C Williamson (2001). " Estimating the support of a high-dimensional distribution ". Neural Computation, 13(7):1443–1471, Jul.

C. Huang, L. Zheng, S. Wang, V. C. M. Leung, T. Lin, and K. Peng, **2018** “Intrusion Detection System Based on Decision Tree over Big Data in Fog Environment,” Wireless Communications and Mobile Computing, vol. 2018, pp. 1–10.

Chafiq Titouna, Farid Naït-Abdesselam, Ashfaq Khokhar, **2019** "DODS: A Distributed Outlier Detection Scheme for Wireless Sensor Networks" Elsevier Computer Networks 161, pp. 93–101.

Charu C. Aggarwal, 2017. High-Dimensional Outlier Detection: The Subspace Method. From: C. Aggarwal. Outlier Analysis

Chen-Yu Lee, Patrick W. Gallagher, Zhuowen Tu (2016), " Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree", International Conference on Artificial Intelligence and Statistics (AISTATS) 2016, Cadiz, Spain. JMLR: W&CP volume 51. pp. 464–472.

D. M. Farid, N. Harbi, and M. Z. Rahman, 2010 “Combining naive bayes and decision tree for adaptive intrusion detection,” arXiv preprint arXiv:1005.4496.

Efrat Vilenski, Peter Bak, Jonathan D. Rosenblatt, **2019** "Multivariate anomaly detection for ensuring data quality of dendrometer sensor networks". Elsevier Computers and Electronics in Agriculture 162, pp. 412–421.

Evelyn Fix, Joseph Hodges L. (1951). " Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties ".USAF School of Aviation Medicine, Randolph Field, Texas. Archived (PDF) from the original on 26, September 2020.

Felix A. Gers, Jurgen Schmidhuber, Fred Cummins(2000)" Learning to Forget: Continual Prediction with LSTM". Neural Computation 12, 2451–2471 (2000) °c 2000 Massachusetts Institute of Technology.

Gideon Creech and Jiankun Hu, **2014**. “A semantic approach to host-based intrusion detection systems using contiguousand discontiguous system call patterns”. IEEE Transactions on Computers, 63(a):807-81e.

Gonzalo Marin, Pedro Casas, German Capdehourat (2019)." Deep in the Dark - Deep Learning-based Malware Traffic Detection without Expert Knowledge",2019 IEEE Security and Privacy Workshops (SPW).

Guanglu Wei et.al., **2021** "Adoption and realization of deep learning in network traffic anomaly detection device design". Soft computing (2021) 25:1147–1158

Guansong Pang, Chunhua Shen, Longbing Cao, and Anton van den Hengel. 2020. Deep Learning for Anomaly Detection: A Review. ACM Comput. Surv. 1, 1, Article 1 January, 36 pages. https://doi.org/10.1145/3439950

Guansong Pang, Longbing Cao, Charu Aggarwal, 2021. Deep learning for Anomaly Detection: Challenges, Methods and Opportunities. March, Pages 1127–1130https://doi.org/10.1145/3437963.3441659

H. Li, 2010 “Research and implementation of an anomaly detection model based on clustering analysis,” Proceedings - 2010 International Symposium on Intelligence Information Processing and Trusted Computing, IPTC 2010, pp. 458–462.

I. Goodfellow, Y. Bengio, A. Courville (2016). "Deep Learning". Cambridge, MA, USA: MIT Press, book.

J. Goh, S. Adepu, M. Tan, and Z. S. Lee, **2017** “Anomaly detection in cyber physical systems using recurrent neural networks,” Proceedings of IEEE International Symposium on High Assurance Systems Engineering, pp. 140–145.

J. Inoue, Y. Yamagata, Y. Chen, C. M. Poskitt, and J. Sun, **2017** “Anomaly detection for a water treatment system using unsupervised machine learning,” IEEE International Conference on Data Mining Workshops, ICDMW, vol. 2017-Novem, pp. 1058–1065.

Jonathan Masci, Ueli Meier, Dan Cire¸san, and J¨urgen Schmidhuber (2011), "Stacked convolutional auto-encoders for hierarchical feature extraction". Berlin, Germany: Springer, pp. 52–59. Available:http://www.springerlink.com/index/U47243T6702223K4.pdf.

Kurt Hornik, (1990), "Approximation capabilities of multilayer feedforward networks", Neural Netw., vol. 4, no. 2, pp. 251–257.

Larry M. Manevitz ,Malik Yousef (2001). " One-Class SVMs for Document Classification ". Journal of Machine Learning Research 2 139-154.

M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, **2016** “Building an intrusion detection system using a filter-based feature selection algorithm,” IEEE Transactions on Computers, vol. 65, no. 10, pp. 2986–2998.

M. Almiani, A. A. Ghazleh, A. Al-Rahayfeh, and A. Razaque, **2019** “Intelligent intrusion detection system using clustered self organized map”, 5th International Conference on Software Defined Systems, SDS 2019, no. 1, pp. 138–144.

M. H. Kamarudin, C. Maple, T. Watson, and N. S. Safa, **2017** “A logitboostbased algorithm for detecting known and unknown web attacks,” IEEE Access, vol. 5, pp. 26 190–26 200.

Mahmoud Said Elsayed, Nhien-An Le-Khac, Soumyabrata Dev, Anca Delia Jurcut(2020)." Network Anomaly Detection Using LSTM Based Autoencoder ". Q2SWinet 16–20, Alicante, Spain

N. B. Aissa and M. Guerroumi, **2015** “A genetic clustering technique for Anomaly-based Intrusion Detection Systems,” IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD 2015 - Proceedings, pp. 1–6.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, et. al., (2015)"ImageNet Large Scale Visual Recognition Challenge ", Int J Comput Vis (2015) 115:211–252 DOI 10.1007/s11263-015-0816-y,

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol (2010). "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion", J. Mach. Learn. Res., vol. 11, no. 12, pp. 3371–3408.

Quoc Phong Nguyen , Kar Wai Lim, Dinil Mon Divakaran, Kian Hsiang Low, Mun Choon Chan (2019). "GEE: A Gradient-based Explainable Variational Autoencoder for Network Anomaly Detection", IEEE Conference on Communications and Network Security (CNS), 978-1-5386-7117-7/19/$31.00 ©2019 IEEE.

R. M. Elbasiony, E. A. Sallam, T. E. Eltobely, and M. M. Fahmy, **2013** “A hybrid network intrusion detection framework based on random forests and

weighted k-means," Ain Shams Engineering Journal, vol. 4, no. 4, pp. 753–762.

R. S. Naoum, N. A. Abid, and Z. N. Al-Sultani, 2013 "An Enhanced Resilient Backpropagation Artificial Neural Network for Intrusion Detection System," IJCSNS International Journal of Computer Science and Network Security, vol. 13, no. 3, pp. 98–104. [Online]. Available: http://paper.ijcsns.org/07fn gbook/201203/20120302.pdf

Raghavendra Chalapathy, Aditya Krishna Menon, Sanjay Chawla (2019). " ANOMALY DETECTION USING ONE-CLASS NEURAL NETWORKS ".

Raghavendra Chalapathy, Sanjay Chawla, 2019. Deep Learning For Anomaly Detection: A Survey. arXiv:1901.03407v2 [cs.LG] 23 Jan

Ralf C. Staudemeyer, **2015**. "Applying long short-term memory recurrent neural networks to intrusion detection", SACJ No. 56.

REN-HUNG HWANG, Senior Member IEEE, MIN-CHUN PENG, CHIEN-WEI HUANG, PO-CHING LIN, VAN-LINH NGUYEN (2020)."An Unsupervised Deep Learning Model for Early Network Traffic Anomaly Detection "VOLUME 8.

Ritesh K. Malaiya, Donghwoon Kwon, Jinoh Kim, Sang C. Suh, Hyunjoo Kim, Ikkyun Kim (2018)," An Empirical Evaluation of Deep Learning for Network Anomaly Detection " International Conference on Computing, Networking and Communications (ICNC): Network Algorithms and Performance Evaluation.

S. Jose, D. Malathi, B. Reddy, and D. Jayaseeli, **2018** "A Survey on Anomaly Based Host Intrusion Detection System," Journal of Physics: Conference Series, vol. 1000, no. 1.

S. Sahu and B. M. Mehtre, **2015** "Network intrusion detection system using J48 Decision Tree," 2015 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2015, pp. 2023– 2026.

S.S. Keerthi and C.J. Lin (2003). " Asymptotic behaviors of support vector machines with Gaussian Kernel ". Neural Computation, vol. 15, no. 7, pp. 1667-1689.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, Yoshua Bengio (2011), "Contractive auto-encoders: Explicit invariance during feature extraction", Int. Conf. Mach. Learn. (ICML), pp.833–840.

Sara A. Althubiti , Eric Marcell Jones Jr, Kaushik Roy(2018)." LSTM for Anomaly-Based Network Intrusion Detection ".978-1-5386-7177-1/18, IEEE.

Sepp Hochreiter, Jurgen Schmidhuber(1997) "Long short-term memory" Neural computation, vol. 9, no. 8, pp. 1735–1780.

Sepp Hochreiter,(1998) "The vanishing gradient problem during learning recurrent neural nets and problem solutions". International Journal of Uncertainty, Fuzziness and Knowledge -Based Systems, vol. 6, no. 02, pp. 107–116.

Sheraz Naseer, Yasır Saleem, Shehzad Khalıd, Muhammad Khawar Bashır, Jıhun Han, Muhammad Munwar Iqbal, Kıjun Han (2018), "Enhanced Network Anomaly Detection Based on Deep Neural Networks", Digital Object Identifier 10.1109/ACCESS.2018.2863036, volume 6.

T. Shon, Y. Kim, C. Lee, and J. Moon, 2005 “A machine learning framework for network anomaly detection using svm and ga,” in Proceedings from the sixth annual IEEE SMC information assurance workshop. IEEE, pp. 176–183.

Taejoon Kim, Sang C. Suh, Hyunjoo Kim, Jonghyun Kim, Jinoh Kim (2018). " An Encoding Technique for CNN-based Network Anomaly Detection", 978-1-5386-5035-6/18 ©2018 IEEE.

V.N. Vapnik (1999). " An overview of statistical learning theory ". IEEE Trans. On Neural Networks, vol. 10, issue 5, pp. 988-999.

Wei Wang, Ming Zhu, Xuewen Zeng, Xiaozhou Ye, Yiqiang Sheng (2017)." Malware Traffic Classification Using Convolutional Neural Network for Representation Learning" , pp. 712–717, 978-1-5090-5124-3/17/$31.00 ©2017 IEEE.

Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, Fuad E. Alsaadi (2017). "A survey of deep neural network architectures and their applications",Neurocomputing 234 (2017), 11–26.

X. Zhao and W. Zhang, **2016** “An anomaly intrusion detection method based on improved K-means of cloud computing,” Proceedings – 2016 6th International Conference on Instrumentation and Measurement, Computer, Communication and Control, IMCCC 2016, no. 61272172, pp. 284–288.

Y. Chuan-long, Z. Yue-fei, F. Jin-long, and H. Xin-zheng, **2017** “A Deep Learning Approach for Intrusion Detection using Recurrent Neural Networks,” IEEE Access, vol. 3536, no. c, pp. 1–1. [Online]. Available: http://ieeexplore.ieee.org/document/8066291/

Yuhuai Wu , Saizheng Zhang , Ying Zhang, Yoshua Bengio, Ruslan Salakhutdinov (2016)." On Multiplicative Integration with Recurrent Neural Networks ", arXiv:1606.06630v2 [cs.LG].

Zhitang Chen, Ke He, Jian Li, Yanhui Geng (2017). "Seq2Img: A sequence-to-image based approach towards IP traffic classification using convolutional neural networks" in Proc. IEEE Int. Conf. Big Data (Big Data), pp. 1271–1276.

Zornitza Genova Prodanoff et. al., **2020** "Anomaly Detection in RFID Networks Using Bayesian Blocks and DBSCAN".

CURRICULUM VITAE

ROAA MOHAMMED, She started a bachelor of Software Engineering department, Iraq, and graduated in 2017. In 2019, She began studying the MSc program at the Department of Computer Engineering, Çukurova University, Adana, under the supervision of Prof. Dr. M.Fatih Akay, and graduated in 2022.