

# **AN-IOT NIDS USING ASYMMETRIC PARALLEL AUTO-ENCODER**

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE  
REQUIREMENTS FOR THE DEGREE OF  
**M.Sc COMPUTER SCIENCE**

**MALLARAPU VIJAY**

Reg.No: PCS052121

UNDER THE GUIDANCE OF  
**Dr. JAYASUDHA J S**



DEPARTMENT OF COMPUTER SCIENCE  
SCHOOL OF PHYSICAL SCIENCES  
CENTRAL UNIVERSITY OF KERALA  
TEJASWINI HILLS PERIYE, KASARAGOD - 671316,  
KERALA, INDIA JULY 2023




**Department of Computer Science, Central University of Kerala  
Thejaswini Hills, Periyar - 671316, Kasaragod**

### **CERTIFICATE**

This is to certify that the thesis entitled, "**NETWORK INTRUSION DETECTION SYSTEM FOR IOT USING ASYMMETRIC PARALLEL AUTO-ENCODER**" submitted by **MALLARAPU VIJAY (REG. NO: PCS052121)** in partial fulfillment of the requirements for the award of M.Sc in Computer Science at the Central University of Kerala, is an authentic work carried out by him under my supervision and guidance.

To the best of my knowledge, the matter embodied in the report has not been submitted to any other University Institute for the award of any degree.

DATE : 24/8/2023

  
**Dr. JAYASUDHA J S**  
Professor and Internal Guide  
Head Of The Department  
Department of Computer Science  
Central University of Kerala  
Kasaragod, Kerala - 671316



**Department of Computer Science, Central University of Kerala  
Thejaswini Hills, Periyar - 671316, Kasaragod**

### **CERTIFICATE**

This is to certify that the thesis entitled, "**NETWORK INTRUSION DETECTION SYSTEM FOR IOT USING ASYMMETRIC PARALLEL AUTO-ENCODER**" is a bonafide work carried out by **MALLARAPU VIJAY (REG. NO: PCS052121)** in partial fulfillment of the requirements for the award of M.Sc in Computer Science at the Central University of Kerala, Kasaragod, during the academic year **2021-2023**.

The work is satisfactory to award a Master's Degree in Computer Science.

**DATE :** 24/8/2023

**1. EXTERNAL EXAMINER:**

*Mallarapu Vijay*  
24/8/2023

**2. INTERNAL EXAMINER:**

*[Signature]*

*[Signature]*

**Dr. JAYASUDHA J S**

Professor

Head Of the Department

Department of Computer Science

Central University of Kerala

# DECLARATION

---

I, MALLARAPU VIJAY, Reg No: PCS052121, student of Fourth Semester M.Sc Computer Science, Central University of Kerala, do hereby declare that the report entitled, "NETWORK INTRUSION DETECTION SYSTEM FOR IOT USING ASYMMETRIC PARALLEL AUTO-ENCODER", submitted to the Department of Computer Science is an original record of studies and bonafide work carried out by me from March 2021 to July 2023.

DATE: 24/08/2023

PLACE: Periyar.

M. Vijay

MALLARAPU VIJAY

PCS052121

# ACKNOWLEDGEMENT

I would like to express my sincere gratitude to all those who contributed to the realization of this research endeavor. My journey from conceptualization to the creation of the Asymmetric Parallel Auto-Encoder (APAE) intrusion detection system for IoT networks has been both enlightening and enriching.

First and foremost, I extend my heartfelt appreciation to my advisor Dr.JAYASUDHA J S, Professor and Head of the Department of Computer Science, Central University Of Kerala, for their unwavering guidance, insightful suggestions, and relentless encouragement throughout the course of this research. Their expertise and mentorship played a pivotal role in shaping the direction of my work and elevating its quality.

I also extend my gratitude to Dr. Manohar Naik, Assistant Professor, Department of Computer Science, Central University Of Kerala, and scholars whose foundational work paved the way for the concepts and methodologies employed in this study. The academic community's contributions to the fields of intrusion detection, machine learning, and IoT security have been invaluable in informing our approach.

I also thank our Teachers, Staff, and Friends of the Department of Computer Science for sharing their knowledge and suggestions with me.

**MALLARAPU VIJAY**

**PCS052121**

## **Abstract**

With the rapid growth of Internet of Things (IoT) networks, ensuring the security and integrity of these interconnected devices has become paramount. In this context, we introduce a novel Intrusion Detection System (IDS) named Asymmetric Parallel Auto-Encoder (APAE) designed to effectively detect a wide range of attacks in IoT networks. The core of the APAE model resides in its lightweight yet powerful architecture, centered around an asymmetric parallel auto-encoder framework. The encoder component of APAE employs a unique structure comprising two parallel encoders, each composed of three consecutive layers of convolutional filters. The initial encoder focuses on capturing local features by utilizing standard convolutional layers and a positional attention module. Meanwhile, the second encoder is responsible for capturing long-range information through the application of dilated convolutional layers and a channel attention module. Unlike the encoder, the decoder in the APAE system exhibits distinct characteristics, consisting of eight successive transposed convolution layers. This architecture has been meticulously crafted to cater to real-time attack detection while maintaining a streamlined design. Remarkably, the proposed APAE model showcases exceptional generalization performance, even when trained with a minimal amount of training data. To assess the efficacy of APAE, extensive evaluations are conducted using three widely recognized public datasets, specifically the UNSW-NB15 dataset. The results highlight the system's ability to detect various types of attacks effectively while maintaining a suitable level of computational efficiency.

# Table of contents

<b>List of figures</b>	<b>ix</b>
<b>List of tables</b>	<b>x</b>
<b>1 INTRODUCTION</b>	<b>1</b>
1.1 Introduction . . . . .	1
1.2 Background and Motivation . . . . .	2
1.2.1 Convolutional neural networks . . . . .	2
1.2.2 Auto-encoder . . . . .	3
1.3 Evaluation and Validation . . . . .	4
1.4 Contributions and Significance . . . . .	4
<b>2 LITERATURE REVIEW</b>	<b>6</b>
2.1 Introduction . . . . .	6
2.2 Related works . . . . .	6
2.3 Summary . . . . .	8
<b>3 METHODOLOGY</b>	<b>10</b>
3.1 Introduction . . . . .	10
3.2 Data Collection and Data Augmentation . . . . .	10
3.3 Preprocessing . . . . .	11
3.3.1 Label Encoding . . . . .	11
3.3.2 Data Normalization . . . . .	11
3.4 Proposed approach . . . . .	11
3.4.1 1D feature vector versus 2D feature vector . . . . .	12
3.4.2 Dilated convolution . . . . .	13
3.4.3 Attention modules . . . . .	14
3.4.4 Asymmetric parallel auto-encoder . . . . .	16

3.5	Network intrusion detection . . . . .	19
3.5.1	Network intrusion detection Architecture . . . . .	19
3.5.2	Training and Testing the Model . . . . .	21
<b>4</b>	<b>RESULTS AND DISCUSSIONS</b>	<b>22</b>
4.1	DISCUSSIONS . . . . .	22
4.1.1	Introduction . . . . .	22
4.1.2	Implementation Details . . . . .	22
4.1.3	Implementation Details . . . . .	24
4.1.4	Drawback and solution . . . . .	24
4.2	Results . . . . .	25
4.2.1	Anomaly detection . . . . .	25
4.2.2	Multi-class classification . . . . .	26
4.3	Conclusion . . . . .	28
4.4	Screenshots of Best Results by APAE . . . . .	29
	<b>BIBLIOGRAPHY</b>	<b>30</b>



# List of figures

1.1	<i>A typical structure for an auto – encoder . . . . .</i>	4
3.1	<i>1Dfeature vectorversus 2Dfeaturevector . . . . .</i>	13
3.2	<i>Dilated convolution versus standard convolution . . . . .</i>	14
3.3	<i>Positional self – attention module . . . . .</i>	15
3.4	<i>Channel self – attention module . . . . .</i>	16
3.5	<i>Asymmetric parallel auto – encoder . . . . .</i>	18
3.6	<i>The proposed model : network intrusion detection using an APAE together with a classifier . . . . .</i>	20
4.1	<i>CONFUSION MATRIX OF IDAE and APAE ON UNSW – NB15 DATASET</i>	26
4.2	<i>MULTICLASS CONFUSION MATRIX OF IDAE and APAE ON UNSW – NB15 DATASET . . . . .</i>	28
4.3	<i>Screenshot of the accuracy On train dataAnd test data . . . . .</i>	29

## List of tables

4.1	<i>The results of anomaly detection for UNSW – NB15 dataset . . . . .</i>	25
4.2	<i>The results of anomaly detection for UNSW – NB15 dataset . . . . .</i>	27
4.3	Comparison of results, using UNSW-NB15 Dataset . . . . .	27

# Chapter 1

## INTRODUCTION

### 1.1 Introduction

In the real world of the Internet of Things (IoT), the rapid growth of interconnected devices has showed in unparalleled convenience and efficiency across a multitude of applications. However, this surge in connectivity has also engendered new avenues for cyber threats, necessitating robust security mechanisms to safeguard the integrity and confidentiality of data transmitted through IoT networks. One of the key challenges in this domain is the development of effective Intrusion Detection Systems (IDS) capable of identifying and preventing various forms of malicious activities within IoT environments.

Addressing this challenge, the research introduces a novel and intelligent network intrusion detection system named Asymmetric Parallel Auto-Encoder (APAE). This innovative system leverages the power of deep learning, specifically an auto-encoder architecture, to discern and mitigate attacks in IoT networks. The APAE system is specifically designed to handle the intricate and dynamic nature of IoT traffic while maintaining a lightweight and efficient structure that facilitates real-time attack detection.

- Novel use of 2D representation for the input vectors of intrusion detection neural network that, unlike popular 1D representations bring the individual parameters of input

vectors close together and allow the convolutional filters to extract discriminative features from minority classes.

- Novel use of dilated convolution filters that (unlike standard convolutional filters) allows the network to have good generalization performance on minority classes by aggressively extracting spatial information across the inputs with fewer layers.

- Novel asymmetric auto-encoder named APAE that provides unsupervised feature learning by combining channel and position attention modules with dilated and standard deep auto-encoders and achieves high classification accuracy while maintaining compact architecture with very few numbers of parameters.

## **1.2 Background and Motivation**

Traditional approaches to intrusion detection often struggle to accommodate the unique characteristics of IoT networks, including the diverse types of devices, resource limitations, and varied communication patterns. The authors of this paper recognize the need for sophisticated and adaptable techniques to safeguard IoT ecosystems. To address this, they propose an IDS grounded in the power of deep learning, aiming to harness the capabilities of auto-encoders to create an intelligent system that can discern both known and novel attacks.

### **1.2.1 Convolutional neural networks**

It's a type of deep learning model specifically designed for processing and analyzing visual data, such as images and videos. CNNs have proven to be highly effective in tasks like image classification, object detection, facial recognition, and more.

The design of CNNs is inspired by the human visual system. They use a hierarchical structure of layers that automatically learn and extract features from the data. The key architectural components of a CNN include:

**Convolutional Layers:** Convolutional layers consist of multiple filters (also known as kernels) that slide over the input image to perform convolution operations. These filters capture various features of the input image, such as edges, textures, and patterns. The outputs of these filters are called feature maps, which represent the learned features of the input.

**Pooling Layers:** Pooling layers reduce the spatial dimensions (width and height) of the feature maps while retaining their important features. Common pooling operations include max pooling and average pooling, which capture the most significant information within each region.

**Activation Functions:** Activation functions like ReLU (Rectified Linear Activation) introduce non-linearity to the network, allowing it to model complex relationships in the data.

**Fully Connected Layers:**

These layers connect the output from previous layers to the final output layer using densely connected neurons. They help in making the final predictions based on the learned features.

### 1.2.2 Auto-encoder

The Auto-Encoder (AE) is a powerful unsupervised neural network that can be used for dimensionality reduction. As depicted in Figure 1.1, the AE consists of two main parts - the encoder and the decoder. The encoder transforms input data into a lower-dimensional space, while the decoder expands it to reproduce the initial input data. During the training process, the AE learns to capture the most prominent features of the data distribution in the hidden layers, creating a compressed feature vector that provides a better representation of

the data points than the raw data itself. This feature vector can be used for classification purposes, making the AE an excellent tool for data analysis as shown in fig. 1.1.

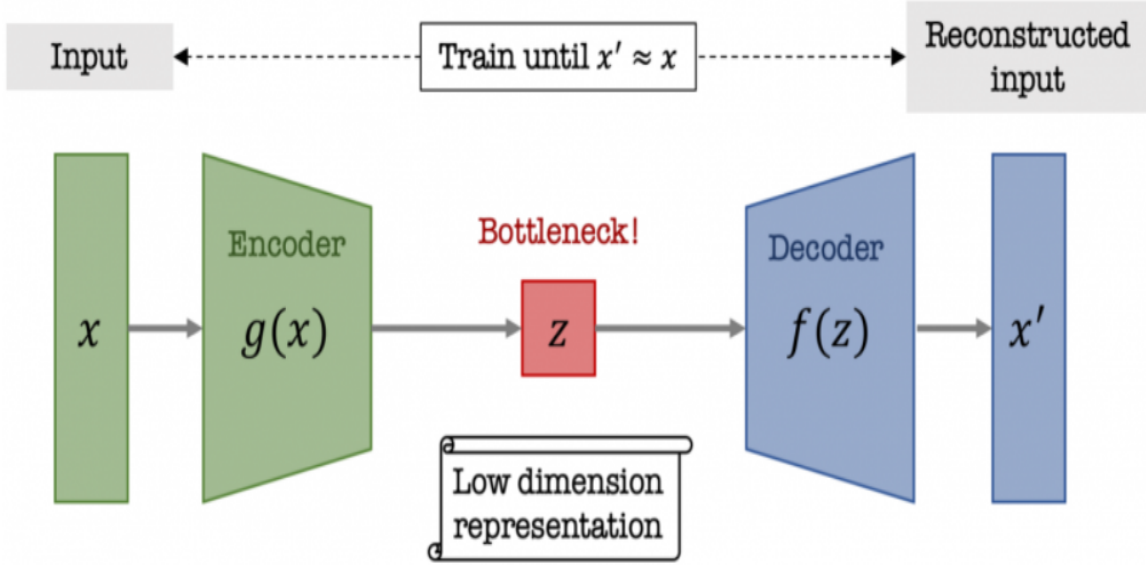


Fig. 1.1 A typical structure for an auto – encoder

### 1.3 Evaluation and Validation

The efficiency of this work is critically assessed through experimentation using the widely recognized UNSW-NB15 dataset. This dataset, which covers diverse attack scenarios, provides a robust foundation for evaluating the detection capabilities of the APAE system. By training the model with a limited number of training records, the authors showcase the system's ability to generalize effectively and detect various attacks with high accuracy.

### 1.4 Contributions and Significance

To sum up, deep learning techniques, particularly the APAE architecture, have a significant potential to enhance the security of IoT networks against various attacks. The

---

combination of local and global feature extraction, along with the capability to operate in real-time scenarios, make it possible to establish a more secure and robust IoT environment.

# **Chapter 2**

## **LITERATURE REVIEW**

### **2.1 Introduction**

An extensive literature review is important as it offers an up-to-date understanding of the subject matter and its significance. It gives an idea of the previous approaches used for similar studies. In this study, this chapter addresses some of the prior research in this area.

### **2.2 Related works**

The auto-encoder has been used in anomaly detection by many researchers [1] [2]. The idea is to use an AE that has been trained by some data representing the normal states of a system. Having such an AE, a new unseen state of the system can be classified as normal or anomaly as follows: feeding AE with the new state as input, the new state is classified as normal if the AE can accurately reconstruct it in its output. On the other hand, the new state is classified as an anomaly if the reconstruction error is high (above some threshold).

Another way of using auto-encoder in intrusion detection has also been used in some research [3]. Here, the idea is to use the auto-encoder as a dimensionality reduction tool. To



explain that, the AE can be divided into two main parts: encoder and decoder. In general, the encoder transforms the input data into a lower-dimensional space, and then, the decoder expands it to reproduce the initial input data. The training procedure forces the AE to capture the most prominent features of the data distribution as a lower-dimensional representation at the heart of the hidden layers. Ideally, this lower-dimensional feature vector provides a better representation of the data points than the raw data itself and can be used as a compressed feature vector for classification.

Using a lightweight neural network, reference [4] proposed an IDS for resource-constrained mobile devices. The model, which is called IMPACT, uses a stacked auto-encoder (SAE) to reduce the length of the feature vector, besides using an SVM classifier for the detection of only impersonation attacks.

Reference [5] presents a variational auto-encoder-based method for intrusion detection. This method is an anomaly detection approach that detects network attacks using reconstruction error of auto-encoder as an anomaly score. However, like any other anomaly detection method, this method is only capable of distinguishing between good and bad packets and cannot detect different attack types.

Reference [6] used deep learning together with shallow learning. It proposes to use a non-symmetric deep auto-encoder (NDAE) for non-symmetric data dimensionality reduction alongside a random forest for classification. The model has a light architecture and archives good classification accuracy on KDDCup99 and NSL-KDD datasets.

Reference [7] MSML is an intrusion detection framework with four modules: pure cluster extraction, pattern discovery, fine-grained classification, and model updating. The pure cluster module extracts all pure clusters using a hierarchical semi-supervised k-means algorithm. Pattern discovery identifies unknown patterns and labels test samples. The fine-grained classification module determines the class for unknown pattern samples, and

the model-updating module provides a mechanism for retraining. The KDDCup99 dataset evaluation shows that MSML provides good accuracy and F1-score.

An IoT ensemble IDS is presented in [8] that mitigates particular botnet attacks against DNS, HTTP, and MQTT protocols. In this work, new statistical flow features from the protocols are generated using the UNSW-NB15 dataset, and an AdaBoost learning method is developed using three machine-learning techniques: decision tree, naive Bayes, and artificial neural networks. The authors have evaluated their ensemble technique and showed that the generated features have the potential characteristics of malicious and normal activity using the correntropy and correlation coefficient measures.

Reference [9] uses a deep neural network named FCNet to detect attacks when only a few malicious samples are available for training. The method performs two operations: feature extraction and comparison. It learns a pair of feature maps from a pair of network traffic samples for classification and then compares the resulting feature maps to infer whether that pair of samples belong to a particular class or not.

## 2.3 Summary

It is an innovative Intrusion Detection System (IDS) named "APAE" that focuses on enhancing security within the Internet of Things (IoT) networks. This IDS is built upon the foundation of an asymmetric parallel autoencoder, enabling it to effectively identify a range of attacks within IoT environments. The architecture of the APAE system is carefully designed for real-time attack detection, making it suitable for the resource-constrained nature of IoT devices.

The encoder component of the APAE consists of two parallel encoders, each comprising three consecutive layers of convolutional filters. The first encoder focuses on extracting local features from the data using standard convolutional layers combined with a positional

attention module. The second encoder, on the other hand, captures long-range information through dilated convolutional layers and a channel attention module. This dual-encoder setup ensures that the system is capable of comprehensively analyzing the input data for both local and global characteristics associated with potential attacks.

In contrast to the encoder, the decoder segment of the APAE employs eight successive transposed convolution layers. This decoder is distinct from the encoder, allowing the model to reconstruct the input data and identify deviations from the expected reconstruction as potential intrusion events. To assess its efficacy, the APAE IDS is evaluated using the UNSW-NB15 dataset, which is a well-known benchmark for intrusion detection research. The results of these evaluations demonstrate the system's capability to accurately detect various types of attacks, further establishing its potential as a valuable tool for bolstering security in IoT networks. The APAE model represents an intelligent and effective solution for IoT network intrusion detection. Its innovative combination of asymmetric parallel autoencoders, lightweight architecture, and adaptability to limited training data showcases its potential to safeguard IoT systems against a range of cyber threats.

# Chapter 3

## METHODOLOGY

### 3.1 Introduction

This work utilizes a powerful combination of convolutional and attention mechanisms to extract and process features from input data. By using this technique to develop an autoencoder, we are able to effectively reconstruct input data while also capturing crucial features through the use of attention mechanisms. This architecture is particularly valuable for tasks that require feature extraction and generation from complex data, especially when it is crucial to capture both long-range and local dependencies.

### 3.2 Data Collection and Data Augmentation

To ensure compatibility with previous works, the UNSW-NB15 dataset's train and test split configuration[10] , consists of 1,75,341 records in the training set and 82,332 records in the testing set. This dataset contains nine types of attacks (Generic, Exploits, Fuzzers, DoS, Reconnaissance, Analysis, Backdoor, Shellcode, Worms) and Normal, with each record containing 43 features. Of these features (2 binary, 3 categorical, 37 numerical input attributes, and 1 class attribute).

## 3.3 Preprocessing

This dataset comprises network traffic data used for cybersecurity purposes. Preprocess data refers to the data that has undergone a series of cleaning, transformation, and normalizing steps to prepare it for use in a machine learning model. Preprocessing is a crucial step in the machine learning pipeline as it ensures that the data is in a suitable format for training and evaluation.

### 3.3.1 Label Encoding

In order to use certain parameters from network packets, such as protocol type, and other categorical features like service and flag they must be encoded into integer form. This is achieved through the use of "Label Encoding," where each categorical parameter is replaced with its corresponding integer index found in an array containing all unique values for that parameter.

### 3.3.2 Data Normalization

Normalization is often confused with standardization, but they have different effects on the data. Normalization scales data into a specific range, usually between 0 and 1, while standardization transforms data to have a mean of 0 and a standard deviation of 1.

## 3.4 Proposed approach

The primary objective of a Network Intrusion Detection System (NIDS) is to identify potential security breaches by monitoring network traffic data. Typically, NIDSs capture network traffic and use a one-dimensional feature vector extracted from network packets to determine whether to raise an intrusion alarm. This feature vector includes parameters such

as protocol type, service type, and the number of failed logins. Current public datasets like UNSW-NB15 follow this format. However, this approach creates an obstacle to achieving high classification accuracy with convolutional neural networks. Unlike voice signals, where the order of points is fixed, the order of individual parameters in the input vector of a CNN-based NIDS can significantly impact network complexity and classification accuracy. Despite this, current state-of-the-art NIDSs that rely on deep learning also use one-dimensional feature vectors as their network input.

If A and B are two individual parameters in the input vector of a CNN and their relation is correlated with the class of the input vector, it is crucial for A and B to be as close as possible for the CNN to be simple and accurate. This is because convolution filters can only extract information from neighboring parameters in the feature vector. If A and B are far apart in the input vector, larger filters or additional layers of filters are required to extract this information.

### 3.4.1 1D feature vector versus 2D feature vector

The use of a 2D feature vector allows for each parameter to have eight direct neighbors, as opposed to only two in a 1D feature vector. This means that a 2D filter can extract more neighborhood information, as demonstrated in Fig. 3.1. While it is possible to increase the size of a 1D filter or add more layers to the network to extract correlation information from distant parameters, this approach can lead to decreased accuracy and increased complexity, requiring more training data and time.

2D representation for feature vectors as network input. This allows for the extraction of relative information between x6 and x18 using a 2D filter of size 3x3, whereas a filter of size 19x13 would be necessary for the same task with the original 1D feature vector. To achieve this, feature vectors are transformed into their 2D representation with equal width

and height during the preprocessing phase, with the necessary number of zeros padded at the end as shown in fig. 3.1.

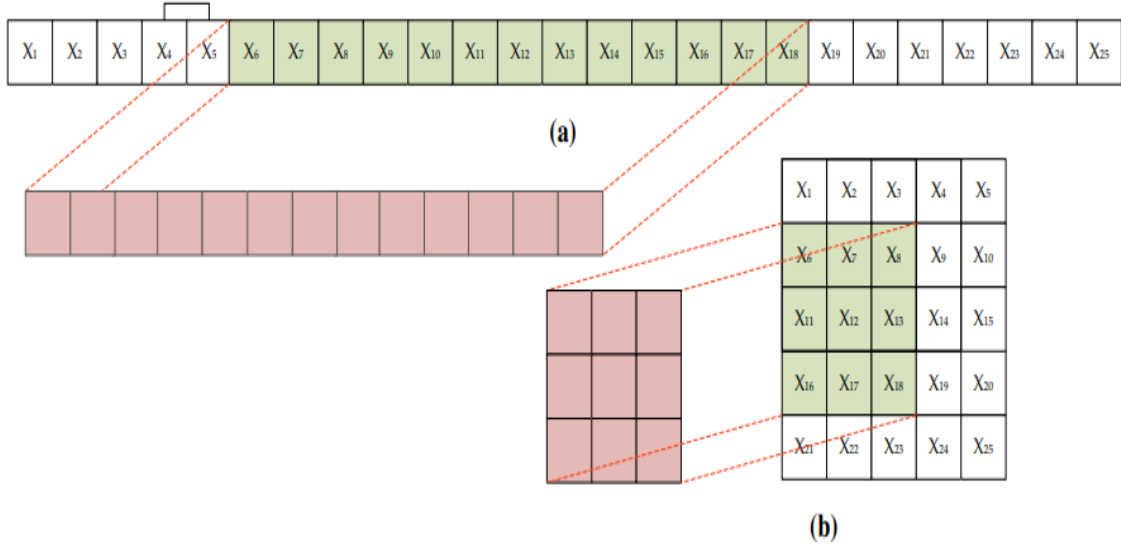


Fig. 3.1 1D feature vector versus 2D feature vector

### 3.4.2 Dilated convolution

Standard convolution filters can extract spatial information only from neighboring parameters. Therefore, to extract spatial information from distant parameters, larger filters or more successive layers of filters are required which makes the network more complex. To overcome this problem, dilated convolution can be used as a generalized form of convolution. It adds one hyper parameter to the convolution filter called dilation. Figure 3.2 shows the difference between standard convolution and dilated convolutions. As can be seen, the dilation parameter represents the space between neighboring parameters.

In other words, an L-dilated convolution filter considers cells with the distance of L as neighbors and extracts their relative spatial information. Therefore CNNs have a major advantage over traditional neural networks: unlike traditional neural networks that need human effort in preprocessing and feature design, CNNs need little preprocessing and

achieve independence from prior knowledge by learning the convolutional filters as part of network trainable parameters. A dilation factor of 1 corresponds to a standard convolution with no gaps. But when the dilation factor is increased to, say, 2 or 3, the filter "sees" a wider region of the input, which allows it to capture more context without increasing the number of parameters significantly as shown in fig. ??

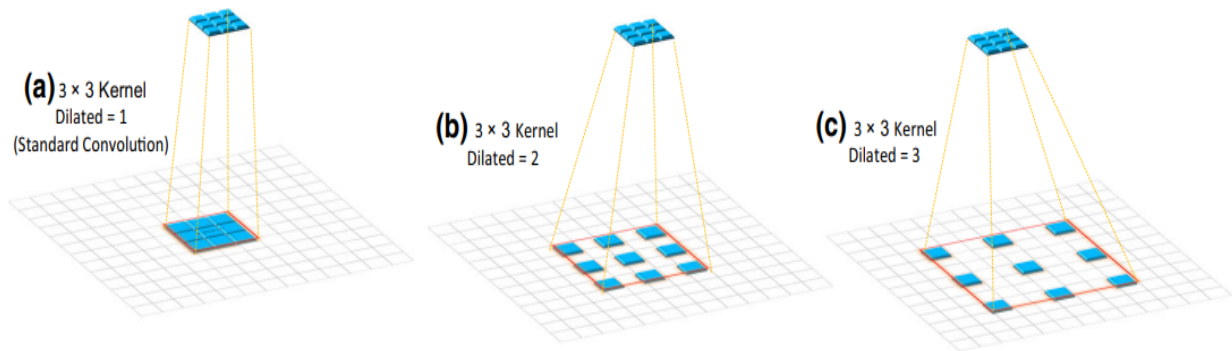


Fig. 3.2 *Dilated convolution versus standard convolution*

### 3.4.3 Attention modules

When performing convolution operations, it's important to note that the obtained features may differ for input vectors with the same receptive field[11]. These differences can create intra-class discrepancies and negatively impact classification accuracy. To address this issue, it's crucial to extract global contextual information from input feature vectors by building associations among long-range features. One effective approach is to use attention modules, such as the self-attention module, which can extract global contextual information from long-range input features. In fact, our proposed approach utilizes two types of self-attention modules, as explained in the following subsections.



### Positional self-attention module

A layer in a neural network called Positional Self-Attention. This layer is commonly used in models for understanding and processing sequences of data to enhance the representation capability of local features by encoding a wider range of contextual information into them.

When working with data, it's important to identify the significant words and their relationships. This can be achieved by utilizing a layer that enables the model to focus on the most relevant sections of the data - as illustrated in the accompanying fig. ??

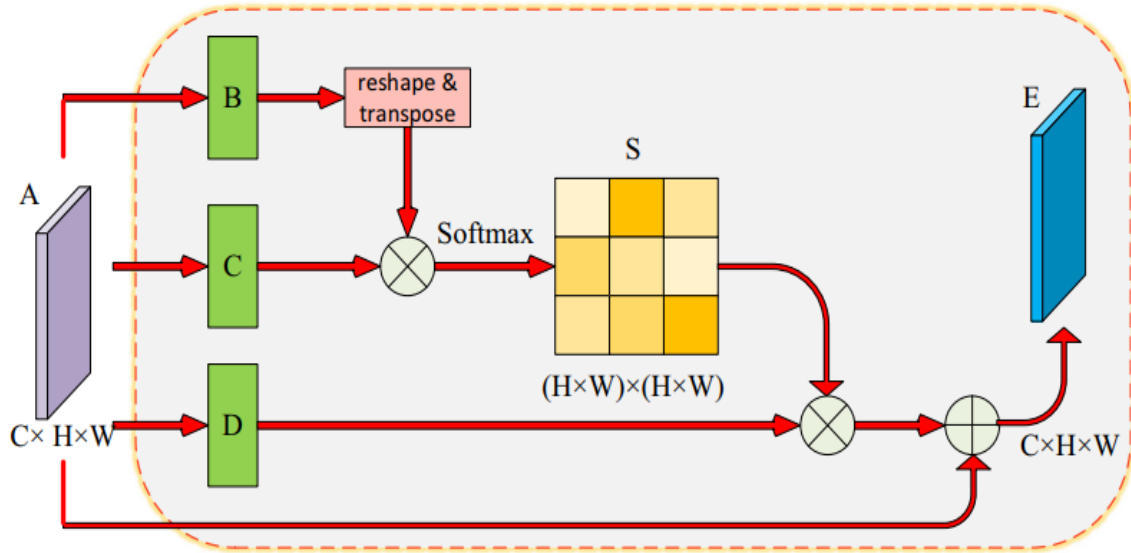


Fig. 3.3 *Positional self – attention module*

### Channel self-attention module

Channel Attention is a mechanism that helps the model focus on important channels (or feature maps) while filtering out less relevant information[12] . This can enhance feature representation and improve the model's performance on various tasks. The Channel Attention module is applied within the "Long-Range Feature Extractor with Dilated Convolution" part

of the architecture. This demonstrates how channel attention can be integrated into neural network layers to improve feature representation by focusing on relevant information while discarding noise. Figure 5 demonstrates the direct calculation of the channel attention map from the input features  $A$ . The process involves reshaping  $A$  to  $R$  power of  $C \times N$ , followed by obtaining  $X$  through the application of a softmax layer on the multiplication of  $A$ . This module's final value is a weighted sum of all channel and original features, which effectively models the long-range semantic dependencies between feature maps, as depicted in the fig. 3.4

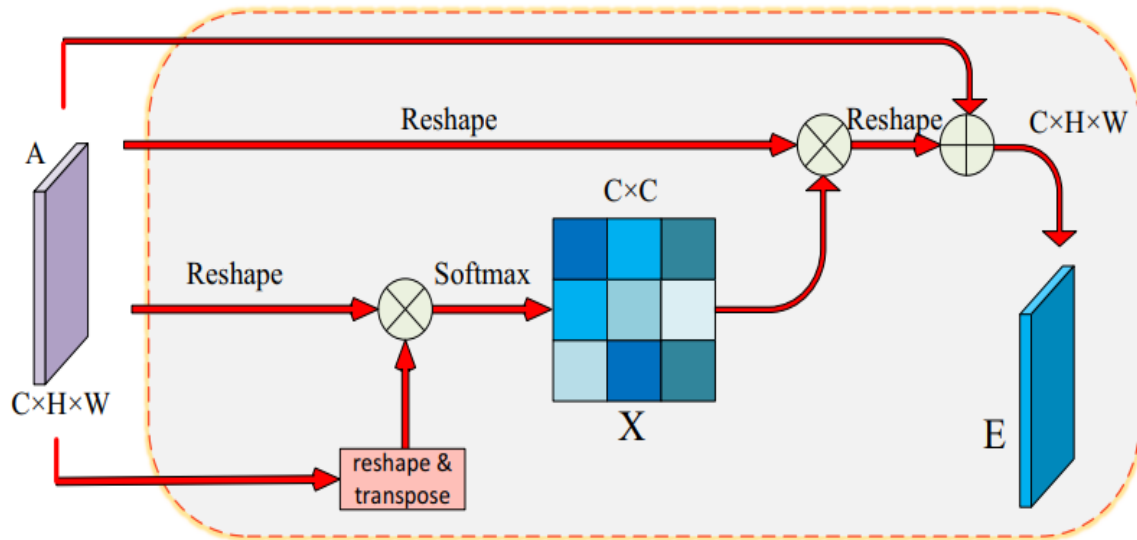


Fig. 3.4 *Channel self – attention module*

### 3.4.4 Asymmetric parallel auto-encoder

An autoencoder is a neural network architecture used for feature learning, data compression, and reconstruction. It consists of an encoder that transforms input data into a compressed representation and a decoder that reconstructs the original data from the compressed representation[13]. Autoencoders are widely used in various domains for dimensionality

reduction, anomaly detection, and feature extraction. A deep auto-encoder can be used to reduce the dimension of the input vector using the encoder-decoder paradigm described in Sect. 1.1. This way, one can extract the most descriptive features for classification and discard deceptive information in the input vector that misleads the classifier. However, 1D filter have their limitations in IDAE. They use 1D filters[14], and therefore, to achieve good accuracy in extracting discriminative features, multiple numbers of them should be stacked together. This results in complex deep networks that are not suitable for IoT devices. the encoder and decoder have an equal number of layers, and each layer in the decoder exactly does the reverse operations of its corresponding layer in the encoder. Therefore, the proposed approach uses an advanced autoencoder called asymmetric parallel auto-encoder (APAE). fig. 3.5 shows the overall structure of an APAE. As you can see, APAE is an asymmetric auto-encoder that combines two encoders and a decoder together. The input to APAE is the 2D representation of the input vector obtained by applying the preprocessing procedure described in Sect. 3.4.1. Generally, APAE consists of a input layer and three other major parts: encoder, latent feature, and decoder.

The encoder contains two feature extractors: a longrange feature extractor and a local feature extractor. The long-range feature extractor is the encoder part of an autoencoder with dilated convolutional filters of size  $3 \times 3$ . The local feature extractor is also the encoder part of an auto-encoder with standard  $3 \times 3$  convolutional filters. In each feature extractor, there are three layers of successive convolutional filters that each one (together with downsampling), extracts some lower dimension features from its input and feed them to the next layer. The final parts of the two feature extractors are also different. The local feature extractor has a positional self-attention module, and the long-range feature extractor has a channel self-attention module. Using a positional self-attention module as the last part of the local feature extractor is because the positional self-attention module can enhance the representation capability of local features, as explained in Sect. 3.4.3. The channel

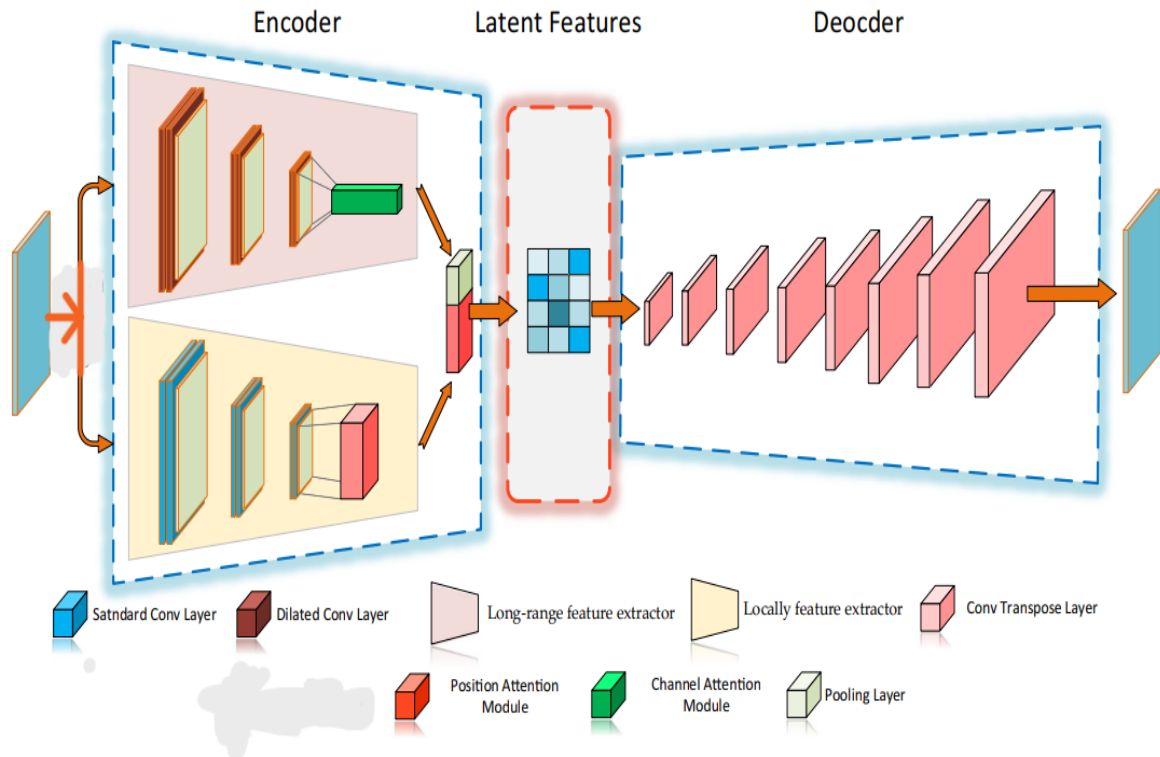


Fig. 3.5 *Asymmetric parallel auto – encoder*

self-attention module, as the last part of the longrange feature extractor, is to emphasize interdependent feature maps and to enhance the extracted long-range features, as described in Sect. 3.4.3. At the end of each feature extractor, there are compressed features with lower dimensions that together make the latent features that are used to feed the decoder. Like normal auto-encoders, the aim of training an APAE is to approximate the identity function. In addition, the latent features of a trained APAE can also be used as a compressed and lower dimension form of its inputs. However, the difference between APAE and normal autoencoders is that the APAE is more robust and can efficiently produce the lower dimension representation of its input with a smaller number of layers and parameters. In addition, APAE is more suitable for NIDs because it acts more aggressively in extracting the relative spatial information of its input parameters comparing to normal autoencoders.

Typically, an asymmetric parallel auto-encoder uses 1D filters and convolution layers (as which is implemented in IDAE ) together with pooling layers in the encoder to reduce the dimensionality of the inputs and obtain a lower-dimensional representation. The decoder part also uses upsampling layers (the reverse of pooling layer) together with convolutional layers to regenerate the original inputs using the output of the encoder. However, as explained in previous paragraphs, the APAE encoder generates the lowerdimensional representation of input data by combining the outputs of two parallel feature extractors that each one uses a different type of convolutional layers. Therefore, the APAE decoder should be able to reverse the operation of both dilated and standard convolution layers.

## **3.5 Network intrusion detection**

### **3.5.1 Network intrusion detection Architecture**

APAE is first trained on a dataset to estimate the identity function for the training data. Then, the final model (as shown in fig.3.6 is obtained by concatenating a fully connected classifier layer at the end of the first parts (transfer layer, encoder, and latent features) of the trained APAE. After that, the final model is trained again using the training data to obtain the classifier weights while fine-tuning the weights of the APAE encoder for accurate classification[15] . Note that, in deep learning research, the success of a model is dictated by its structural architecture. Existing deep learning approaches typically use 1-dimensional input vector and they also have sequential structures that use only standard convolution layers in sequence. Therefore, they need to have many layers to achieve good classification accuracy, and as a result, they require many processing resources, which make them not suitable for real-time attack detection in IoT networks.

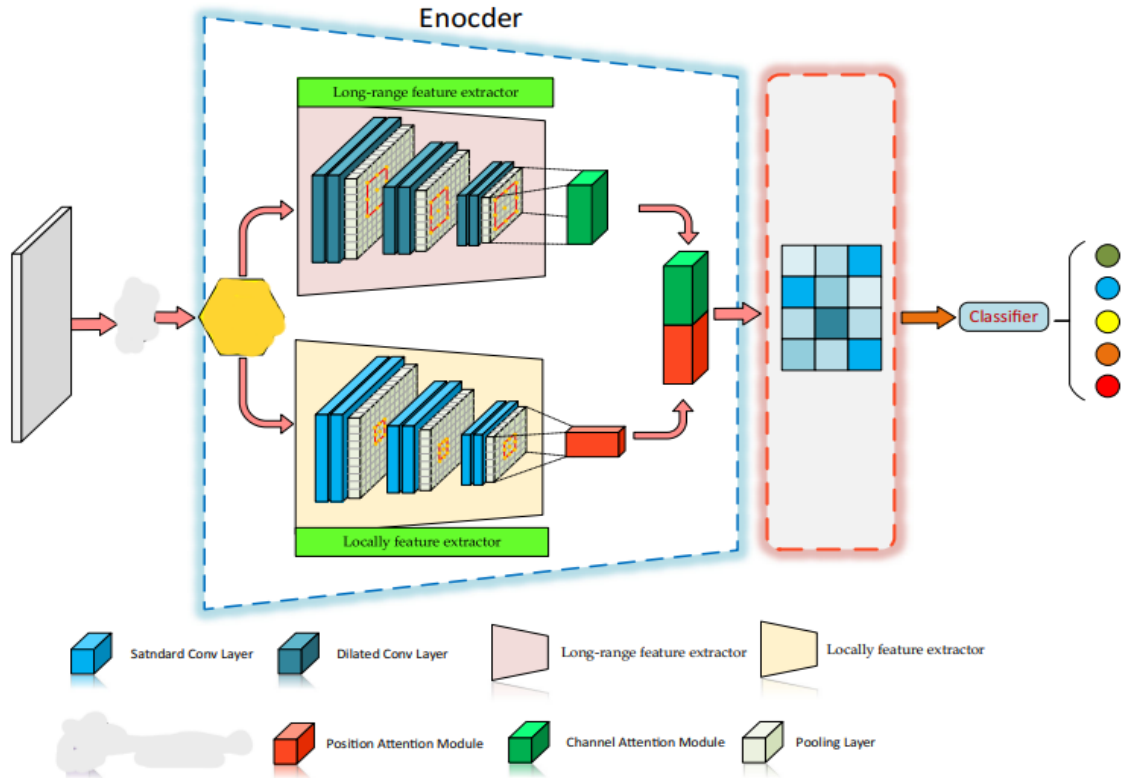


Fig. 3.6 The proposed model : network intrusion detection using an APAE together with a classifier

Our model uses parallel structure that have two feature extractors, each having few sequential layers with different types of convolutional filters. It also uses 2-dimensional input vector which brings the long-range features close together and allows the convolutional filters of the local feature extractor to extract more useful information. In addition, the long-range feature extractor uses dilated convolutional filters that can extract more information from long-range features. The proposed model also uses a positional selfattention module to enhance the representation capability of local feature extractor, and a channel self-attention module to enhance the extracted long-range features by the long-range feature extractor. Therefore, as the results of experiments in the next section show, the proposed model is highly superior comparing to the current state of the art IDAE as it has an efficient and lightweight architecture with very few parameters and also, it can accurately discern normal network traffic and different classes of attacks (even minority classes) from each other.

### 3.5.2 Training and Testing the Model

To evaluate this work, a feedforward neural network is utilized with the Sequential API due to its lightweight architecture that can detect in real-time, making it suitable for IoT networks. The architecture starts with an input layer (Input) that matches the input data's shape. To introduce non-linearity, multiple dense layers are sequentially added using Dense layers, where each layer is fully connected to the previous one. The model's design determines the number of neurons in each dense layer (e.g., 128, 64, 32) and the activation function (ReLU). For binary classification, the final dense layer has a single neuron with a sigmoid activation function, while for multi-class classification, the softmax activation function is used, and the units in the last layer change based on the types of attacks. The final dense layer outputs the probability of the positive class.

# Chapter 4

## RESULTS AND DISCUSSIONS

### 4.1 DISCUSSIONS

#### 4.1.1 Introduction

In the preceding sections, we explained the methodology used to achieve the study's objectives. This section will now discuss the results of the implemented models and approaches, providing a thorough understanding of the valuable insights gained through rigorous experimentation and evaluation. Specifically, we will analyze the classifier model's performance and its effectiveness in both binary and multi-class classification tasks.

#### 4.1.2 Implementation Details

To justly compare the efficiency and accuracy of the proposed model with other NIDSs, the following four criteria have been used:

**Overall classification accuracy:** Considering  $N$  as the total number of records in the test set and  $T$  as the total number of correct classifications made by a classifier, the overall accuracy of the classification can be calculated as the ratio between  $T$  and  $N$  multiplied by 100.



$$\text{accuracy} = \frac{T}{N} \times 100$$

**Confusion matrix:**

A confusion matrix is a squared table that allows visualization of the performance of a classification algorithm. It makes it easy to see if the algorithm confused between two classes and mislabeled each one as another. In this table, each row and column represent a class. The value of a cell at the index (i, j) shows the number of data instances that actually belong to the *i*th class, but the algorithm predicted them as members of the *j*th class. It is easy to visually inspect the prediction errors using this table as all the correct predictions are on the diagonal of the table, and the values outside the diagonal represent the prediction errors.

**Precision, recall, and F-Score:**

These criteria are useful for class-wise evaluation of the output of the classifier. To define these criteria, the following definitions are also needed for each class *c*:

- True positive (TP): number of records that are properly classified into class *c*.
- False positive (FP): number of records that are mistakenly classified into class *c*.
- False negative (FN): number of records from class *c* that are mistakenly classified into other classes.

Having previous definitions, the precision, recall, and F score for each class can be obtained using the following formulas:

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F - \text{score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The F score is the harmonic mean of the recall and the precision. The highest possible value of the F score is 1, which indicates perfect precision and recall, and the lowest possible value is 0, if either the precision or the recall is zero. In the case of classes with uneven distribution, the F score is a better criterion comparing to the accuracy.

### **4.1.3 Implementation Details**

The proposed model was trained using Python and the Tensorflow library, implementing the Adam optimizer with a batch size of 32 and 40 epochs on Google Colab infrastructure. This paper presents the results of experiments conducted on the public UNSW-NB15 dataset, comparing the proposed approach with existing works. Two types of experiments are conducted: the first used the proposed model as an anomaly detector (binary classification) on each dataset, while the second is used the proposed method to distinguish between different classes of attack (multiclass classification) on the UNSW-NB15 dataset. The following sections provide more detailed information on the results obtained.

### **4.1.4 Drawback and solution**

The Number of parameters: To effectively protect IoT networks, an NIDS(network intrusion detection system) must prioritize computational efficiency over perfect attack classification. Highly CPU-intensive NIDS options are unsuitable for IoT, as they cannot adequately process the limited capabilities of IoT devices. Instead, the best NIDS for IoT is one that achieves good classification results with low computational complexity. This work measures computational complexity using the number of parameters in deep learning methods. Which uses deep learning and is compared to other methods based on this criterion.

## 4.2 Results

### 4.2.1 Anomaly detection

This section presents the results of the anomaly detection experiment[16] . In this experiment, the records of all attack types are combined into a single attack class and each record of this dataset is labeled as either Normal or Attack. Section 4.2.1 shows the results of this experiment on the UNSW-NB15 dataset.

The results of anomaly detection in this work using the UNSW-NB15 dataset are presented in Table 1 in comparison to other algorithms.

Table 4.1 *The results of anomaly detection for UNSW – NB15 dataset*

<b>scores</b>	<b>IDAE</b>	<b>APAE</b>
Training Accuracy(%)	92.61	<b>99.80</b>
Testing Accuracy(%)	72.50	<b>97.90</b>
precision	92	<b>99.90</b>
recall	97.68	<b>99.78</b>
f1 score	93.80	<b>99.90</b>
parameters	6176	<b>3168</b>

The reason is that the binary classification on this dataset is easy. It is calculated the first and second principal components of this dataset and shows the level of intertwining between the two classes of this dataset is very low, which makes the binary classification of this dataset very easy. APAE is highly superior to IDAE in terms of efficiency and performance. The model has only 3168 parameters where as IDAE has 6176 parameters which shows about 50 and 55 percent decrease in network complexity comparing to IDAE respectively. This is very important as the proposed model can be used in devices with low processing power, and hence, it is better suits to IoT networks comparing to IDAE.

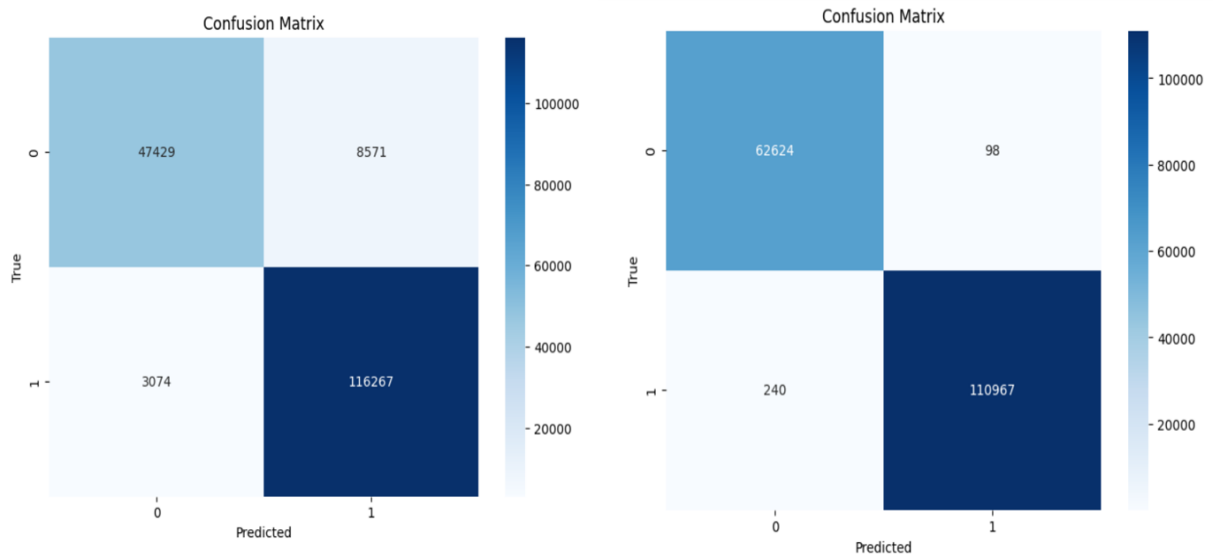


Fig. 4.1 *CONFUSION MATRIX OF IDAE and APAE ON UNSW – NB15 DATASET*

#### 4.2.2 Multi-class classification

This section presents the results of the multi-class classification experiment. In this experiment, the algorithms are compared based on their ability to predict the true classes of dataset records. This section displays the results of an experiment conducted on the UNSW-NB15 dataset. The dataset comprises ten classes, including Normal records and nine attack classes, such as Reconnaissance, Backdoor, Dos, Exploit, Analysis, Fuzzers, Worms, Shellcode, and Generic.

Table 4.2 shows the classification results for the 10-class attack detection experiment on the UNSW-NB15 dataset for the APAE against other algorithms. As you can see, the APAE defeats the IDAE. The total number of records in The UNSW-NB15 dataset has 44 records in the Worms class, compared to the Generic class with 18871 records. The IDAE performs poorly in most classes, including zero precision and recall for the Worms and Shellcode classes. The IDAE's precision and recall for the Exploits and Analysis classes are also nearly zero. However, the situation is better for the IDAE in the Exploits, Analysis, and Shellcodes classes. To improve classification, the over-sampling method is used to balance all attacks,

resulting in an equal number of samples for each attack type. This is a comparison of APAE and IDAE classification performance, with results. APAE has higher precision and recall for all classes compared to IDAE, which has worse results overall.

However, APAE has a significantly higher classification accuracy than IDAE, as shown in Table 4.2.

Table 4.2 *The results of anomaly detection for UNSW – NB15 dataset*

<b>Classes</b>	<b>IDAE</b>	<b>APAE</b>
Normal	83.02	<b>99.87</b>
Reconnaissance	40.42	<b>97</b>
Backdoor	65.23	<b>86</b>
DoS	62.21	<b>90</b>
Exploits	64.47	<b>88</b>
Analysis	54.45	<b>96</b>
Fuzzers	61.32	<b>97</b>
Worms	89.58	<b>99.73</b>
Shellcode	75.33	<b>85</b>
Generic	84.72	<b>97</b>

Table 4.3 Comparison of results, using UNSW-NB15 Dataset

<b>Classes</b>	<b>IDAE</b>		<b>APAE</b>	
	<b>Precision</b>	<b>Recall</b>	<b>Precision</b>	<b>Recall</b>
Normal	100	83.02	<b>100</b>	<b>99</b>
Reconnaissance	100	40.41	<b>100</b>	<b>97</b>
Backdoor	100	65.22	<b>100</b>	<b>86.40</b>
DoS	100	62.20	<b>100</b>	<b>90.10</b>
Exploits	100	64.46	<b>100</b>	<b>88.10</b>
Analysis	100	54.45	<b>100</b>	<b>96.20</b>
Fuzzers	100	61.31	<b>100</b>	<b>97</b>
Worms	100	75.32	<b>100</b>	<b>99</b>
Shellcode	100	89.58	<b>100</b>	<b>97</b>
Generic	100	84.71	<b>100</b>	<b>97</b>

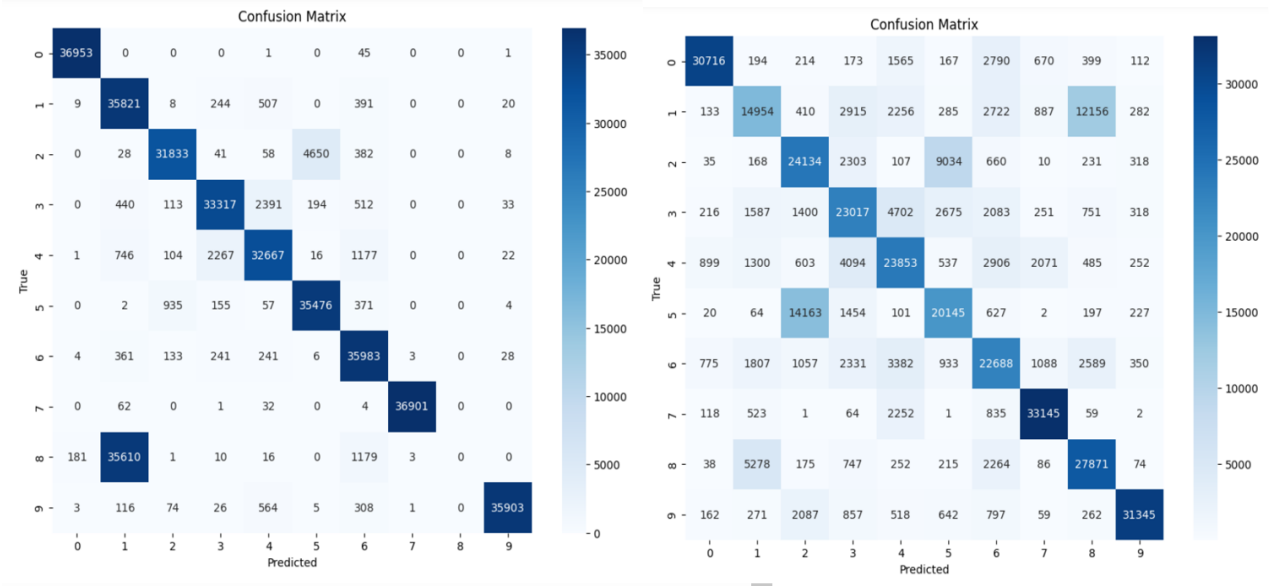


Fig. 4.2 *MULTICLASS CONFUSION MATRIX OF IDAE and APAE ON UNSW – NB15 DATASET*

### 4.3 Conclusion

A novel autoencoder architecture based on asymmetric parallel auto-encoder (APAE) is proposed. It uses dilated and standard convolutional filters to extract both local and long-range information from individual values in the feature vector. The architecture also uses a positional self-attention and a channel self-attention module to enhance the local and long-range features. This separation of features allows the APAE to accurately detect attacks, even in minority classes, with a small and lightweight deep neural network that is suitable for IoT devices with low processing capabilities. The proposed neural network is evaluated using UNSW-NB15. The efficiency of these systems is also very important, particularly in the IoT world in which the hardware devices have very limited processing capabilities. The experimental results demonstrate its promise for improving feature representation and reconstruction quality. This contributes to the field of neural architecture design by introducing advanced components into the architecture that pave the way for more sophisticated and accurate models in various data analysis tasks.

## 4.4 Screenshots of Best Results by APAE

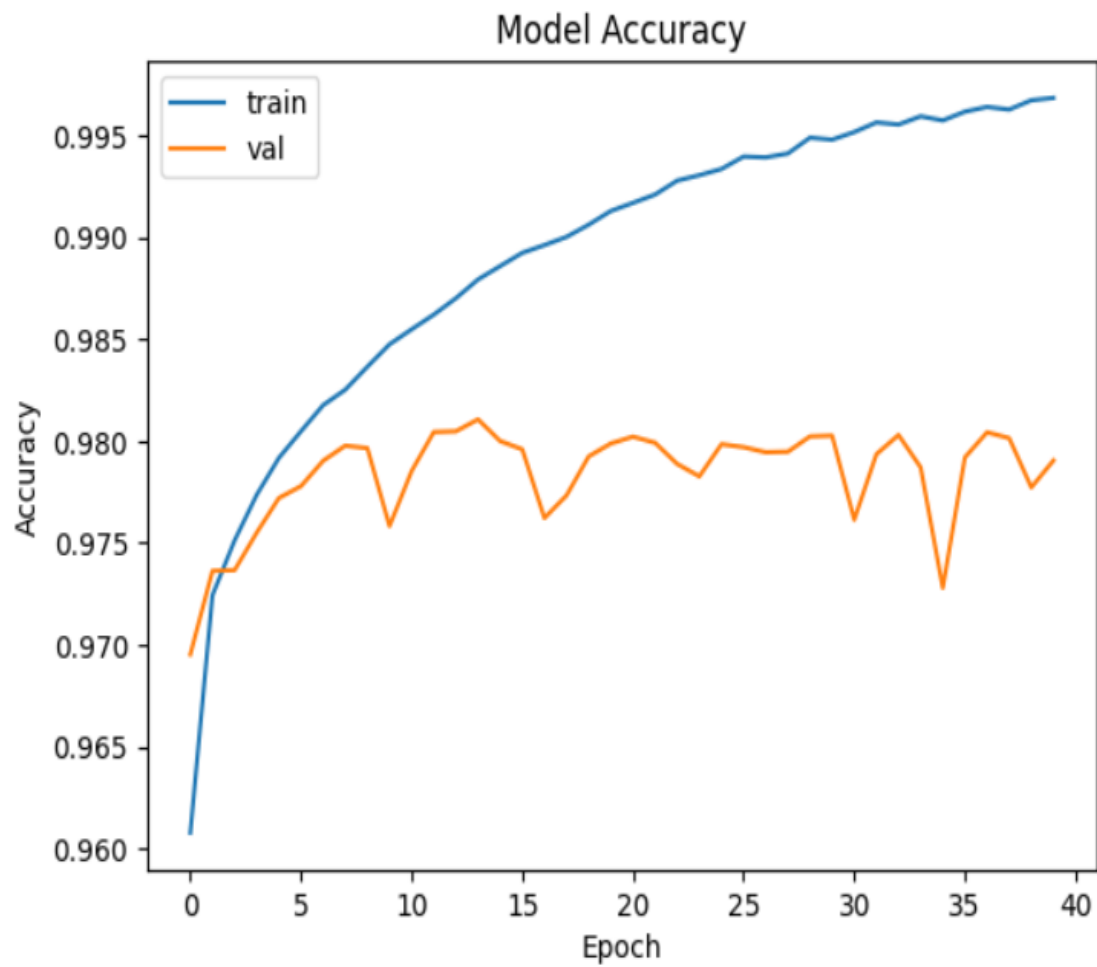


Fig. 4.3 Screenshot of the accuracy On train dataAnd test data

# BIBLIOGRAPHY

## References

- [1] H. D. et al, “Intrusion detection based on stacked autoencoder for connected healthcare systems,” . *IEEE Netw* 33, vol. 6, p. :64–69, 2019.
- [2] T. T. Van NT, “Temporal features learning using autoencoder for anomaly detection in network traffic. international conference on green technology and sustainable development,” 2020.
- [3] V. S. S. A. Ramamurthy M, Robinson YH, “Auto encoder based dimensionality reduction and classification using convolutional neural networks for hyperspectral images.” *Microprocess Microsyst*, vol. 79, p. 103280, 2020.
- [4] L. S. et al, “Impersonation attack detection via edge computing using deep autoencoder and feature abstraction.” *IEEE Access*, vol. 8, p. 65520–65529, 2018.
- [5] I. M. Zavrak S, “Anomaly-based intrusion detection from network flow features using a variational autoencoder.” *IEEE Access*, vol. 8, p. 65520–65529, 2020.
- [6] P. V. S. Q. Shone N, Ngoc TN, “A deep learning approach to network intrusion detection,” *IEEE Trans Emerg Topics Comput Intel*, vol. 2(1), p. :41–50, 2018.



- [7] Z. P. L. M. L. Y. Yao H, Fu D, “Msml: a novel multilevel semi- supervised machine learning framework for intrusion detection system,” *IEEE Int Things J*, vol. 6(2), p. 1949–1959, 2019.
- [8] C. K. Moustafa N, Turnbull B, “An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things,” *IEEE Int Things*, 2019.
- [9] D. X. Xu C, Shen J, “A method of few-shot network intrusion detection based on meta-learning framework,” *IEEE Trans Inf Forensics Secur*, vol. 15, p. 3540–3552, 2020.
- [10] S. J. Moustafa N, “Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data se,” *2015 Military commu- nications and information systems conference (MilCIS)*,, p. 1–1, 11 2015.
- [11] V. A. et al., “Attention is all you need. in: Presented at the 31st interna- tional conference on neural information processing systems,” *IEEE Trans Inf Forensics Secur*, 2017.
- [12] F. J. et al., “Dual attention network for scene segmentation.” *EEE/CVF conference on computer vision and pattern recognition (CVPR)*,, vol. 15–20, p. 3141–3149, 2019.
- [13] G. Q. Y. Z. Sun Y, Mao H, “Learning a good representation with unsymmetri- cal auto-encoder.” *Neural Comput Appl*, vol. 27(5), p. 3540–3552, 2016.
- [14] “Ndae source code: <https://github.com/ngoctn-lqdtu/a-deeplearning-approach-to-network-intrusion-detection>.”
- [15] N. A. S. A. Injadat M, Moubayed A, “Multi-stage optimized machine learning frame- work for network intrusion detection.” *IEEE Trans Netw Serv Manag 1–1*, 2020.

- 
- [16] S. B. M. M. V. S. V. D. H. A. Gong LLD, Le V, “Memorizing normality to detect anomaly: memory-augmented deep autoencoder for unsuper-vised anomaly detection,” *IEEE/CVF international conference on computer vision (ICCV)*, p. 1705–1714, 11 2019.