

**ANOMALY DETECTION WITH ATTENTION-BASED DEEP AUTOENCODER**

**BY**

Yong Hong Long

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

**BACHELOR OF COMPUTER SCIENCE (HONOURS)**

Faculty of Information and Communication Technology

(Kampar Campus)

MAY 2022

## REPORT STATUS DECLARATION FORM

**Title:** Anomaly Detection with Attention-based Deep Autoencoder

\_\_\_\_\_  
\_\_\_\_\_

**Academic Session:** MAY 2022

I YONG HONG LONG  
(CAPITAL LETTER)

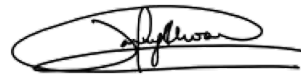
declare that I allow this Final Year Project Report to be kept in  
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.



(Author's signature)

Verified by,



(Supervisor's signature)

**Address:**

19, Jalan Anggerik 21,  
Taman Johor Jaya,  
81100, Johor Bahru, Johor

Tan Hung Khoo

Supervisor's name

**Date:** 9 September 2022

**Date:** 12/9/2022

<b>Universiti Tunku Abdul Rahman</b>			
Form Title : <b>Sample of Submission Sheet for FYP/Dissertation/Thesis</b>			
Form Number: <b>FM-IAD-004</b>	Rev No.: <b>0</b>	Effective Date: <b>21 JUNE 2011</b>	Page No.: <b>1 of 1</b>

**FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY**

**UNIVERSITI TUNKU ABDUL RAHMAN**

Date: 9 September 2022

**SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS**

It is hereby certified that Yong Hong Long (ID No: 19ACB05614) has completed this final year project entitled “Anomaly Detection with Attention-based Deep Autoencoder” under the supervision of Ts Dr Tan Hung Khoon (Supervisor) from the Department of Computer Science, Faculty of Information and Communication Technology, and Prof. Dr Leung Kar Hang (Moderator) from the Department of Computer Science, Faculty of Information and Communication Technology.

I understand that University will upload softcopy of my final year project in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

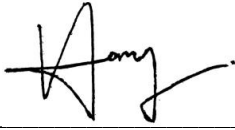


---

(*Yong Hong Long*)

## DECLARATION OF ORIGINALITY

I declare that this report entitled “**ANOMALY DETECTION WITH ATTENTION-BASED DEEP AUTOENCODER**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature :  \_\_\_\_\_

Name : Yong Hong Long

Date : 9 September 2022

## **ACKNOWLEDGEMENTS**

I would like to express my sincere thanks and appreciation to my supervisor, Dr. Tan Hung Khoon who has given me this bright opportunity to engage in an anomaly detection project. It is my first step to establish a career in the Artificial Intelligence field. A million thanks to you.

To a very special person in my life, Chong, W.Y., for her patience, unconditional support, and love, and for standing by my side during hard times. Finally, I must say thanks to my parents and my family for their love, support, and continuous encouragement throughout the course.

# ABSTRACT

Anomaly detection has become one of the most trending topics in the Information Technology domain. There are many existing approaches that deeply investigate the application of anomaly detection in several domains, such as video surveillance, financial technology, telecommunication, and healthcare. However, to the best of our knowledge, there is currently no absolute best solution that is reliable to be deployed on real-world applications. We investigate the key observation from the real-world application and tend to improve the existing anomaly detection model to a certain that we may find it is practical and reliable for real-world applications. We have found several key observations that might help to improve the existing work of MemAE from [4]. Firstly, anomaly detection on surveillance cameras will always process frames from the same scene. Secondly, it is not practical for an anomaly detection model to be trained with a huge amount of data in actual deployment. Thirdly, the anomalies that happen in a single video frame often occupy only a small portion of the video frame instead of the whole frame.

In this project, a Conv2D autoencoder was built from scratch that mimics the Conv3D autoencoder to process images. Two attention mechanisms were applied to the baseline Conv2D autoencoder separately and thus forms two different attention-based deep autoencoders. The two attention mechanisms are Convolutional Block Attention Module (CBAM) [10], and an attention-based approach proposed by [11]. Throughout the experiments, improvement can be seen by implementing the attention mechanisms onto the baseline autoencoder and so the accuracy of anomaly detection

# TABLE OF CONTENTS

<b>TITLE PAGE</b>	<b>i</b>
<b>REPORT STATUS DECLARATION FORM</b>	<b>ii</b>
<b>FYP THESIS SUBMISSION FORM</b>	<b>iii</b>
<b>DECLARATION OF ORIGINALITY</b>	<b>iv</b>
<b>ACKNOWLEDGEMENTS</b>	<b>v</b>
<b>ABSTRACT</b>	<b>vi</b>
<b>TABLE OF CONTENTS</b>	<b>vii</b>
<b>LIST OF FIGURES</b>	<b>ix</b>
<b>LIST OF TABLES</b>	<b>x</b>
<b>LIST OF SYMBOLS</b>	<b>xi</b>
<b>LIST OF ABBREVIATIONS</b>	<b>xii</b>
<b>CHAPTER 1 INTRODUCTION</b>	<b>1</b>
1.1 Anomaly Detection	1
1.2 Problem Statement and Motivation	3
1.3 Project Scope and Objectives	4
1.3.1 Project Scope	4
1.3.2 Project Objectives	4
<b>CHAPTER 2 LITERATURE REVIEW</b>	<b>6</b>
2.1 Memorizing Normality to Detect Anomaly: Memory-augmented Deep Autoencoder for Unsupervised Anomaly Detection [4]	6
2.1.1 Memory-augmented Autoencoder	6
2.1.1 Memory Module in MemAE	7
2.1.2 Limitation and Proposed Solutions	8
2.2 Few-Shot Scene-Adaptive Anomaly Detection [7]	9
2.2.1 Problem Setup	9
2.2.2 Meta-learning Framework	10
2.3 Convolutional Block Attention Module [10]	12
2.4 Learn to Pay Attention [11]	13

<b>CHAPTER 3 PROPOSED APPROACH</b>	<b>15</b>
3.1 System Requirements	15
3.2 System Design	15
3.2.1 General Flow of the Proposed Model	15
3.2.2 Attention Mechanism	16
3.3 Model Architecture	17
3.3.1 Baseline Deep Autoencoder	17
3.3.2 CBAM-based Deep Autoencoder	18
3.3.3 Attention-based Deep Autoencoder	19
<b>CHAPTER 4 EXPERIMENTS</b>	<b>20</b>
4.1 Experiment Settings	20
4.1.1 Dataset	20
4.1.2 Training and Evaluation Setup	20
4.2 Experiments with Different Model Settings	22
4.3 Anomaly Detection Performance Analysis	24
<b>CHAPTER 5 CONCLUSION</b>	<b>26</b>
<b>REFERENCES</b>	<b>27</b>
<b>APPENDIX</b>	<b>A-1</b>
<b>WEEKLY LOG</b>	<b>A-7</b>
<b>POSTER</b>	<b>A-10</b>
<b>PLAGIARISM CHECK RESULT</b>	<b>A-11</b>
<b>FYP2 CHECKLIST</b>	<b>A-18</b>



## LIST OF FIGURES

<b>Figure Number</b>	<b>Title</b>	<b>Page</b>
Figure 1-1	Anomaly where the data pattern does not conform with expected behaviour	1
Figure 1-2	Anomalies in a two-dimensional dataset where the region with the minority of data points are anomalies	2
Figure 1-3	Anomaly locality in a household burglary video frame	3
Figure 2-1	An overview of the Memory-augmented Autoencoder	6
Figure 2-2	An overview of the Model-Agnostic Meta-Learning algorithm for Scene Adaptive Anomaly Detection	9
Figure 2-3	Summary of the meta-training phase [7]	11
Figure 2-4	The overall view of Convolutional Block Attention Module [10]	12
Figure 2-5	Channel Attention Module [10]	12
Figure 2-6	Spatial Attention Module [11]	13
Figure 2-7	The overview of the attention mechanism introduced by [11]	13
Figure 3-1	Block diagram of proposed model	15
Figure 3-2	Illustration of the proposed deep 2D convolutional autoencoder	17
Figure 3-3	Implementation of CBAM on the encoder of the model	18
Figure 3-4	Implementation of [11] proposed attention mechanism on the encoder of the model	19
Figure 4-1	MNIST dataset class distribution (0 ~ 9) [1]	20
Figure 4-2	The proposed concept of calculating the buffer value for anomaly detection threshold	22

## LIST OF TABLES

<b>Table Number</b>	<b>Title</b>	<b>Page</b>
Table 4-1	Experiment metrics with MNIST dataset	22
Table 4-2	Normal class reconstruction results with MNIST dataset	23
Table 4-3	Abnormal class reconstruction results with MNIST dataset	24
Table 4-4	Performance metrics of each model with MNIST dataset using buffered threshold value, classifying class “1”	25
Table 4-5	Performance metrics of each model with MNIST dataset using the average reconstruction loss of the normal data as threshold, classifying class “1”	25

## LIST OF ABBREVIATIONS

<i>2D</i>	2-dimensional
<i>3D</i>	3-dimensional
<i>AUC</i>	Area under the ROC curve
<i>CBAM</i>	Convolutional Block Attention Module
<i>CNN</i>	Convolutional Neural Network
<i>Conv2D</i>	2-dimensional Convolutional
<i>Conv3D</i>	3-dimensional Convolutional
<i>Leaky ReLU</i>	Leaky Rectified Linear Unit
<i>MAML</i>	Model-Agnostic Meta-Learning algorithm
<i>MemAE</i>	Memory-augmented Autoencoder
<i>MNIST</i>	Modified National Institute of Standards and Technology
<i>MSE</i>	Mean Square Error
<i>ReLU</i>	Rectified Linear Unit

## CHAPTER 1 INTRODUCTION

In this chapter, the background and motivation of our research, our contributions to the field are presented.

### 1.1 Anomaly Detection

Information technologies have been more important than ever nowadays, and this field has been opened up by many related field specialists. The increase in the importance of information technology leads to an increase in the importance of data, where data is the raw form of information that has not been transformed into useful details. The data used in this information era is tremendous, therefore we need to have some methods to analyse the data, and anomaly detection is one of the techniques for analysing abnormal data.

The term Anomaly Detection is self-explanatory, which is to identify the anomaly data from the normal data. More specifically, anomaly detection is the process of discovering patterns in data that do not conform with expected behaviour, and those non-conforming patterns are known as anomalies or outliers [2]. Anomaly detection can be also referred to as the technique of finding data points that fall apart from the majority of data points [9]. Figure 1-1 illustrates the anomaly that the pattern in actual data does not conform with the pattern in expected data, and the part where the actual data spikes out is identified as an anomaly. Figure 1-2 illustrates the anomalies in a two-dimensional data set, where the data points at o1, o2 and in the O3 region are anomalies because the majority of data points lie in the N1 and N2 regions.

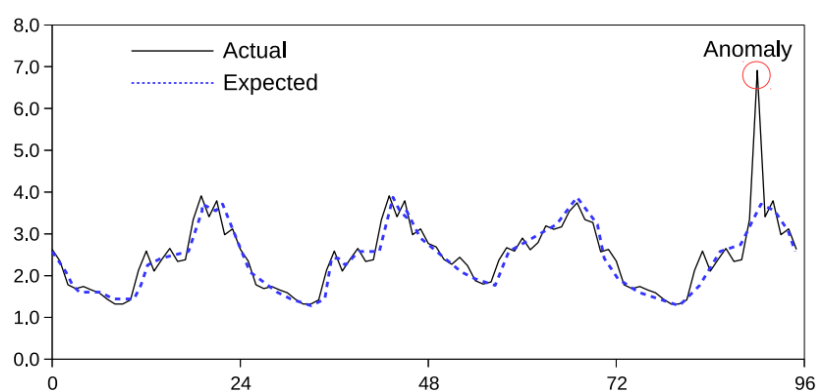
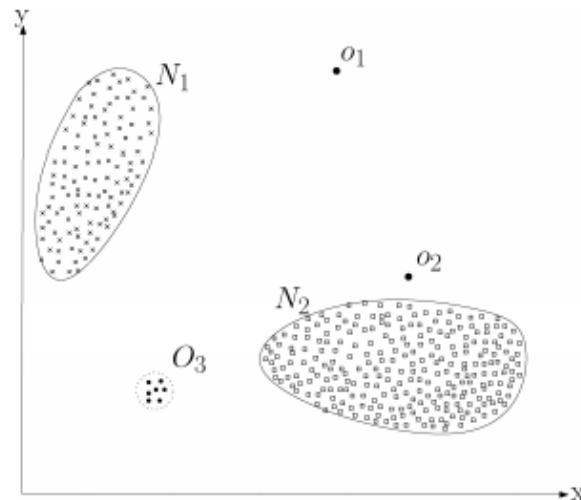


Figure 1-1: Anomaly where the data pattern does not conform with expected behaviour [6].



*Figure 1-2: Anomalies in a two-dimensional dataset where the region with the minority of data points are anomalies [2].*

The topic of anomaly detection has gained strong attention from researchers due to its increasing demand where the domain of anomaly detection applications is vast, such as video surveillance, financial technology, telecommunication, and healthcare. The importance of anomaly detection is proven by the increasing demand for it in various fields, as it helps humans to identify abnormal data among a huge dataset in a manner of rapid and accurate, such that humans could not match its performance. Finding anomalies is crucial because anomalies often deliver important information to us [2]. Most of the anomaly detection model is in the unsupervised mode because the anomalous data are often limited, i.e., the normal data are much more than the anomalous data since anomalies are rare in nature. Therefore, it is impossible to perform anomaly detection by using a standard classification framework. The existing anomaly detection models are realized by learning the normal profile with the normal data only, and anomalies are identified based on the distance between the given sample and the normal profile learned [7].

In this project, we are focusing on the application of anomaly detection on images. The application of anomaly detection in images is also significant, where the anomalies in video surveillance camera footage could indicate an unexpected situation happened, such as if the placement of the camera is in a motorway, the anomaly scene could be a traffic accident; if its placement is in a bank, the anomaly scene could then be an intrusion. However, the application of anomaly detection in videos is not popular due to its reliability. Video anomaly detection is far more challenging in that the data points lie in a high-dimensional space and modelling the high-dimensional data is extremely complex [4]. Furthermore, video anomaly detection

becomes more difficult, considering the scene of the video is often affected by various conditions, such as lighting conditions, camera settings, object occlusions, crowd density, etc. [8]. Since video footage is formed by sequential frames and each frame is an image, in order to make this project more manageable and controllable, we will focus on anomaly detection on images, and it will directly imply video applications.

### 1.2 Problem Statement and Motivation

As mentioned, our project is focusing on image anomaly detection. Assume that the video comes from a surveillance camera in any place, such as a motorway, household, bank and anywhere else that would require surveillance from a camera containing an anomaly frame. Our key objective is to identify the frame and mark the video as an anomaly.

Given a video frame input, the region where the anomaly happens is often known as anomaly locality. Most of the past approaches that deal with anomaly detection only try to study the anomalies in a full video frame, yet it is considered low accuracy due to the fact that anomalies often only happen in a narrow portion of the frame [5]. Accuracy is one of the crucial factors to consider as it is the benchmark of how well the model could perform. It is important to note that the portion where anomalies happen in the frame is the criterion of anomaly detection, and the model should pay attention to the anomaly locality. As shown in Figure 1-3, assuming a surveillance camera that is fixed in a household trying to detect burglary, the anomaly locality in the video frame, i.e., the precise region in the frame where the burglar is trying to steal something (bounded by the green box), occupies only a limited portion of the frame instead of the whole frame. The inspiration for adding attention mechanism to the anomaly detection model is drawn after understanding this observation.



*Figure 1-3: Anomaly locality in a household burglary video [5].*

[4] proposed a memory-augmented deep autoencoder for unsupervised anomaly detection to solve the weakness of autoencoder where it often generates low reconstruction error for anomalies and leads to failure of anomaly detection. However, the model in [4]'s paper does not consider the key observations discussed above. Therefore, we are aiming to improve the work of [4] by adding the attention mechanism to it.

### 1.3 Project Scope and Objectives

#### 1.3.1 Project Scopes

This project aims to mimic video anomaly detection models by building a deep learning model that is able to perform anomaly detection on images. To improve the quality of anomaly detection, we try to add an attention mechanism to the model to enhance the anomalies on the image and therefore, the model should only pay attention to the anomaly locality as it could retrieve the most important information from that frame region.

We will be using a Conv2D deep autoencoder as our foundation to build the anomaly detection model and incorporating the Convolutional Block Attention Module (CBAM) from [10] and the attention mechanism proposed by [11] separately and forms two new models to boost the performance of deep autoencoder when performing anomaly detection. The flow of the proposed anomaly detection model should be as follow, the input is first resized and transformed into the desirable input format of the model. Then, the feature maps produced by subsequent convolutional layers are enhanced by CBAM to focus on the anomalies. Then, the encoding of the input will go through the latent representation space, i.e., fully connected layers to learn the extracted features and the output will be a vector  $z$ . Next, the decoder will use vector  $z$  to reconstruct the input. Lastly, the reconstructed input will be compared with the original input and compute reconstruction error, and anomalies are detected based on the reconstruction error.

#### 1.3.2 Project Objectives

The aim of this project is to mimic the video anomaly detection model by using a similar setting from [4] to build a Conv2D deep autoencoder for processing images. Then, attention mechanisms will be implemented on the model to enhance its performance.

1. To build a Conv2D deep autoencoder from scratch that is capable of detecting image anomalies.

## CHAPTER 1: INTRODUCTION

2. To incorporate CBAM to form a CBAM-based deep autoencoder and monitor its performance.
3. To incorporate the attention mechanism proposed by [11] to build an attention-based deep autoencoder and monitor its performance.
4. To compare the performance between baseline deep autoencoder, CBAM-based deep autoencoder, and attention-based deep autoencoder.



## CHAPTER 2 LITERATURE REVIEW

### 2.1 Memorizing Normality to Detect Anomaly: Memory-augmented Deep Autoencoder for Unsupervised Anomaly Detection [4]

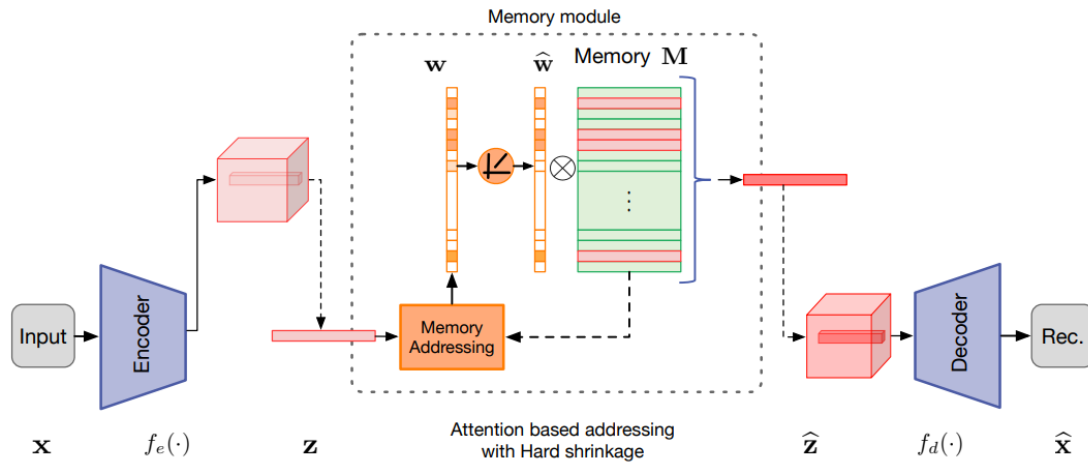


Figure 2-1: An overview of the Memory-augmented Autoencoder [4].

#### 2.1.1 Memory-augmented Autoencoder

[4] introduced the memory-augmented autoencoder (MemAE) for reconstruction-based anomaly detection. The motivation for proposing MemAE is to solve some limitation that occurs in its basic form, which is the autoencoder. Deep autoencoders have been extensively used for anomaly detection in the existing researches, it is an unsupervised deep learning algorithm that is capable of reconstructing high-dimensional input. To avoid confusion, a deep autoencoder is an autoencoder with many hidden layers in the middle, where the number of hidden layers depends on the user settings. An autoencoder has 3 main parts, the Encoder that is responsible for compressing the input data into compressed encoding (low-dimensional data); the Decoder that can reconstruct the data from the encoding; the “bottleneck” hidden layer in the middle containing the latent representation of the input data. The process of compressing input data is simply meant by omitting the irrelevant information in the high-dimensional data while retaining the important information, therefore the decoder is able to reconstruct the compressed input based on the typical patterns of the original data.

As for the applications of autoencoder in anomaly detection, the autoencoder works as mentioned above, i.e., the encoder compressed the high-dimensional data, the “bottleneck” layer learns the encoding (latent representation of the input), the decoder uses the encoding to reconstruct the data. Then, the reconstruction error is computed by taking the difference in the

original data and the reconstructed data, and anomalies are able to be detected based on the reconstruction error since they tend to produce higher reconstruction error. Note that, the autoencoder is often trained with only normal data and it is able to minimize the reconstruction error on normal data since the training data are closer to normal data, therefore the reconstruction error on abnormal data are relatively higher. However, the scenario of anomalies producing higher reconstruction errors is not constant. In the existing researches, it is shown that the autoencoder are able reconstruct the abnormal data so well that the reconstruction error is similar to those from normal data. Also, the assumption of anomalies producing higher reconstruction errors cannot be proven as there are no training samples for anomalies. To address the issue introduced above, [4] proposed MemAE which could strengthen the reconstruction errors on anomalies.

The MemAE is an improved version of the autoencoder by adding a memory module in the middle of the encoder and decoder as shown in Figure 2-1. Similarly, the MemAE is also having an encoder and decoder, however, the output of the encoder (encoding) is not directly fed into the decoder, but use as a query to retrieve the most relevant items in the memory via the attention-based sparse addressing, the retrieved item is then sent to the decoder for reconstruction. The flow of anomaly detection on MemAE is shown in Figure 2-1. First, the input  $x$  is fed into the encoder  $f_e(x; \theta_e)$  and the encoding  $z$  is obtained. Then, the retrieved item (latent representation)  $\hat{z}$  is obtained through the attention-based addressing with hard shrinkage operator and it is passed to the decoder  $f_d(\hat{z}; \theta_d)$  to obtain the reconstructed input  $x$ . Next, given an input  $x$ , the reconstruction performance is measured by  $\ell_2$ -norm based mean square error (MSE), i.e.,  $e = \|x - \hat{x}\|_2^2$ , and act as a benchmark for anomaly detection. Since the encoder and decoder are very much the same as mentioned above, we focus on the memory module proposed by [4].

### 2.1.2 Memory Module in MemAE

In the memory module, given an encoding (query), the memory network retrieves  $\hat{z}$  from the memory  $M$  based on a soft addressing vector  $w$  that follows the equation:

$$z = wM = \sum_{i=1}^N w_i m_i \quad [4] \quad (2.1)$$

where  $w$  (representing weight vector) is a row vector with non-negative entries that sum to one,  $w_i$  representing the  $i^{\text{th}}$  row of  $w$ ,  $m_i$  is the  $i^{\text{th}}$  row of  $M$  (also can view as a memory item), and  $N$  is the maximum capacity of the memory. The memory  $M$  is designed to record the prototypical

normal patterns during the training using normal data, and it is addressable by content with an addressing scheme that computes attention weights  $w$  based on the closeness of the memory items  $m_i$  and the query  $z$ . The computation of attention weights  $w_i$  for each memory item  $m_i$  is realized by a softmax operation. Because of the restricted memory size and the sparse addressing technique, only a limited number of memory items can be addressed every time. Therefore, the memory  $M$  will only record the most representative prototypical pattern in the normal input data during the training phase, and each row in the memory  $M$  can view as a single memory slot that records the prototypical normal pattern in the training data. Given an abnormal input in the testing phase, since the trained memory is full of prototypical elements from the training data, the encoding of it could not find the respective pattern for it, and it will then be substituted by the retrieved prototypical normal pattern and fails to reconstruct back to the original abnormal input, leads to high reconstruction error and we may understand that it is an anomaly. The sparse addressing technique mentioned is realized by hard shrinkage operation as shown in the following:

$$\widehat{w}_i = h(w_i; \lambda) \begin{cases} w_i, & \text{if } w_i > \lambda \\ 0, & \text{otherwise} \end{cases} \quad [4]$$

It is used to stimulate the sparsity of  $w$  to prevent anomalies from making use of a complex combination of memory items via a dense  $w$  to reconstruct well. For any  $w_i$  larger than the threshold  $\lambda$ , its value will be retained or else turned into zero, and the updated weight vector is denoted as  $\widehat{w}_i$ . Therefore, the way to obtain latent representation  $\hat{z}$  become  $\hat{z} = \widehat{w}M = \sum_{i=1}^N \widehat{w}_i m_i$ .

### 2.1.3 Limitations and Proposed Solutions

The MemAE proposed by [4] has perfectly solved the problem where autoencoders tend to reconstruct abnormal data successfully and leads to failure of anomaly detection. However, this model still contains some limitations. As mentioned in Chapter 1.2 where we have shown the problems that should be solved.

We found that the anomaly detection process using the MemAE model is by processing the whole frame from a video. We understand that the MemAE captures the important features (prototypical normal pattern) of the input data and store it in the memory module, then use it for frame reconstruction and detect anomalies based on the reconstruction error. Given an abnormal input frame, we have shown the concept of anomaly locality in Chapter 1.2, where the precise region of the abnormal act is often occupied only a limited portion of the input

frame instead of the whole frame. Since the autoencoders tends to capture the important features in a frame, the anomaly locality should be focused on by autoencoders where it could gain the most representative information from there. Therefore, we aim to incorporate the attention mechanism on the autoencoders, and the anomalous region of the input could gain attention from the autoencoders.

## 2.2 Few-Shot Scene Adaptive Anomaly Detection [7]

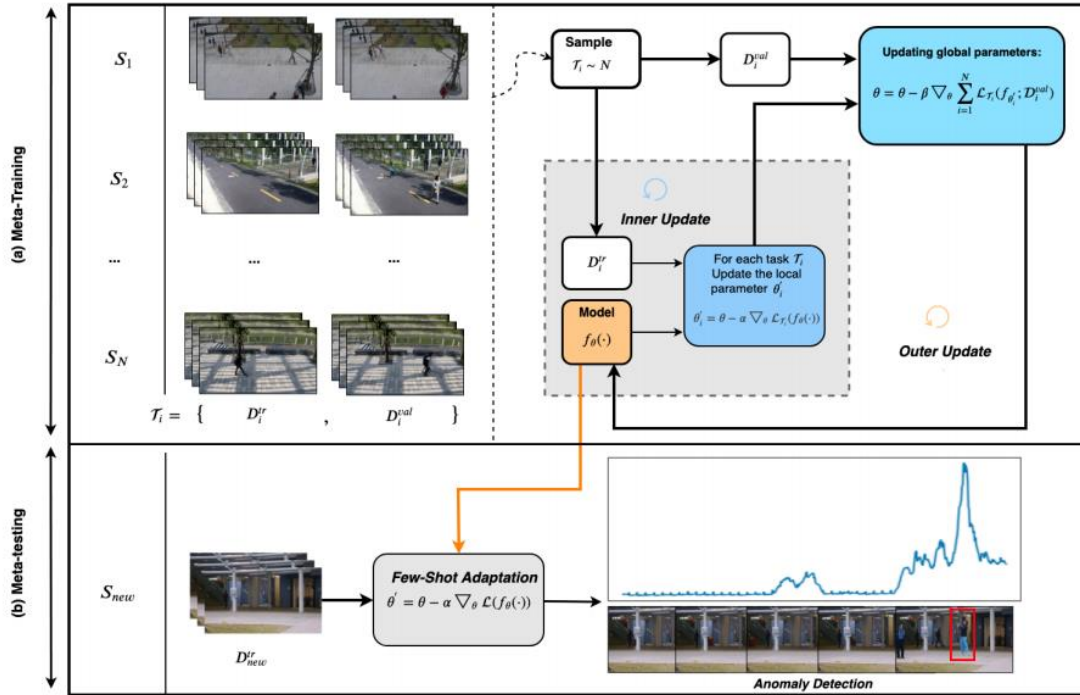


Figure 2-2: An overview of the Model-Agnostic Meta-Learning algorithm for Scene-Adaptive Anomaly Detection [7].

### 2.2.1 Problem Setup

[7] introduced the few-shot scene-adaptive anomaly detection problem setup. To solve the problem, they proposed a meta-learning based approach, i.e., the Model-Agnostic Meta-Learning (MAML) algorithm for Scene-Adaptive Anomaly Detection. The motivation of this problem setup is by understanding the fact that most of the existing anomaly detection approaches are not practical in real-world applications, as they require a huge number of videos from a scene for training to obtain great results in that particular scene, and this is not a practical way to do in real-life scenario due to collecting a huge number of videos is costly and time-consuming. Also, [7] found out the existing approaches are having the same weakness, which

is the model learned from the training dataset (videos) performs poorly with the testing dataset (unseen videos) if the videos in both datasets are taken from different scenes.

[7] observed that the scene taken by a surveillance camera will always be in that particular scene since most of the surveillance camera in daily life is in a fixed position. This observation generates the idea that an anomaly detection system on any surveillance camera is considered ideal, as long as it could perform well (adapt) to the particular scene where the system is deployed. Furthermore, the model should only be trained with a limited number of videos before the deployment to be in line with real-world applications, which is a reasonable assumption as surveillance cameras often have a calibration process before their actual operation. To solve the problem setup introduced, [7] proposed the MAML algorithm for scene-adaptive anomaly detection, and the ultimate goal of this algorithm is to let the anomaly detection model adapt to a new scene using only a limited number of videos from that particular scene, which it makes the anomaly detection becomes more practical to use in a real-world scenario. Therefore, by using the meta-learning based approach, which is also known as learning to learn, the model is learning to learn to adapt to a new scene with a limited number of frames from that scene.

### 2.2.2 Meta-learning Framework

The meta-learning framework is divided into 2 parts, i.e., the meta-training part and the meta-testing part as shown in Figure 2-2. The goal of meta-training is to let the anomaly detection model learn to quickly adapt to a new task  $T_i$ , where each task consists of a training set  $D_i^{tr}$ , and a validation set  $D_i^{val}$  from a given scene  $S_i$ , therefore we can also view it as  $T_i(D_i^{tr}, D_i^{val})$ . Assuming we are now having access to  $M$  scenes, i.e.,  $S_1, S_2, S_3, \dots, S_M$ . Before diving into the meta-training phase, the tasks need to be constructed. First, the video from a given scene  $S_i$  is split into many overlapping sequential segments, where each segment are the frames from the video  $(I_1, I_2, I_3, \dots, I_t, I_{t+1})$ . Considering a frame-prediction based anomaly detection model, the goal is to predict the frame at  $t+1$ , therefore the first  $t$  frames in the segment are considered as the input  $x$ , and the last frame is the output  $y$ , i.e.,  $x = (I_1, I_2, I_3, \dots, I_t)$  and  $y = I_{t+1}$ , forming an input/output pair  $(x, y)$ . The frame-prediction based anomaly detection model can also be denoted as  $f_\theta = x \rightarrow y$ , where  $\theta$  is the parameter of the model. In the training set  $D_i^{tr}$ ,  $K$  numbers of  $(x, y)$  from the video is sampled out to learn the frame-prediction based anomaly detection model, i.e.,  $D_i^{tr} = \{(x_1, y_1), (x_1, y_1), (x_1, y_1), \dots, (x_k, y_k)\}$ , and also  $K$  numbers of  $(x, y)$  (excluding those in  $D_i^{tr}$ )

are sampled out from the same video to form the validation set  $D_i^{val}$ . In Figure 2-2,  $N$  scenes are sampled out and a task  $T_i$  is constructed for each scene  $S_i$ , therefore, there is  $N$  number of  $T_i$  for meta-training.

In the meta-training part, for each task  $T_i$ , the training set  $D_i^{tr}$  is used for *Inner Update* through the gradient descent algorithm, and the gradient update is used to update the anomaly detection model parameter from  $\theta$  to  $\theta'_i$  for each task  $T_i$ , and the scene-adapted parameter  $\theta'_i$  is adapted to task  $T_i$ . The equation for the algorithm above is shown as following [7]:

$$\theta'_i = \theta - \alpha \nabla_{\theta} L_{T_i}(f_{\theta}; D_i^{tr}), \text{ where}$$

$$L_{T_i}(f_{\theta}; D_i^{tr}) = \sum_{(x_j, y_j) \in D_i^{tr}} L(f_{\theta}(x_j), y_j) \quad (2.2)$$

Then, the performance of scene-adapted parameters  $\theta'$  obtained from all tasks  $T$  are validated on the validation set  $D_i^{val}$  with the equation [7]:

$$L_{T_i}(f_{\theta'}; D_i^{val}) = \sum_{(x_j, y_j) \in D_i^{val}} L(f_{\theta'}(x_j), y_j) \quad (2.3)$$

As we can see from the equation, the scene-adapted parameters  $\theta'$  will update the model parameters  $\theta$  through the *Outer Update* process, in other words,  $\theta'$  will minimize the loss during the *Updating global parameters* process as shown in Figure 2-2. At the end of meta-training, the initial model parameters  $\theta$  are obtained and ready for meta-testing. Figure 2-3 shows the general flow of the meta-training algorithm, where Eq. 3 mentioned in it is the equation used for validation.

---

```

Input: Hyper-parameters  $\alpha, \beta$ 
Initialize  $\theta$  with a pre-trained model  $f_{\theta}(\cdot)$ ;
while not done do
    Sample a batch of scenes  $\{S_i\}_{i=1}^N$ ;
    for each  $S_i$  do
        Construct  $\mathcal{T}_i = (D_i^{tr}, D_i^{val})$  from  $S_i$ ;
        Evaluate  $\nabla_{\theta} \mathcal{L}_{\mathcal{T}_i}(f_{\theta}; D_i^{tr})$  in Eq. 1;
        Compute scene-adaptative parameters  $\theta'_i = \theta - \alpha \nabla_{\theta} \mathcal{L}_{\mathcal{T}_i}(f_{\theta}; D_i^{tr})$ ;
    end
    Update  $\theta \leftarrow \theta - \beta \sum_{i=1}^N \nabla_{\theta} \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_i}; D_i^{val})$  using each  $D_i^{val}$  and  $\mathcal{L}_{\mathcal{T}_i}$  in Eq. 3;
end

```

---

Figure 2-3: Summary of the meta-training phase [7].

In the meta-testing part, the model takes in few frames of a video from a new scene  $S_{new}$  for training, and obtain the scene-adapted parameter  $\theta'$  to become adapted to the new scene  $S_{new}$ . Lastly, the model takes in the remaining frames of the video (excluding those for training) for

validation purposes. Note that, [7] used a frame-prediction based anomaly detection model in their paper, however, they also mentioned the algorithm is suitable for any anomaly detection model as the backbone architecture, such as the frame-reconstruction based model.

### 2.3 Convolutional Block Attention Module [10]

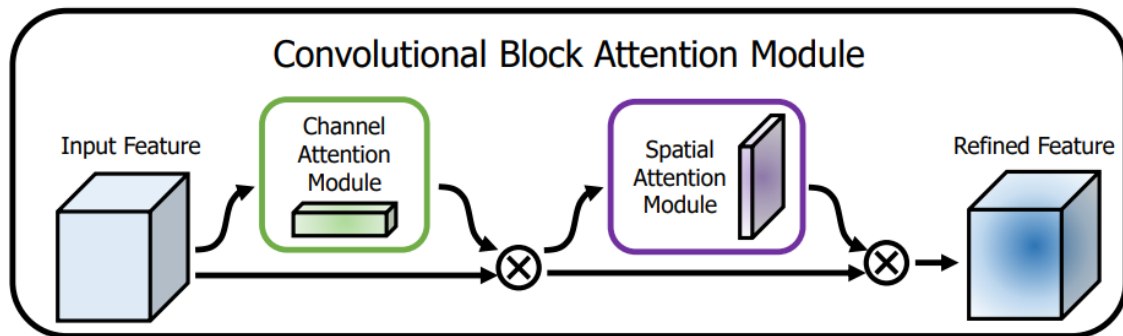


Figure 2-4: The overall view of Convolutional Block Attention Module [10].

[10] introduced Convolutional Block Attention Module (CBAM) as an attention mechanism that may boost the performance of the model. The input feature may see it as a feature map from a convolutional layer, and the refined feature is the enhanced feature map with characteristics of the image emphasized on it, such as edges, colors, background, etc.

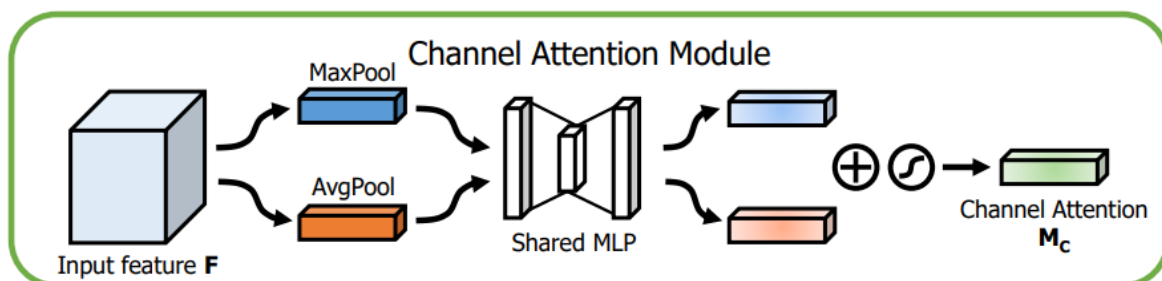


Figure 2-5: Channel Attention Module [10].

Firstly, the input features pass through the channel attention module. The feature map  $F$  will be processed by a 2D average pooling layer to get a channel vector ( $\text{channel\_num} * 1 * 1$ ). Then, this channel vector will go through a shared Multi-layer Perceptron (MLP) that consists of 2 fully connected layers with a ReLU activation function between them. Next, the same feature map  $F$  will also go through a max pooling layer to get another channel vector, similarly, it also goes through the MLP. After that, the 2 channel vectors that have been processed by MLP will be summed up and pass to a sigmoid activation function which then maps the value in the

vector within 0 and 1. Then, the vector will multiples with the original input to get a channel-refined feature map  $F'$ .

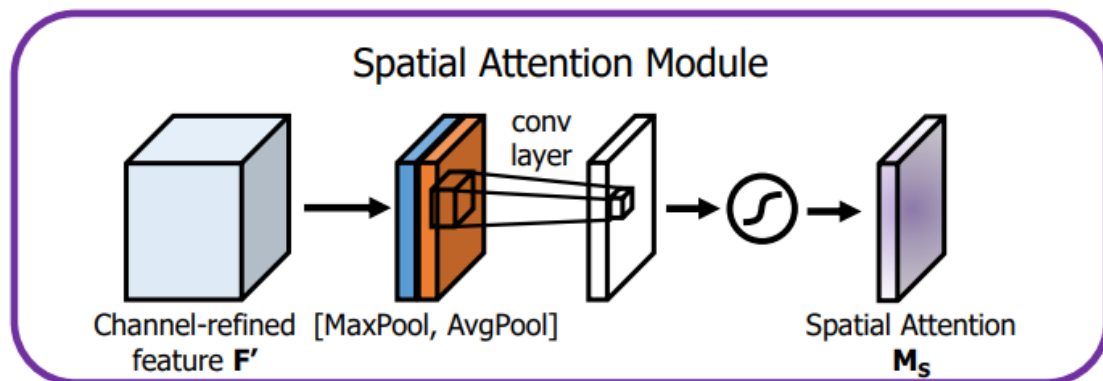


Figure 2-6: Spatial Attention Module [10].

Then, the channel-refined feature map  $F'$  will go through the spatial attention module. The spatial attention module consists of 3 different parts. Firstly,  $F'$  go through the Channel Pool and it is decomposed into 2 channels ( $2 * h * w$ ), where each of the 2 channels will be max pooled and averaged pooled across the channels by using torch.max and torch.mean. Then, the processed channels will be concatenated and serves as the input to a convolution layer which produces 1-channel feature map ( $1 * h * w$ ) to preserves the spatial dimension, followed by batch normalization and ReLU activation. Then, the output go through a sigmoid activation function to maps all the values within 0 and 1. Lastly, the attention mask is applied to all the feature maps using an element-wise product.

## 2.4 Learn to Pay Attention [11]

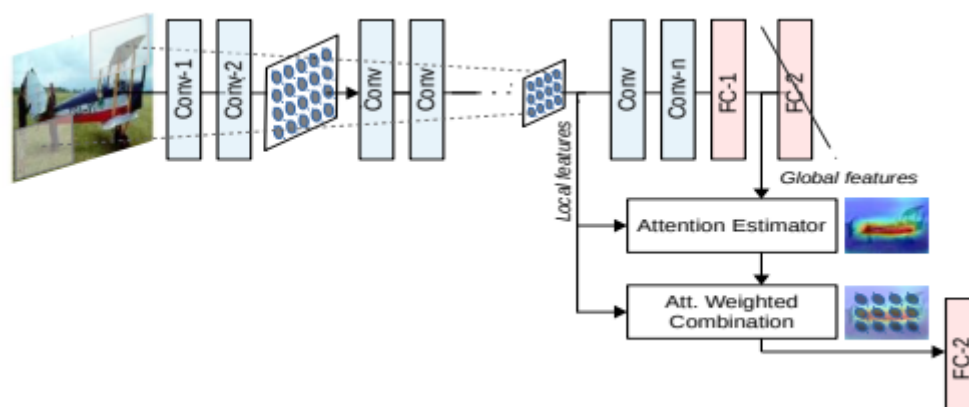


Figure 2-7: The overview of the attention mechanism introduced by [11].



[11] introduced an end-to-end-trainable attention module for convolutional neural network (CNN) models built for image classification. The main objective of the paper is to employ attention maps to identify and utilize the useful spatial information used by CNNs in making their classification decision while suppressing the irrelevant spatial information on the image. The proposed approach by [4] is as following, intermediate outputs (feature maps) of the CNN model are taken out and act as local descriptors  $l_i$  of the input, and they are used to contribute to the final classification step with proportional to its compatibility with the final output (global image descriptor  $g$ ) of the CNN model. To measure their compatibility, dot product between  $l_i$  and  $g$  is used, and the value will be high only if the corresponding image patch of the  $l_i$  contain parts of the dominant image category. The measurement of the compatibility score will be dependent on the alignment between  $l_i$  and  $g$  in the high dimensional feature space and the intensity of activation of  $l_i$ .

## CHAPTER 3 PROPOSED APPROACH

In this chapter, the methods and technologies involved in building the anomaly detection model will be presented.

### 3.1 System Requirements

The model is developed using Python programming language and Pytorch deep learning framework. The model is developed on Jupyter Notebooks environment and Google Colab's GPU will be utilized to accelerate the training and testing process.

### 3.2 System Design

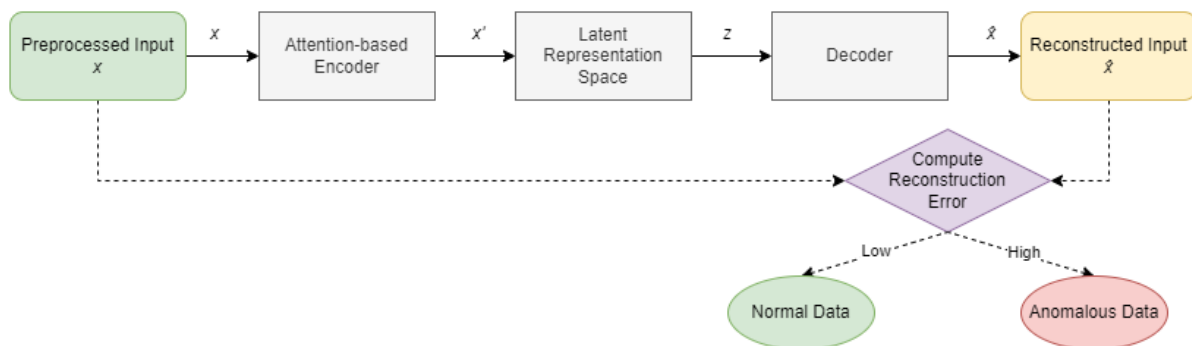


Figure 3-1: Block diagram of proposed model.

#### 3.2.1 General Flow of Proposed Model

Figure 3-1 shows the overview of the model proposed in this project. When training, the input will first be preprocessed by some procedures such as image resizing so that the model could process all the frames in the same size, transform into torch.tensor and normalize it. Next, the preprocessed input  $x$  will be fed into the encoder, and the encoder will compress the input and generate the feature maps (compressed input  $x'$ ). The CBAM will be placed in between the encoder layers to enhance the features on the feature maps and let the latent representation space learn more about the representative features. Then, the decoder uses the vector  $z$  generated from the latent representation space to reconstruct the input and gives us the reconstructed input  $\hat{x}$ .

When testing, the autoencoder is expected to be trained on normal data only. Therefore, the latent representation space only knows how to recognize normal data but not anomalous data, and any anomalous input that tries to reconstruct using the normal data features will fail. Next, the decoder will reconstruct the vector  $z$  and generate the reconstructed input  $\hat{x}$ . Lastly, the

reconstructed input  $\hat{x}$  is used to compare with the original input  $x$ . The reconstruction error will be used as a benchmark to detect whether the input is anomalous. For anomalous data, since it could not be reconstructed back into the original anomalous form, therefore leads to high reconstruction error and is identified as anomaly. For normal data, the reconstruction error will be low because the decoder is able to reconstruct it well using the normal features in the latent representation space that is relevant to the original normal data.

### 3.2.2 Attention Mechanism

For the attention mechanism, the Convolutional Block Attention Module (CBAM) and the attention mechanism proposed by [11] will be used to implement on the model. The details are presented in Chapter 2.3 and Chapter 2.4, and the implementation of our proposed model would be quite similar.

For CBAM, the feature maps produced by the subsequent encoder convolutional layer will go through the CBAM, and it will generate a refined feature map for the subsequent encoder convolutional layer. Then, the refined feature map goes through the encoder convolutional layer and CBAM with a similar process, and this produces a more refined feature map for the latent representation space. With this concept, we suppose to see the latent representation space learns better from the data features and gives us better results.

For the attention mechanism proposed by [11], the attention computation block takes in intermediate outputs of the encoder and sums it up with the final output of the encoder to generate a similarity weight. Then, the similarity weight is used to enhance the intermediate outputs of the encoder and it is then fed into the bottleneck layer of the autoencoder. By doing so, the bottleneck layer can learn from the enhanced feature maps and recognizes the normal input while suppressing the anomalous input.

### 3.3 Model Architecture

#### 3.3.1 Baseline Deep Autoencoder

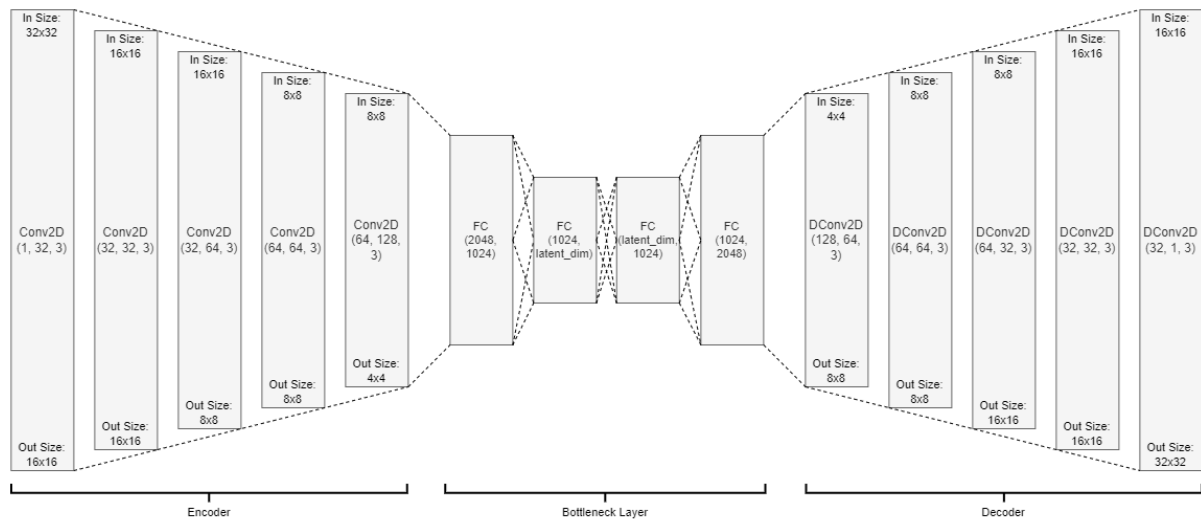


Figure 3-2: Illustration of the proposed deep 2D convolutional autoencoder.

Figure 3-2 shows the baseline model of this project. It is a typical deep autoencoder consist of three main components, the encoder, the latent representation space (bottleneck layer), and the decoder. The encoder consists of five 2D convolutional (Conv2D) blocks and each block contains a 2D convolutional layer, a 2D batch normalization layer, and a Leaky Rectified Linear Unit (Leaky ReLU) activation layer with a negative slope of 0.2. The number of input channels, output channels and kernel size of the convolutional layers are shown in the figure (input, output, kernel).

The bottleneck layer consists of four fully connected (FC) layers, with a ReLU activation layer placed between them. The number of inputs is the flattened size of the output from the encoder, i.e.,  $4 * 4 * 128 = 2048$ . It serves to learn the extracted features from the encoder, and one may think of it as a space with the latent representation of the inputs.

The decoder consists of five 2D transposed convolutional (DConv2D) blocks. Similarly, the DConv2D blocks have the same contents as the Conv2D blocks, just the input channels and output channels are flipped since they are doing opposite actions. The last DConv2D block has its batch normalization layer and activation layer removed to maintain the reconstructed values.

The following 2 models proposed in this project are both attention-based, implemented on the encoder of the baseline model to enhance the model to pay attention to the most representative features of the input and learn from it.

### 3.3.2 CBAM-based Deep Autoencoder

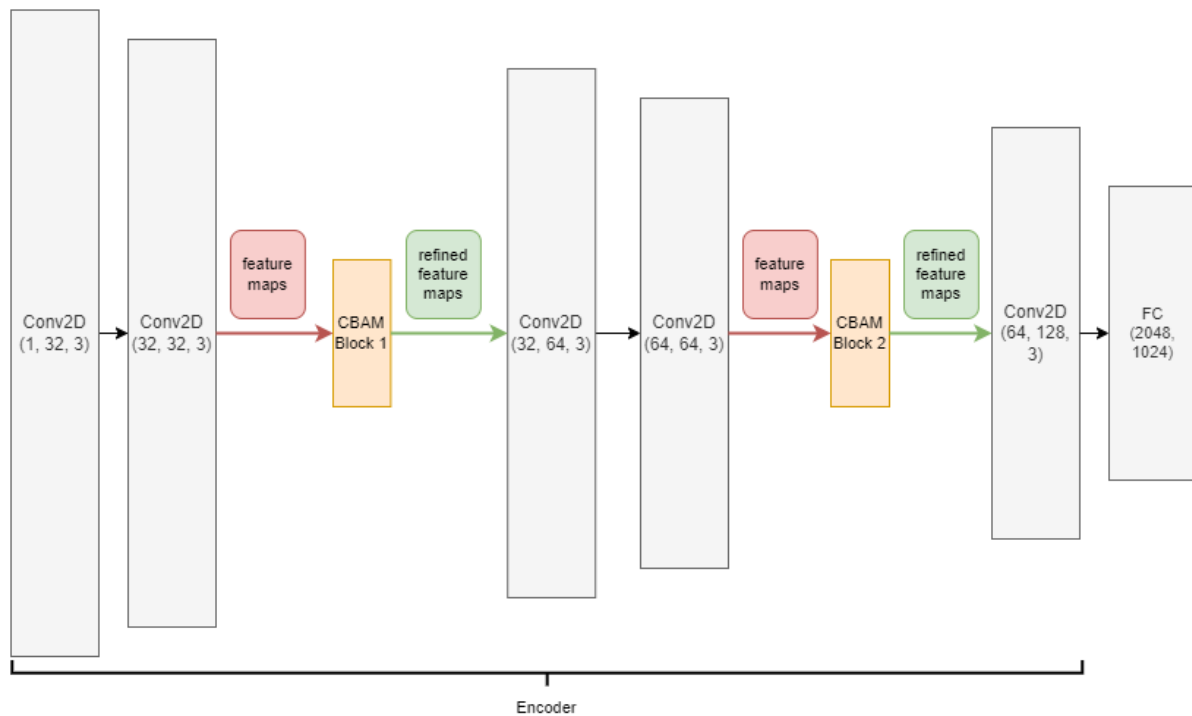


Figure 3-3: Implementation of CBAM on the encoder of the model.

The second model proposed in this project is inspired by [10] where the CBAM is implemented on the encoder of the model. The output of the second Conv2D block will go through the CBAM Block 1 and obtains refined output. Then, the refined output is passed down to the deeper layer of the encoder. The output is once again refined by the CBAM Block 2 when obtained the feature maps from the fourth Conv2D block. The fifth Conv2D block will receive twice-refined feature maps and feed them into the latent representation space for it to learn about feature-emphasized outputs. A better learning outcome is expected from the model, and it should perform reconstruction better. The CBAM block is a general module for CNN architectures with the least amount of configuration needed, thus, there might be no big improvements to be observed.

### 3.3.3 Attention-based Deep Autoencoder

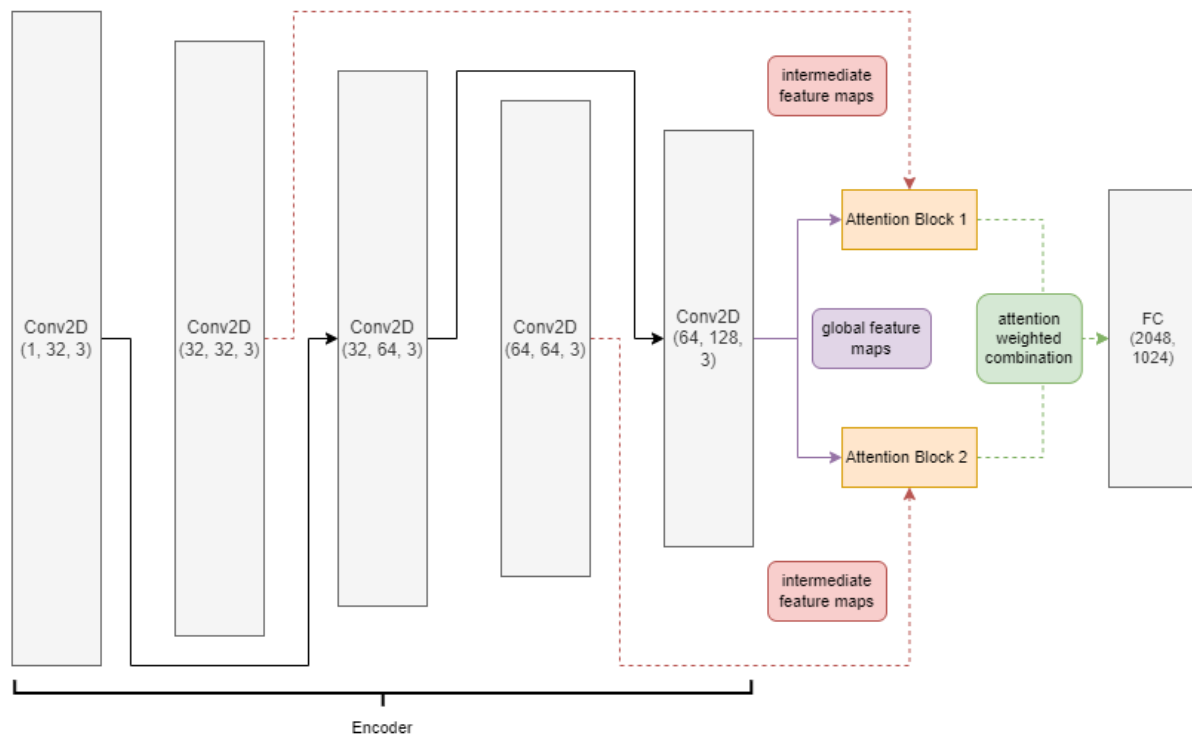


Figure 3-4: Implementation of [11] proposed attention mechanism on the encoder of the model.

The third proposed model in this project is implementing the attention mechanism proposed by [11]. The outputs of the second Conv2D and fourth Conv2D blocks are pulled out and fed into attention block 1 and attention block 2 respectively. The attention computation happens in the blocks where the intermediate feature maps will be enhanced by the global feature maps, then the output is concatenated and fed into the bottleneck layer. The concept is similar to the idea from Chapter 2.4 where the intermediate feature maps will first sum up with the global feature maps and produces a similarity matrix (weighted matrix). Then, the matrix is multiplied with the intermediate feature maps to emphasize the relevant pixels on the image while suppressing those irrelevant pixels, and this enlarges the gap between relevant and irrelevant data, and this also helps the autoencoder to reconstruct normal data and anomalous input identified by computing the reconstruction error.

## CHAPTER 4 EXPERIMENTS

In this chapter, the experiments for the proposed model will be discussed.

### 4.1 Experiment Settings

In this project, we will build a Conv2D autoencoder from scratch and implement attention mechanisms on it.

#### 4.1.1 Dataset

The dataset used in this project is Modified National Institute of Standards and Technology (MNIST) database. It consists of 70,000 handwritten digits from 0 to 9, where 60,000 of it are categorized as training set and the remaining are the test set. All images in the dataset are grayscale, size-normalized and centred in a 28x28 image. The MNIST dataset is made available in the PyTorch framework in the torchvision.datasets module.

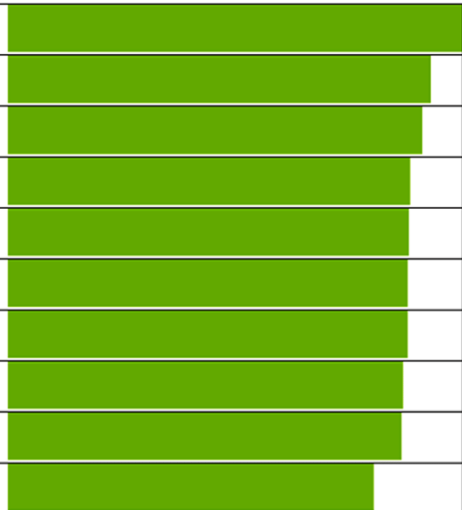
Value	Count	Percent	
1	6742	11.237%	
7	6265	10.442%	
3	6131	10.218%	
2	5958	9.93%	
9	5949	9.915%	
0	5923	9.872%	
6	5918	9.863%	
8	5851	9.752%	
4	5842	9.737%	
5	5421	9.035%	

Figure 4-1: MNIST dataset class distribution (0 ~ 9) [1].

#### 4.1.2 Training and Evaluation Setup

In this research, the objective is to detect anomalies by learning the pattern from normal data. Autoencoder is considered as an unsupervised learning technique by itself where it compresses the input into a latent representation, using use that representation to reconstruct the input and the difference between the input and reconstructed input is the key for autoencoders to learn. Therefore, the labels from both datasets are not needed in the training and evaluation phase.

Since the major task is to detect anomalies rather than perform classification, we need to do some preliminary work on both datasets. We propose to train the implemented models with one selected class from both datasets, and the samples in those classes will be identified as normal data and others being anomalies. The selected class should have the greatest number of training samples since autoencoders work better when learning more normal data features and building up a normal data profile. Below are some key training settings that have been used:

1. Transformed images in the dataset into 32x32 resolution, tensor form, and normalize the data.
2. Shuffled the dataset when loading in.
3. Batch size used is 256 with 50 epochs, early stopping mechanism is implemented where the model will stop training when convergence is reached.
4. Used Adam as the optimizer to update the model parameters with a 0.0002 learning rate.
5. Used mean-squared error as the criterion (loss function).
6. Selected normal class for MNIST: Class “1” .

To detect anomalies, a threshold value should be computed in order to classify whether the data is normal or anomalous. The proposed threshold value is obtained by computing the reconstruction error of the normal class using training data, with an addition of a buffer value on top of it to allow more samples to be identified correctly. The proposed buffer value is the mean of the quarter distance between the average reconstruction loss of the normal class and the average reconstruction loss of the abnormal classes. Assume there are  $n$  number of abnormal classes,  $\alpha$  denotes the average reconstruction loss of the abnormal class,  $\beta$  denotes the average reconstruction loss of the normal class, then it may express in

$$Threshold = \beta + \frac{\sum_{i=0}^n \left| \frac{\alpha_i - \beta}{4} \right|}{n}$$

Any reconstruction error between the original input and the reconstructed input higher than the threshold value is considered as anomalies, and vice versa. Note that the average reconstruction loss here are computed by using training data but not testing data to stimulate



the real-world scenario. However, this is not a general solution for every dataset, it needs further testing on other datasets.

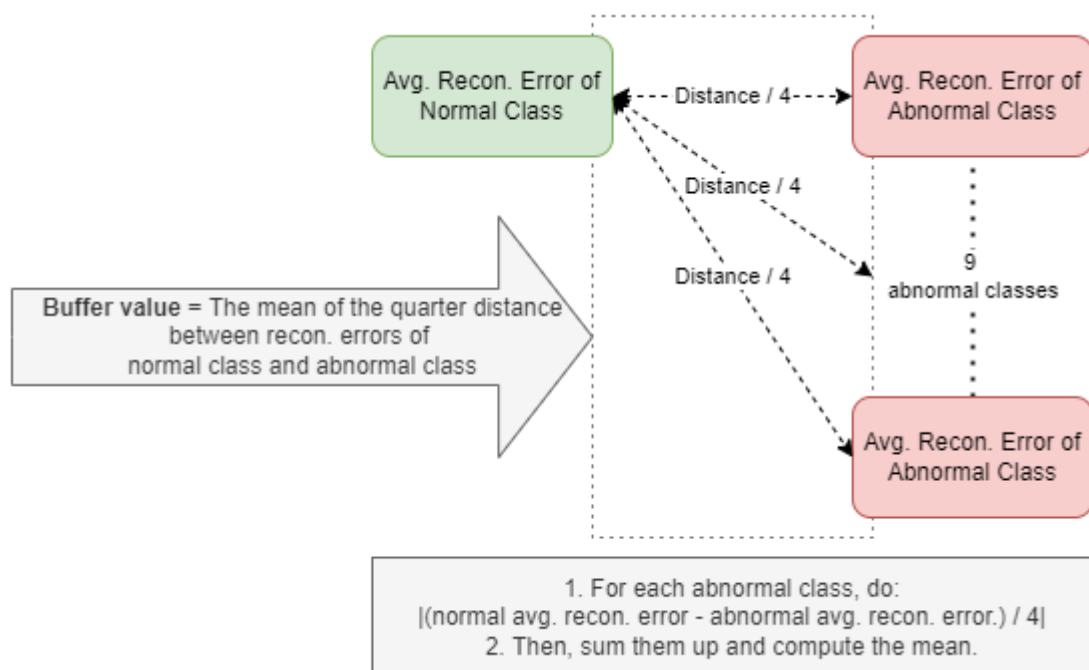


Figure 4-2: The proposed concept of calculating the buffer value for anomaly detection threshold.

Besides detecting anomalies, we could also investigate how well could the model reconstructs an image by observing the reconstructed image and the original image through bare eyes, and reconstruction error is also computed for more precise evaluation. Similar to the training settings, we used a batch size of 256 for testing as well. Then, the same criterion is used to compute and track the evaluation loss (reconstruction error) using the test set.

## 4.2 Experiment with Different Model Settings

MNIST dataset is used for each experiment.


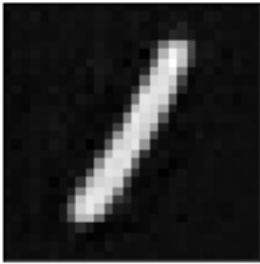
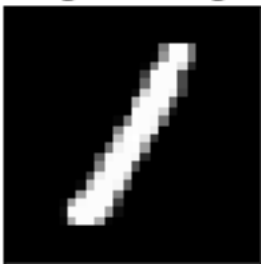
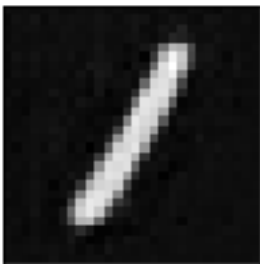
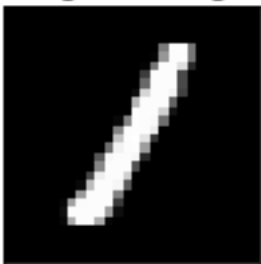
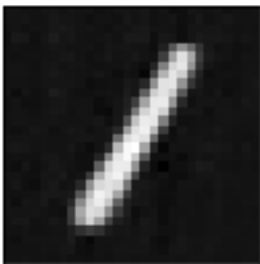
	Avg. Training Loss	Avg. Evaluation Loss	Avg. Reconstruction Error + Buffer
Baseline	0.0433	0.0197	0.0202 + 0.0947
CBAM	0.0524	0.0208	0.0207 + 0.0974
Attention-based	0.0634	0.0358	0.0348 + 0.1143

Table 4-1: Experiment metrics with MNIST dataset.


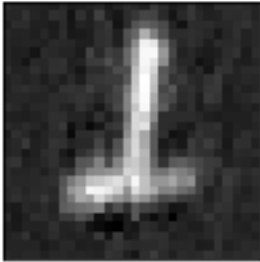



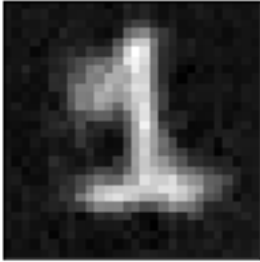
## CHAPTER 4: EXPERIMENTS

During the training with the MNIST dataset, the training loss of each epoch is kept tracked. Then, unseen data of the normal class is fed into the model to obtain average evaluation loss to check if the model is overfitted with the training set. Lastly, the average reconstruction error is computed using training data of the normal class and a buffer value is added to act as the abnormal detection threshold.

Following are some samples of the reconstruction results using unseen data.

Baseline	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Original Image</p>  </div> <div style="text-align: center;"> <p>Reconstructed Image</p>  </div> </div> <p style="text-align: center;">recon. loss = 0.013717941008508205</p>
CBAM	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Original Image</p>  </div> <div style="text-align: center;"> <p>Reconstructed Image</p>  </div> </div> <p style="text-align: center;"><b>recon. loss = 0.010425381362438202 (Lowest -&gt; Best)</b></p>
Attention-based	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Original Image</p>  </div> <div style="text-align: center;"> <p>Reconstructed Image</p>  </div> </div> <p style="text-align: center;">recon. loss = 0.04055452346801758</p>

*Table 4-2: Normal class reconstruction results with MNIST dataset.*

Baseline	<p style="text-align: center;">Original Image</p> 	<p style="text-align: center;">Reconstructed Image</p>  <p style="text-align: center;">recon. loss = 0.3641012907028198</p>
CBAM	<p style="text-align: center;">Original Image</p> 	<p style="text-align: center;">Reconstructed Image</p>  <p style="text-align: center;"><b>recon. loss = 0.442501962184906 (Highest -&gt; Best)</b></p>
Attention-based	<p style="text-align: center;">Original Image</p> 	<p style="text-align: center;">Reconstructed Image</p>  <p style="text-align: center;">recon. loss = 0.41084036231040955</p>

*Table 4-3: Abnormal class reconstruction results with MNIST dataset.*

From the results, CBAM-based autoencoder is overall the best performing model as it has the least reconstruction error when reconstructing the normal class data, and generating highest reconstruction error when reconstructing the abnormal class data. This also implies it should give the best performance metrics in the following chapter.

### 4.3 Anomaly Detection Performance Analysis

In the current stage, anomaly detection can be performed by using the threshold computed that mentioned in Chapter 4.2.

	Accuracy	Precision	Recall	F1 Score
Baseline	0.9947	0.9678	0.9859	0.9768
CBAM	0.9957	0.9748	0.9877	0.9812
Attention-based	0.9961	0.9833	0.9824	<b>0.9828</b>

*Table 4-4: Performance metrics of each model with MNIST dataset using buffered threshold value, classifying class “1”.*

	Accuracy	Precision	Recall	F1 Score
Baseline	0.9656	1.0000	0.6969	0.8214
CBAM	0.9662	1.0000	0.7022	<b>0.8251</b>
Attention-based	0.9635	1.0000	0.6784	0.8084

*Table 4-5: Performance metrics of each model with MNIST dataset using the average reconstruction loss of the normal data as threshold, classifying class “1”.*

The anomaly detection threshold is a tricky value to tackle with. The performance metrics above strongly proves the proposed buffered threshold value computation is effective. However, in some cases which requires high sensitivity, such as rare disease monitoring might still use a lower threshold to allow more data to cross the boundary even it is anomalous. Until this stage, there’s no absolute best value for the anomaly detection threshold and this is believed the only part should be improved in the future work.

Using F1 score to deduce which model performs the best, we may find that different threshold values give us different best-performing model. When using only the average reconstruction error of the normal training data, each model gives us a precision of 1.000 but slightly low recall rate and this indicates it allows too many abnormal data to be categorized as normal data. Always remember there’s no absolute answer on which threshold value is the best, it’s all depending on the usage of the model. It is good to observe that both CBAM-based and attention-based models outperform the baseline model, which indicates our implementation of attention mechanisms into deep autoencoders are effective. Confusion matrices of the models are attached at Appendix for further reference.

## CHAPTER 5 CONCLUSION

An attention-based 2D convolutional autoencoder has been successfully implemented in this research project. This project introduces the implementation of the attention mechanism on deep autoencoders and furtherly proves its effectiveness and improvements.

The CBAM-based autoencoder uses a similar concept of self-attention mechanism. The output of an intermediate layer of the autoencoder is brought into the CBAM block. In the block, a duplicated input is processed and generates an attention map, and it is used to enhance the original input by multiplying them together. Then, the product continues to the main pipeline of the autoencoder for further processing.

The attention-based autoencoder uses the final feature maps to boost the intermediate feature maps and the representative features are enhanced. Then, the enhanced intermediate feature maps are fed into the bottleneck layer for feature learning.

From the experiments conducted, a conclusion can be made which is the CBAM-based autoencoder outperforms the baseline model and the attention-based autoencoder by a minor factor, whereas the attention-based autoencoder performs similar to the baseline model. The performance gap might be even larger if the training set is larger since the autoencoder requires a huge amount of data to learn how to generate complete data from incomplete sources.

Lastly, the anomaly detection threshold is a value that is hard to decide, and this should include many mathematical computations to obtain the optimum value. This is hoped to be coped in the future work.

**REFERENCES**


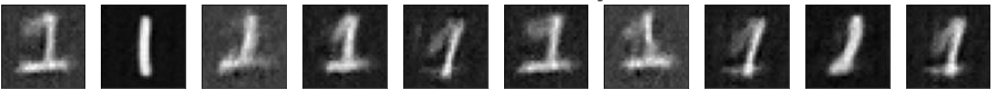



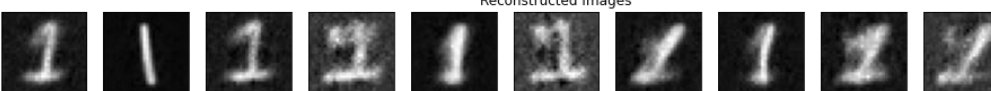
- [1] A. Alsaafin and A. Elnagar, "A Minimal Subset of Features Using Feature Selection for Handwritten Digit Recognition", in *Journal of Intelligent Learning Systems and Applications*, 2017, pp. 55–68, doi:10.4236/jilsa.2017.94006.
- [2] V. Chandola, A. Banerjee and V. Kumar, "Anomaly detection. In ACM Computing Surveys", in *Association for Computing Machinery (ACM)*, 2009, pp. 1–58, doi:10.1145/1541880.1541882
- [3] L. Deng, "The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]", in *IEEE Signal Processing Magazine*, 2012, pp. 141–142, doi:10.1109/msp.2012.2211477.
- [4] D. Gong, L. Liu, V. Le, B. Saha, M. R. Mansour, S. Venkatesh and A. Van Den Hengel, "Memorizing Normality to Detect Anomaly: Memory-Augmented Deep Autoencoder for Unsupervised Anomaly Detection", in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, doi:10.1109/iccv.2019.00179
- [5] F. Landi, C.G. Snoek and R. Cucchiara. (2019). Anomaly locality in video surveillance. [Online]. Available: [https://www.researchgate.net/publication/330726359\\_Anomaly\\_Locality\\_in\\_Video\\_Surveillance](https://www.researchgate.net/publication/330726359_Anomaly_Locality_in_Video_Surveillance)
- [6] X. Liu, and P.S. Nielsen.(2016). Regression-based online anomaly detection for smart grid data. [Online]. Available: <https://www.semanticscholar.org/paper/Regression-based-Online-Anomaly-Detection-for-Smart-Liu-Nielsen/ea729aa7fec8c6808b8eef0e862a29b9ebe8a2da?sort=relevance&citationIntent=methodology>
- [7] Y. Lu, F. Yu, M. K. K. Reddy and Y. Wang, "Few-Shot Scene-Adaptive Anomaly Detection", in *Computer Vision – ECCV 2020*, 2022, pp. 125–141, doi:10.1007/978-3-030-58558-7\_8
- [8] R. Nayak, U.C. Pati and S.K. Das, "A comprehensive review on deep learning-based methods for video anomaly detection", in *Image and Vision Computing*, 2020, doi: 10.1016/j.imavis.2020.104078
- [9] G. Pang, C. Shen, L. Cao, and A.V.D. Hengel. (2021). Deep learning for anomaly detection: A review. Presented at ACM Computing Surveys. [Online]. Available: [https://www.researchgate.net/publication/342732975\\_Deep\\_Learning\\_for\\_Anomaly\\_Detection\\_A\\_Review](https://www.researchgate.net/publication/342732975_Deep_Learning_for_Anomaly_Detection_A_Review)
- [10] S. Woo, J. Park, J.Y Lee and I. S. Kweon, "CBAM: Convolutional Block Attention Module", in *Computer Vision – ECCV 2018*, 2018, pp. 3–19, doi: 10.1007/978-3-030-01234-2\_1

## REFERENCES

- [11] Saumya Jetley, N. A. Lord, N. Lee, and Philip, “Learn to Pay Attention,” OpenReview, Feb. 10, 2022. <https://openreview.net/forum?id=HyzbhfWRW> (accessed Aug. 10, 2022).

**APPENDIX**

**1. Reconstruction Results of Unseen Data using Models Trained on Class “1” Data**

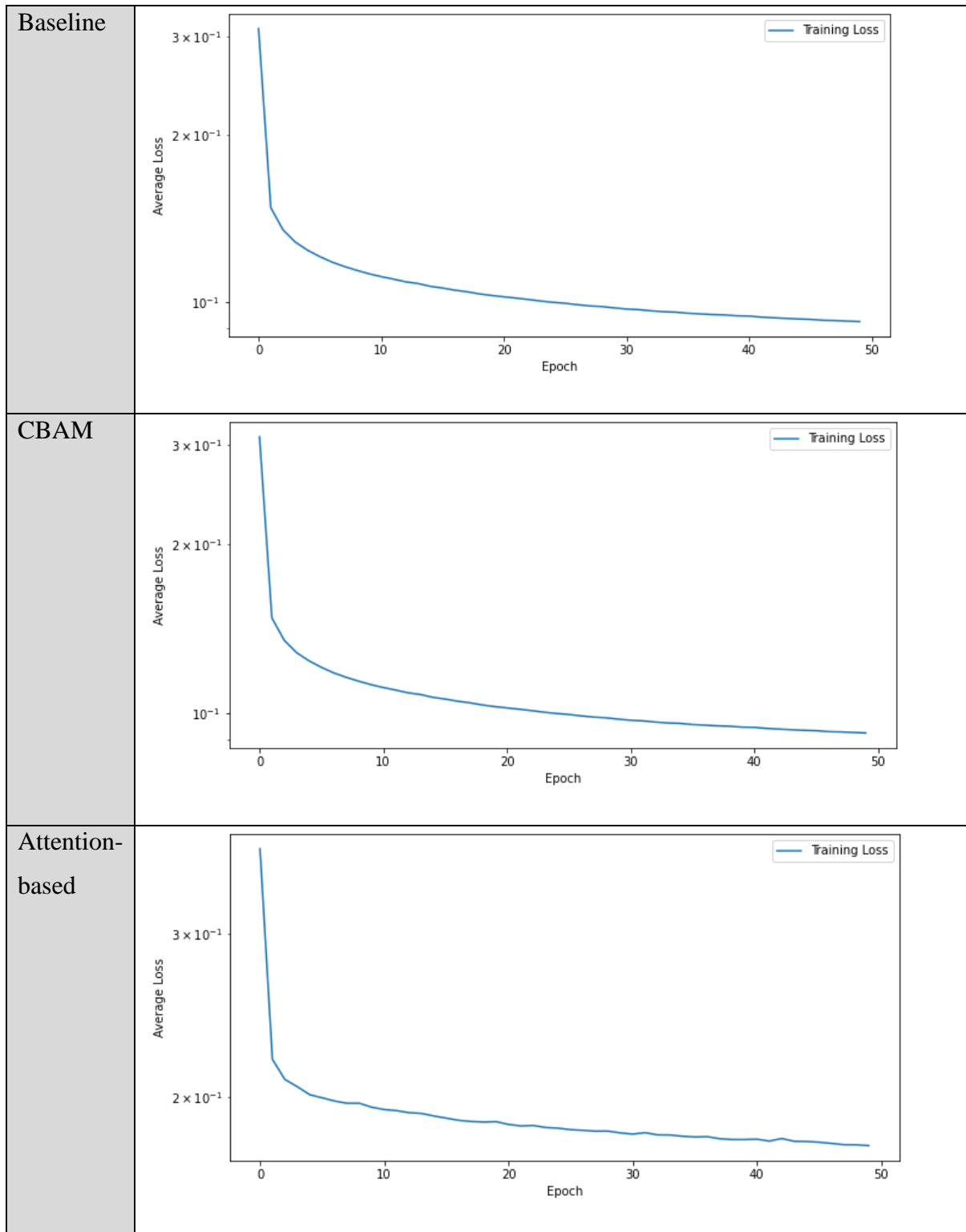
Baseline	<p style="text-align: center;">Original Images</p>  <p style="text-align: center;">Reconstructed Images</p> 
CBAM	<p style="text-align: center;">Original Images</p>  <p style="text-align: center;">Reconstructed Images</p> 
Attention-based	<p style="text-align: center;">Original Images</p>  <p style="text-align: center;">Reconstructed Images</p> 



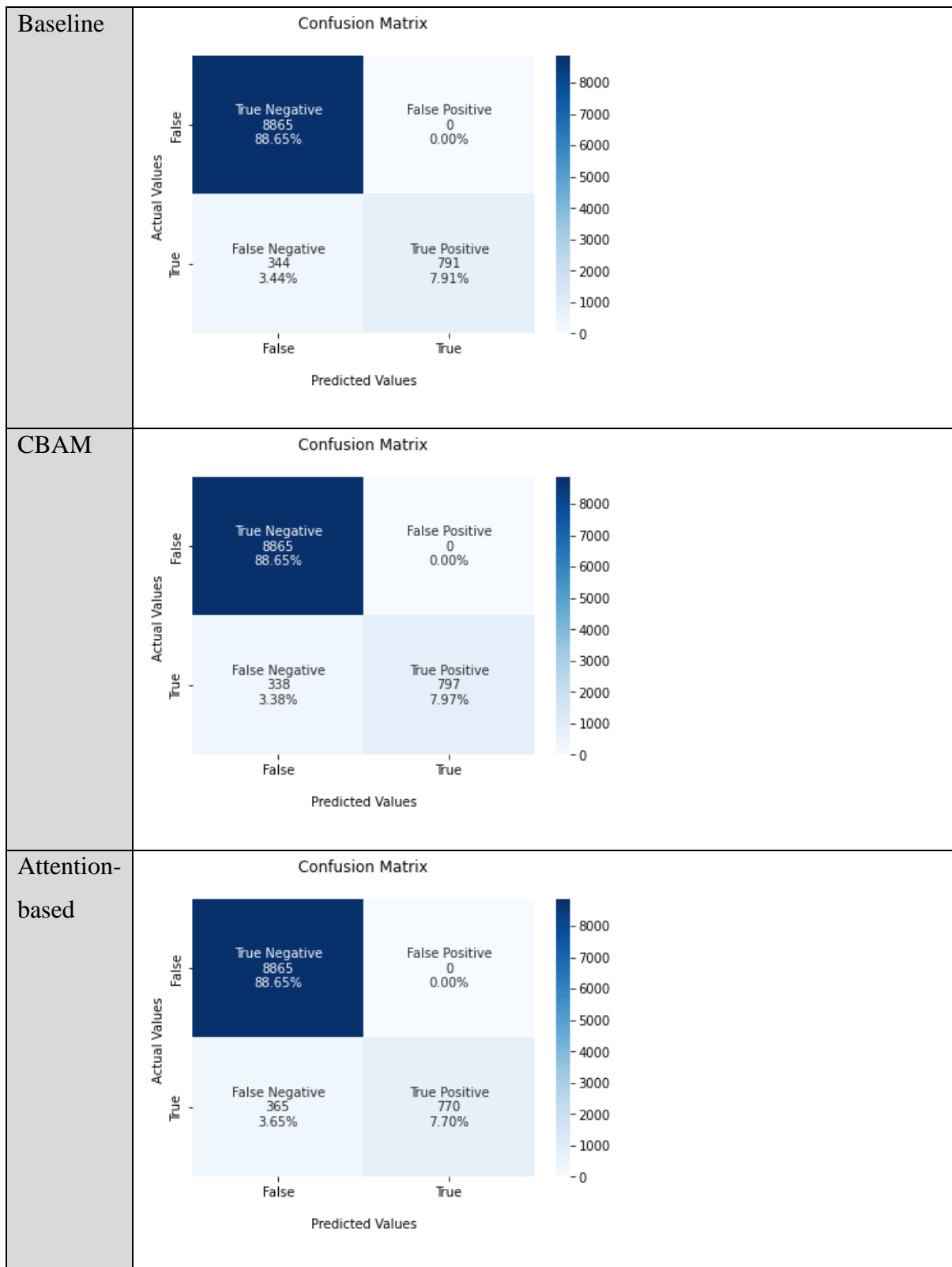
2. Reconstruction Results of Unseen Data using Models Trained on All Training Data

Baseline	<p style="text-align: center;">Original Images</p>  <p style="text-align: center;">Reconstructed Images</p> 
CBAM	<p style="text-align: center;">Original Images</p>  <p style="text-align: center;">Reconstructed Images</p> 
Attention-based	<p style="text-align: center;">Original Images</p>  <p style="text-align: center;">Reconstructed Images</p> 

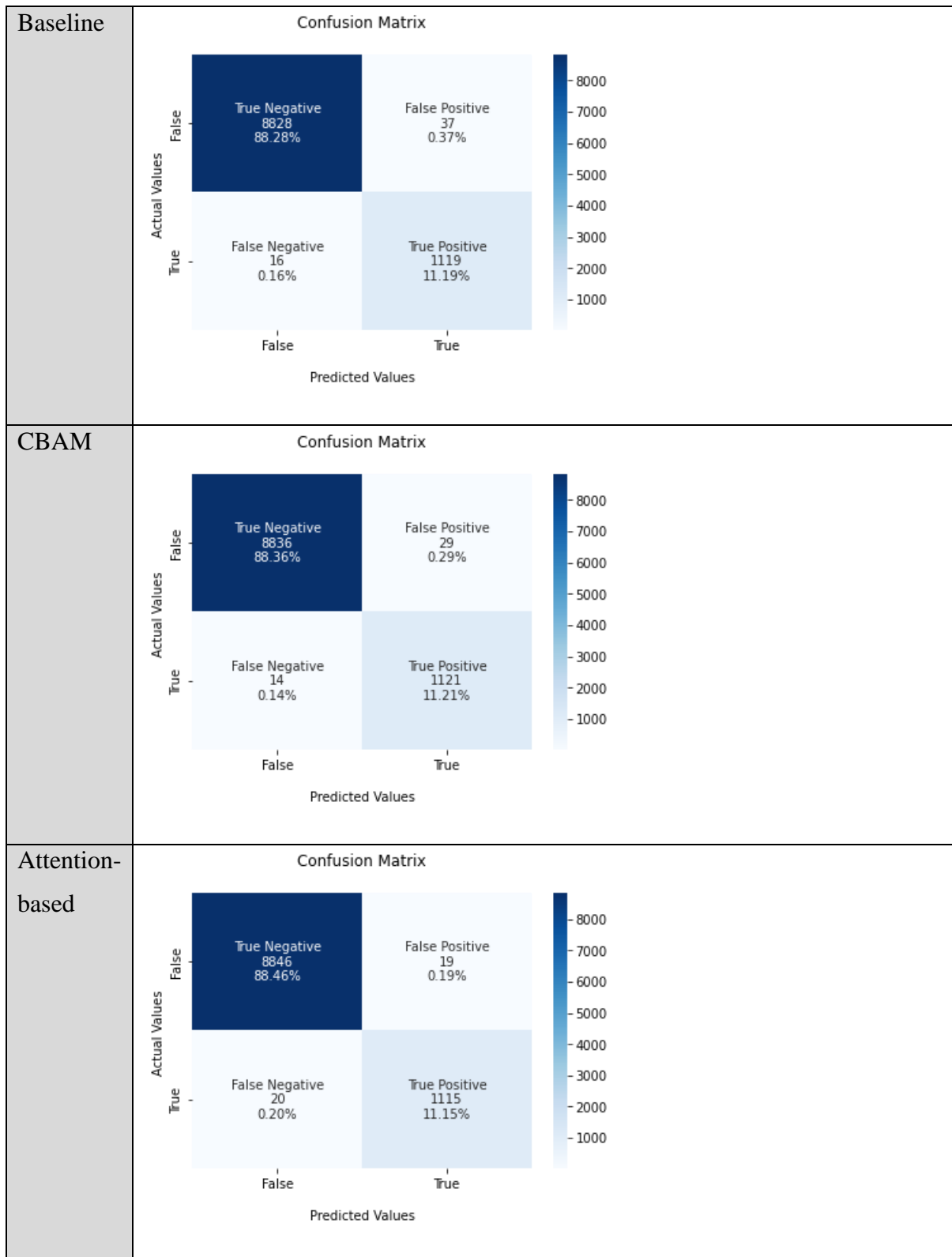
**3. Training Loss of each Model**



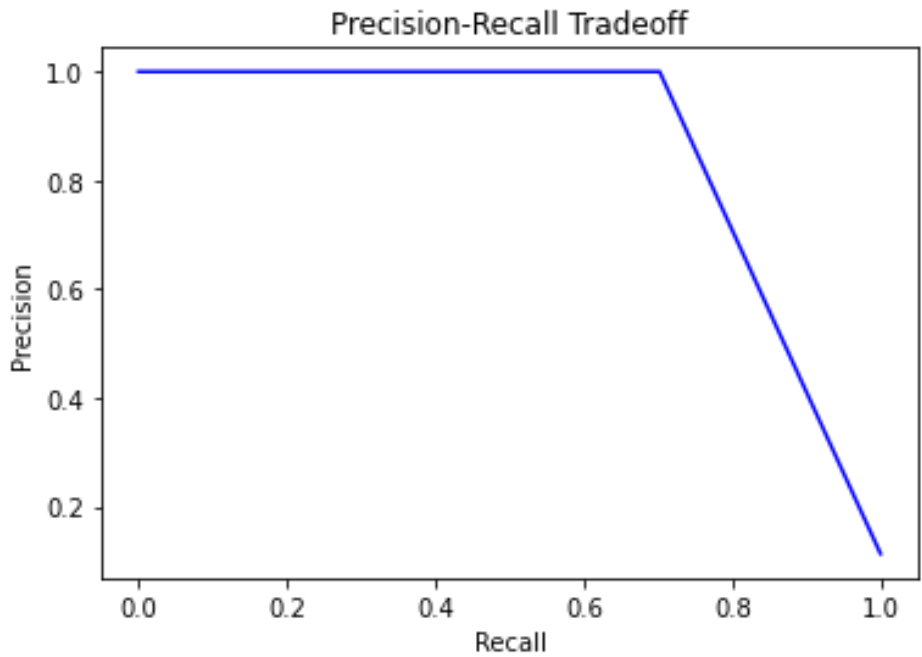
**4. Confusion Matrices of the Models using Models Trained on Class “1” Data**



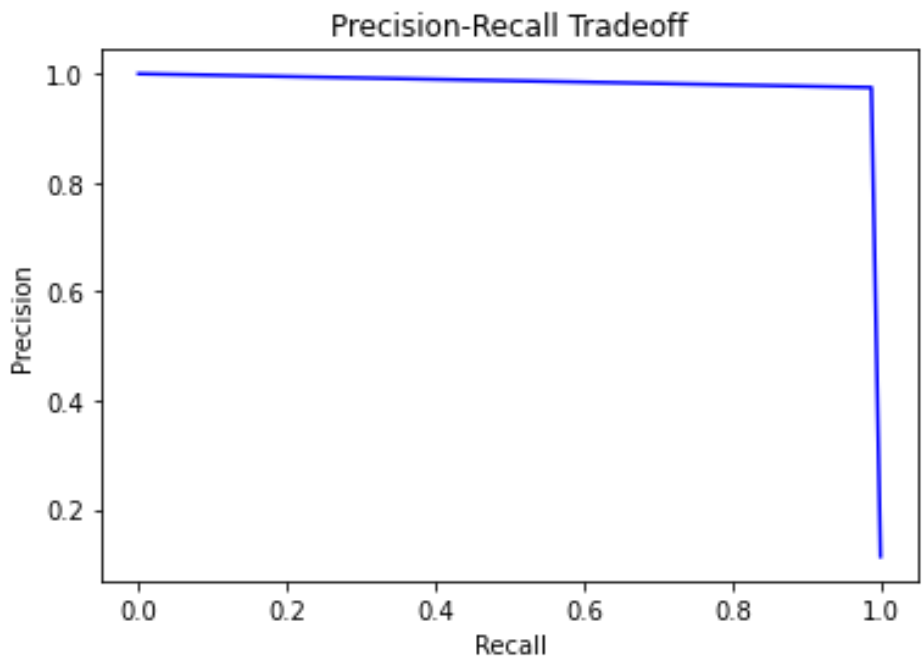
**5. Confusion Matrices of the Models using Models Trained on Class “1” Data with Buffered Threshold Values**



**6. Overall Precision-recall Curve using Models Trained on Class “1” Data**



**7. Overall Precision-recall Curve using Models using Models Trained on Class “1” Data with Buffered Threshold Values**



# FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

<b>Trimester, Year: Year 3 Sem 3</b>	<b>Study week no.: 1</b>
<b>Student Name &amp; ID: Yong Hong Long 19ACB05614</b>	
<b>Supervisor: Ts Dr Tan Hung Khoon</b>	
<b>Project Title: Anomaly Detection with Attention-based Deep Autoencoder</b>	

## 1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Since it is only the first week, I am still looking back on the things that I have done in FYP1 and prepare to start from there.

## 2. WORK TO BE DONE

Build Dong Gong's work in the first few weeks. The outline planned by supervisor is as follows:

Planning:

- Reproduce Dong Gong's result: Week 1 ~ Week 3 (3 weeks)
- Implement new idea/methods (spatial attention module): Week 4 ~ Week 11 (8 weeks)
  - Network architecture of the improved version, further breakdown the tasks in these 8 weeks
  - 5 weeks to implement the method, 3 weeks for experiments
- Write thesis (intensive writing): Week 12 ~ Week 13 (2 weeks)

## 3. PROBLEMS ENCOUNTERED

Work done too slow and hope to boost up. Plus, time compact.

## 4. SELF EVALUATION OF THE PROGRESS

Progress too slow.

Supervisor's signature

Student's signature

# FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

<b>Trimester, Year: Year 3 Sem 3</b>	<b>Study week no.: 7</b>
<b>Student Name &amp; ID: Yong Hong Long 19ACB05614</b>	
<b>Supervisor: Ts Dr Tan Hung Khoon</b>	
<b>Project Title: Anomaly Detection with Attention-based Deep Autoencoder</b>	

## 1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Investigated deeply about Dong Gong's work and found out is too hard to be done.

## 2. WORK TO BE DONE

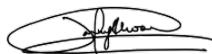
Think of new ideas to achieve the project title's objectives.

## 3. PROBLEMS ENCOUNTERED

Original idea from IIPSPW and FYP I is too hard to be achieved and plans to shrink of scope of the project.

## 4. SELF EVALUATION OF THE PROGRESS

Too slow, need pace up.



Supervisor's signature



Student's signature

# FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

<b>Trimester, Year: Year 3 Sem 3</b>	<b>Study week no.: 11</b>
<b>Student Name &amp; ID: Yong Hong Long 19ACB05614</b>	
<b>Supervisor: Ts Dr Tan Hung Khoon</b>	
<b>Project Title: Anomaly Detection with Attention-based Deep Autoencoder</b>	

## 1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Investigating attention mechanisms to be implemented in the current 2D convolutional autoencoder.

## 2. WORK TO BE DONE

Done the implementation and perform experiments on it.

## 3. PROBLEMS ENCOUNTERED

Time compact and lack of knowledge in related field thus consume many time on learning.

## 4. SELF EVALUATION OF THE PROGRESS

Too slow, need speed up.

Supervisor's signature

Student's signature



# Anomaly Detection with Attention-based Deep Autoencoder

**YONG Hong Long**

Department of Computer Science  
Supervisor: Ts Dr Tan Hung Khoon

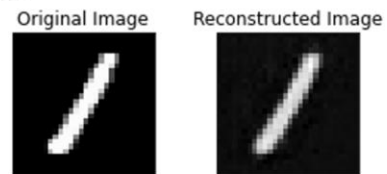
**Abstract** - Anomaly detection has become one of the most trending topics in the Information Technology domain where it detects the data that are not conformed with the expected behaviour. Autoencoder is a well-known unsupervised deep learning model that suits for performing anomaly detection. This paper shows how attention mechanisms could enhance the performance of anomaly detection.

**Methods** - To build a 2D-convolutional deep autoencoder from scratch and applies attention mechanisms to it.

Attention mechanisms involved:

- Convolutional Block Attention Module (CBAM) by [10]
- Attention-based mechanism proposed by [11]

**Results -**

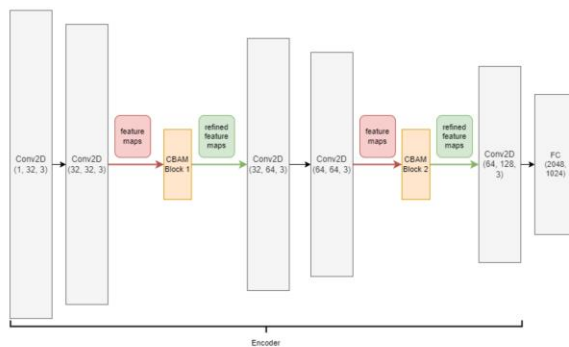


Performance boost approx. 0.6% based on model F1 Score, approx. 20% based on the reconstruction error of the image shown.

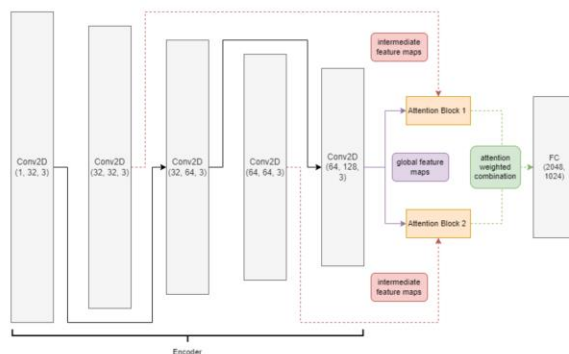
	Accuracy	Precision	Recall	F1 Score
Baseline	0.9947	0.9678	0.9859	0.9768
CBAM	0.9957	0.9748	0.9877	0.9812
Attention-based	0.9961	0.9833	0.9824	<b>0.9828</b>

**Model Architectures -**

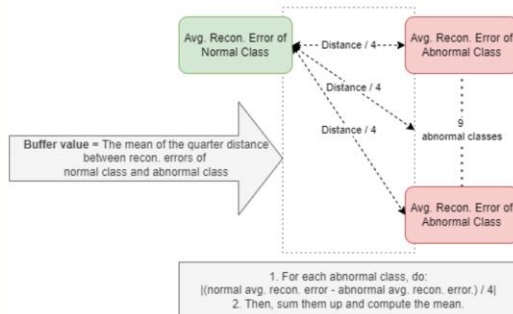
**CBAM-based Deep Autoencoder - Encoder part**



**Attention-based Deep Autoencoder - Encoder part**



**Novelty** - Introduce a method to compute a suitable anomaly detection threshold based on the distance between the average reconstruction error of normal data and anomalous data. However, it needs to be further tested on other datasets to prove its usefulness.



## PLAGIARISM CHECK RESULT

fyp2

---

ORIGINALITY REPORT

---

<b>11</b> %	<b>4</b> %	<b>10</b> %	<b>1</b> %
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

---

PRIMARY SOURCES

---

<b>1</b>	"Computer Vision – ECCV 2020", Springer Science and Business Media LLC, 2020 Publication	<b>2</b> %
<b>2</b>	deepai.org Internet Source	<b>2</b> %
<b>3</b>	"Pattern Recognition and Computer Vision", Springer Science and Business Media LLC, 2021 Publication	<b>&lt;1</b> %
<b>4</b>	"Pattern Recognition. ICPR International Workshops and Challenges", Springer Science and Business Media LLC, 2021 Publication	<b>&lt;1</b> %
<b>5</b>	Yuheng Yin, Yangang Guo, Liwei Deng, Borong Chai. "Improved PSPNet-based water shoreline detection in complex inland river scenarios", Complex & Intelligent Systems, 2022 Publication	<b>&lt;1</b> %
<b>6</b>	"Medical Image Computing and Computer Assisted Intervention – MICCAI 2018",	<b>&lt;1</b> %

Springer Science and Business Media LLC,  
2018  
Publication

---

7 [www.theseus.fi](http://www.theseus.fi) <1 %  
Internet Source

---

8 Guansong Pang, Longbing Cao, Ling Chen, Huan Liu. "Learning Representations of Ultrahigh-dimensional Data for Random Distance-based Outlier Detection", Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018  
Publication

---

9 Qilei Li, Lu Lu, Zhen Li, Wei Wu, Zheng Liu, Gwanggil Jeon, Xiaomin Yang. "Coupled GAN with Relativistic Discriminators for Infrared and Visible Images Fusion", IEEE Sensors Journal, 2019  
Publication

---

10 Yihan Bian, Xinchun Tang. "Abnormal Detection in Big Data Video with an Improved Autoencoder", Computational Intelligence and Neuroscience, 2021  
Publication

---

11 [www.semanticscholar.org](http://www.semanticscholar.org) <1 %  
Internet Source

---

12 [comengapp.com](http://comengapp.com)  
Internet Source

		<1 %
13	Jingzhi Hu, Hongliang Zhang, Boya Di, Zhu Han, H. Vincent Poor, Lingyang Song. "Meta-Material Sensor Based Internet of Things: Design, Optimization, and Implementation", IEEE Transactions on Communications, 2022 Publication	<1 %
14	Xuechao Sun, HaiYan Fu, Yuqing Cheng. "Network Intrusion Detection Based on One-dimensional Convolution Layer Autoencoders", International Conference on Frontiers of Electronics, Information and Computation Technologies, 2021 Publication	<1 %
15	ieeexplore.ieee.org Internet Source	<1 %
16	semiengineering.com Internet Source	<1 %
17	Salimjon Mahmudov. "Modeling Flow of Smart City Network: Review and Analysis", 2021 International Conference on Information Science and Communications Technologies (ICISCT), 2021 Publication	<1 %
18	Kin Gwn Lore, Devu Manikantan Shila, Lingyu Ren. "Detecting Data Integrity Attacks on	<1 %

Correlated Solar Farms Using Multi-layer Data Driven Algorithm", 2018 IEEE Conference on Communications and Network Security (CNS), 2018

Publication

19	ise.ait.ac.th Internet Source	<1 %
20	Lianghua Huang, Xin Zhao, Kaiqi Huang. "Bridging the Gap Between Detection and Tracking: A Unified Approach", 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019 Publication	<1 %
21	github.com Internet Source	<1 %
22	www.ncbi.nlm.nih.gov Internet Source	<1 %
23	"Image Analysis and Recognition", Springer Science and Business Media LLC, 2020 Publication	<1 %
24	"Medical Image Computing and Computer Assisted Intervention – MICCAI 2020", Springer Science and Business Media LLC, 2020 Publication	<1 %
25	Ahmed, Mohiuddin, Abdun Naser Mahmood, and Md. Rafiqul Islam. "A survey of anomaly	<1 %

detection techniques in financial domain",  
Future Generation Computer Systems, 2016.

Publication

26 Haoyu Ren, Darko Anicic, Thomas A. Runkler. "**TinyOL: TinyML with Online-Learning on Microcontrollers**", 2021 International Joint Conference on Neural Networks (IJCNN), 2021

<1 %

Publication

27 Jie Jia, Linjiao Xia, Pengshuo Ji, Jian Chen, Xingwei Wang. "**Access probability optimization for streaming media transmission in heterogeneous cellular networks**", Wireless Networks, 2022

<1 %

Publication

28 Jie Liu, Kechen Song, Mingzheng Feng, Yunhui Yan, Zhibiao Tu, Liu Zhu. "**Semi-supervised anomaly detection with dual prototypes autoencoder for industrial surface inspection**", Optics and Lasers in Engineering, 2021

<1 %

Publication

29 Liangquan Jia, Yawen Wang, Ying Zang, Quanfeng Li, Huanan Leng, Zhanchun Xiao, Wei Long, Linhua Jiang. "**MobileNetV3 with CBAM for Bamboo Stick Counting**", IEEE Access, 2022

<1 %

Publication

30 Qihan Jiao, Zhi Liu, Linwei Ye, Yang Wang. "**Weakly labeled fine-grained classification**"

<1 %

with hierarchy relationship of fine and coarse labels", Journal of Visual Communication and Image Representation, 2019

Publication

---

**31** Shunping Ji, Dawen Yu, Chaoyong Shen, Weile Li, Qiang Xu. "Landslide detection from an open satellite imagery and digital elevation model dataset using attention boosted convolutional neural networks", Landslides, 2020

Publication

---

**32** Sijie Zhu, Chen Chen, Waqas Sultani. "Chapter 845-1 Video Anomaly Detection for Smart Surveillance", Springer Science and Business Media LLC, 2020

Publication

---

<b>Universiti Tunku Abdul Rahman</b>			
<b>Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)</b>			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1 of 1



**FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY**

<b>Full Name(s) of Candidate(s)</b>	Yong Hong Long
<b>ID Number(s)</b>	19ACB05614
<b>Programme / Course</b>	CS
<b>Title of Final Year Project</b>	Anomaly Detection with Attention-based Deep Autoencoder

<b>Similarity</b>	<b>Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)</b>
<b>Overall similarity index:</b> <u>  11  </u> %  <b>Similarity by source</b> Internet Sources: <u>    4    </u> % Publications: <u>    10   </u> % Student Papers: <u>    1    </u> %	
<b>Number of individual sources listed of more than 3% similarity:</b> <u>  0  </u>	
<b>Parameters of originality required and limits approved by UTAR are as Follows:</b> (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

***Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.***

\_\_\_\_\_  
Signature of Supervisor

Name:   Tan Hung Khoon  

Date:   12/9/2022  

-

\_\_\_\_\_  
Signature of Co-Supervisor

Name:                   -                  

Date:                   -





## UNIVERSITI TUNKU ABDUL RAHMAN

### FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

#### CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	19ACB05614
Student Name	Yong Hong Long
Supervisor Name	Ts Dr Tan Hung Khoon

TICK (✓)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
x	Front Plastic Cover (for hardcopy)
✓	Title Page
✓	Signed Report Status Declaration Form
✓	Signed FYP Thesis Submission Form
✓	Signed form of the Declaration of Originality
✓	Acknowledgement
✓	Abstract
✓	Table of Contents
✓	List of Figures (if applicable)
✓	List of Tables (if applicable)
x	List of Symbols (if applicable)
✓	List of Abbreviations (if applicable)
✓	Chapters / Content
✓	Bibliography (or References)
✓	All references in bibliography are cited in the thesis, especially in the chapter of literature review
✓	Appendices (if applicable)
✓	Weekly Log
✓	Poster
✓	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
✓	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

\*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

(Signature of Student)

Date: 9 September 2022