Master thesis

Artificial Intelligence

Radboud University

Implementing continual learning in autoencoders for Network Intrusion Detection Systems in a practical use case

Author:
Hidde Jansen
S1006034
hidde.jansen@ru.nl

External supervisor:
Julik Keijer, MSc
Northwave Cyber Security
julik.keijer@northwave-cybersecurity.com

Internal supervisor:
Prof. J.H.P. Kwisthout
Donders Centre for Cognition
j.kwisthout@donders.ru.nl

June 27, 2024

Abstract

To detect intrusions in and therefore protect a computer network, network intrusion detection systems (NIDS) are broadly used software solutions. Machine learning has shown potential to be used in these systems. However, implementing such a solution has drawbacks. Primarily, anomaly-based NIDS can suffer from catastrophic forgetting and have difficulties adapting to changing environments. As a solution, academics have proposed the concept of continual learning. In this thesis, possibilities are explored to implement continual learning in NIDS. In particular, experience replay will be used in an unsupervised autoencoder NIDS.

It first validates the approach on academic datasets, CICIDS-2017 and Kyoto2006, and then validates the findings on data collected from Northwave Cyber Security. For CICIDS-2017 and real data, no improvement in AUC has been observed. On Kyoto2006, experience replay improved the AUC for unseen data from 0.6024 to 0.6806, which is an improvement of 7.84%. While this shows potential, more work needs to be done to conclusively evaluate the performance. This work serves as exploratory study to an implementation of continual learning in the context of anomaly-based unsupervised NIDS.

Contents

Chapter 1

Introduction

The internet and related digital technologies are an essential part of our modern-day society. Online transactions have become the norm for banking and cloud- and social computing are foundations of our society [1].

Subsequently, the number of cyber-attacks worldwide has increased [2]. Because of the potential consequences of an attack, protecting information and maintaining security of digital infrastructure is of vital importance. One of the tools that can be used in this regard is an intrusion detection system (IDS), which can detect and classify intrusions in a critical infrastructure. IDS is based on the assumption that intrusions or attacks have a distinct pattern of data that separates it from legitimate use [2]. This allows an IDS to detect attacks such as port scans, DoS attacks or malware. In general, an IDS is applied to log data. This log data can either contain events happening on a specific host (Host-Based IDS), or it can gather logs on a network level (Network-based IDS). The two types of IDS are also commonly abbreviated to HIDS and NIDS respectively. In the case of HIDS, a detection can for example occur by a change to a certain important file. In the case of NIDS, an attack can for example be detected by a malformed packet.

Machine learning has recently been shown to be a viable tool to analyze the detections [3]. In particular, machine learning is able to detect anomalies in data [3]. An important challenge within the field of machine learning is that of *catastrophic forgetting*. While machine learning, in particular deep neural networks, is very successful for solving different tasks, it has a tendency to lose previously learned information when a network is trained sequentially on multiple tasks [4]. This is contrary to human behavior. An important aspect of human intelligence is the ability to learn from non-stationary data in an incremental way, while not forgetting the knowledge it has gathered when training for previous tasks [4, 5]. In particular, learning data incrementally from non-stationary streams is also called *online learning*. The concept of online learning while not forgetting knowledge is called *continual learning* [4, 5] and this is still a relatively underexplored area in research [4].

Following up on previous research performed at Northwave Cyber Security [6], the research presented in this thesis focuses on NIDS. The research aims to address some of the challenges that arise when implementing a continual learning autoencoder solution. Furthermore, since the field of continual learning and autoencoders is not yet widely explored in the literature [7], it will also aim to contribute to the general state of the art of this field.

In order to achieve the research goal, the following research question has been determined: ***RQ**: Can the incorporation of continual learning enhance the efficacy and efficiency of autoencoder-based intrusion detection systems in real-time network monitoring?*

To make a valid comparison to existing literature, a subquestion has been defined that addresses the usage of continual learning on academic datasets: ***RQ1****: How can continual learning improve the performance of autoencoders on academic anomaly-based NIDS datasets?*.

Finally, in collaboration with Northwave Cyber Security, this thesis has the unique opportunity to validate findings on data from real networks. To address this contribution, the following subquestion has been defined: ***RQ2****: How can continual learning improve the performance of autoencoders on anomaly-based NIDS using real network data?*

Chapter 2 serves as an introduction to important topics and concepts, as prerequisites for understanding the studies. In this chapter, also relevant examples from the literature are discussed and finally an introduction is done to the available used datasets.
In chapter 3, the approach used for creating the model and assessing the performance are presented.
In chapter 4, the corresponding results from the experiments are presented.
Finally, in chapter 5, the research questions are answered. This chapter also houses an assessment of the used methodology and discusses limitations and possible directions for future research.

The research in this paper will have the following scientific contributions:

- An experimental pipeline where anomaly-based NIDS using autoencoders can be evaluated.
- An empirical study assessing the feasibility of continual learning on anomaly-based NIDS when using academic datasets.
- An empirical study assessing the feasibility of continual learning on anomaly-based NIDS when using data from real networks.
- A contribution to the state of the art of continual learning and in particular, continual learning within the context of anomaly-based NIDS.

Chapter 2

Literature Review

This chapter will briefly introduce the topics of autoencoders, continual learning and the application of these techniques in network intrusion detection systems. It serves as prerequisites for understanding the studies. Furthermore, this chapter introduces available benchmark datasets.

2.1 Background and Theoretical Framework

Figure 2.1 serves as overview where the different concepts are placed.

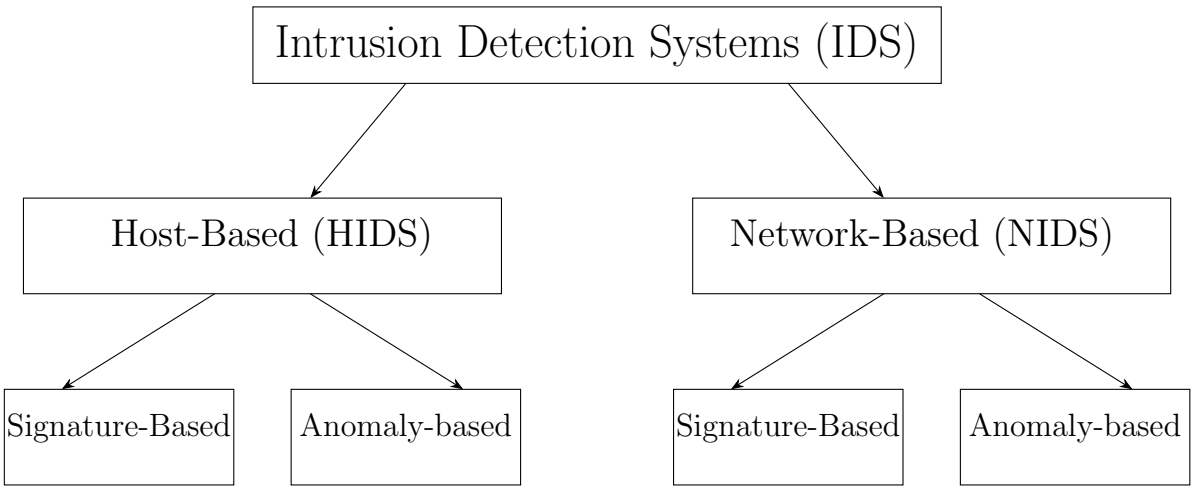


Figure 2.1: Overview of IDS

2.1.1 The context of network-based intrusion detection

In order to understand the intuition behind the research in this thesis, first some background will be discussed on the importance of Intrusion Detection systems (IDS). As discussed in the Introduction, IDS is used to detect intrusions in a critical infrastructure, where network-based IDS (NIDS) acts on a network level. Alternatively, host-based IDS (HIDS) can be used, which monitors a (set of) specific host(s) [8]. This type of IDS works on the host and monitors different types of logs, for example file and policy modifications [8]. Since this thesis focuses on NIDS, HIDS is not further explained. NIDS work primarily on network traffic logs, which contain information about for example TCP/IP packets and connections.

Found intrusions can be divided into two primary groups: external and internal intrusions. External intrusions refers to unauthorized users having access to systems or information and internal to users that already have (restricted) access [2].

Signature-based or anomaly-based

In order to perform the actual detection, NIDS can use both a signature-based approach as well as an anomaly-based approach [8]. With a signature-based approach, a signature or pattern will be observed for incoming data which is compared to a database containing known signatures of attacks. When the patterns match, there is almost a guarantee that the incoming traffic is attack traffic [2]. While this approach works very reliably for known attacks, it is difficult to develop good signatures for new and emerging threats. One key aspect in this scenario is keeping the number of false-negatives (attacks that are incorrectly not identified as such) and false-positives (benign traffic incorrectly classified as attack) low [6, 2].

The challenge as presented in the previous paragraph is addressed by the concept of anomaly-based NIDS. In this approach, a baseline is captured that covers normal traffic. When a deviation from the baseline is captured, it is concluded that there is attack traffic. In theory, this seems a very good approach since it is able to detect attacks for which no signature has been defined yet. However, in practice it turns out that there are multiple challenges when implementing such a solution. Key examples of these challenges include determining the baseline itself and determining the right threshold for concluding that traffic is indeed attack traffic [6, 2]. This anomaly-based approach has shown potential for applying machine learning and as such will be further investigated in this thesis.

Different methods of detecting anomalies

Anomaly-based NIDS can use multiple methods of detecting anomalies. Primarily, these can be divided into statistical, machine learning and hybrid methods, which combine both statistical and machine learning methods [9].

Statistical approaches such as the z-score are used to statistically identify outliers or anomalies within the data. They have the benefit of being based on a statistical foundation, which makes them easily interpretable. A downside of these methods is that they can only identify simple anomalies, and have difficulties when handling high-dimensional or otherwise complex data [9].

Machine learning methods such as neural networks or clustering algorithms such as DBSCAN have the benefit of being able to identify complex patterns in the data and the ability to adapt to changing environments. As a downside, they often work as a "black box", meaning that they are not interpretable. Furthermore, there is a requirement for (large) datasets and resources [9].

Finally, Hybrid methods use a combination of both statistical and machine learning features. By using this combination, in theory, the program may adapt more to changing environments as compared to plain statistical approaches, but still allows some interpretability [9].

Since the work in this thesis focuses on autoencoders, a form of machine learning, this section is explained in greater detail below.

2.2 Machine learning in Anomaly Detection

Machine learning can happen in a supervised or unsupervised way. The difference is their dependency on labeled data.

2.3 Supervised learning

Supervised learning is a method of training neural networks by using labeled data. This implies that for every data point that is available, there is a label present that tells whether the data point is attack or normal data. Most research into anomaly detection using machine learning has been done in this discipline and showed good results in terms of performance [10]. A drawback of using this method is that there needs to be enough correctly labeled data available. Furthermore, supervised methods may not capture drifts in the data and may therefore be less resilient to new and emerging threats [9]. This property is where the potential for unsupervised anomaly detection arises.

2.4 Unsupervised learning

Unsupervised learning methods do not rely on labeled data, and work solely on properties of the data itself. The greatest benefit of this type of learning is that there is no necessity for labeled data by human experts. Furthermore, because of this property, the network should be more resilient to new and emerging threats. A drawback of using unsupervised methods is that they tend to have a higher False Positive Rate [9, 6]. This means that a benign data point is incorrectly flagged as malicious in more cases as compared to supervised learning.

2.4.1 Autoencoders

Autoencoders are a subform of unsupervised learning. In particular, autoencoders are a reconstruction-based algorithm. This class of algorithms aims to learn a reconstruction of their input data with the lowest possible loss [11]. A standard autoencoder consists of three main parts: an encoder, a decoder and a latent feature representation ("*bottleneck*") in between. It can visually be represented as in figure 2.2.

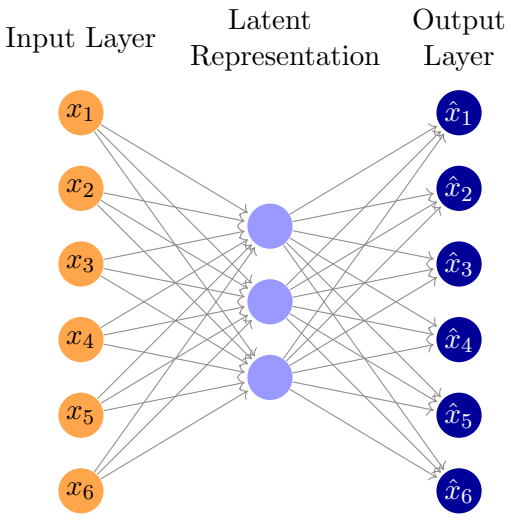


Figure 2.2: Standard representation of an autoencoder [12]

As can be seen in figure 2.2, a standard autoencoder consists of only one hidden layer (the latent representation). The autoencoder is trained using a set of normal traffic, which does not contain any attacks. During training, the input data is being reconstructed by the autoencoder and the reconstruction error is optimized. When deployed, presenting the autoencoder with an attack data sample should result in obtaining a higher reconstruction error.
An expansion of the normal autoencoder is the variational autoencoder (VAE), which is a type of generative model that not only tries to reconstruct the input, but by doing that also learns a

low-dimensional data distribution. As such, it is able to generate samples from this distribution. Anomalies can then be identified when the distribution of the data significantly differs from the learnt distribution [9].
For both types of autoencoders it holds that there are multiple parameters that can be optimzized. For example, more hidden layers could be added, which creates the so-called *deep* autoencoder. Also the number of hidden units in the latent representations can be varied. These parameters can be considered *hyperparameters*.

Both types of autoencoders are in principle able to learn complex patterns. However, both also require a large enough dataset containing normal traffic, i.e. traffic that does not have any attacks in it. Furthermore, it may be difficult to adapt to changing patterns in the data. To mitigate this, continual learning has been proposed by academics.

2.4.2 Continual Learning

Neural networks typically are designed to incrementally adapt to controlled data samples, which differs from the way humans normally process new information. In the real world, humans must deal with uncertainty and maintain knowledge from past events; whilst also adapting to a changing environment [13, 14]. Furthermore, neural networks may suffer from a phenomenon called catastrophic forgetting. This phenomenon implies that previously acquired knowledge is forgotten in favor of newly acquired knowledge [13]. The balance between being flexible to new data and robust to not forgetting old data is called the stability-plasticity dilemma [15]. Recently, the process of lifelong learning, or continual learning, is also being applied to machine learning [13].

2.5 Continual learning in Machine Learning

The goal of continual learning in machine learning is mainly two-fold [13]: implementing online learning where the model can adapt to changing environments and mitigating catastrophic forgetting, such that the model retains all knowledge available. Implementing continual learning in machine learning is not straightforward, and multiple methods have been proposed. This includes regularization, replay methods and dynamic architectures [13].

2.5.1 Regularization

Regularization is a method of applying a certain penalty to the learning optimization of a network. A main example of this method is Elastic Weight Consolidation (EWC). Using the technique adds a quadratic penalty to the loss in order to mitigate catastrophic forgetting [13]. This technique is mainly used in supervised and in reinforcement learning scenarios.

2.5.2 Replay methods

Replay methods are primarily used in unsupervised settings [13]. When implementing these methods, data samples are either generated (generative replay) or stored and later replayed (experience replay [13]). This approach is perhaps the most intuitive. The two types of replay (generative versus experience) differ in the form of which samples are chosen. In the case of generative replay, the neural network aims to learn the distribution of the data. As such, this learnt distribution can be used to generate samples from. In the case of experience replay, no distribution is learnt but old samples are explicitly stored to replay later in the training pipeline [13].

2.5.3 Dynamic architectures

Dynamic architectures are neural network architectures that can change based on the task at hand, instead of relying on a fixed architecture [13].

Progressive Networks

A concept that is coined by Rusu et al [16] is that of progressive networks. This is a technique for reinforcement learning, keeping a pool of trained networks for different tasks. When a new task is presented, only the between-network weights are learned and to prevent catastrophic forgetting, the within-network weights are left untouched [16, 13]. This architecture delivered promising results but when a large number of tasks are presented, this creates very complex networks [16, 13].

A similar approach is proposed by Turner et al [17]. In their work, the authors propose a solution where multiple independent sub-networks are trained and are formed into one tree-like structure. The unique contribution of their work is that every sub-network functions independently for and the network is able to re-organise itself based on new information.

2.6 Anomaly-Based NIDS and Continual Learning

The concept of using continual learning within anomaly-based NIDS is not readily described in literature, in particular when an autoencoder is used. One of the first works that touched upon the concept is that by Wiewel and Yang [18]. In 2019, these authors described an approach using a VAE, which is used as generative replay (GR). Their approach is also compared to a theoretical upper bound in the form of EWC. Their conclusion is that the performance of GR is similar compared to using EWC, but instead does not use any external components besides the VAE itself.

In 2022, work by Prasath et al became available [19]. In their work, continual learning techniques from the image domain have been applied for the first time on this scale to intrusion detection systems. In this work, they also show the potential for experience replay and show that the concept of continual learning is promising to also apply on related datasets. Their main conclusion is that the implementation of continual learning improves accuracy and achieves a lower FPR, while also being able to learn new patterns in the data.

Dominant work in the field is coming from Faber et al. [20, 21], who is experimenting with continual learning on (un)supervised methods. One of the main contributions of this author is VLAD [21], which is a task-agnostic VAE that is suited for a variety of tasks, including anomaly detection and implements continual learning.

2.7 Evaluation metrics and datasets

For the evaluation of performance, a few metrics are commonly used in literature. In particular, precision, recall and F1 score are used [21]. Furthermore, the True Positive Rate and False Positive Rate are important, since a low True Positive Rate means that alarms may be missed whereas a high False Positive Rate means that an alert is triggered when there is in fact not a problem [6].

Apart from the methods mentioned above, the area under the curve (AUC-ROC) is commonly

being used in the literature. This metric provides a solid way of comparing methods within this context, since it is a threshold-free method. This means, that no threshold has to be determined to assess whether a data point is an anomaly. As described earlier in this work, determining the right threshold is a difficult topic. Therefore, in line with [22], the ROC curve is a suitable approach to assess performance.

2.7.1 Benchmark Datasets

Alshamy et al [23] have developed a detailed overview of publicly available datasets that can be used for benchmarking NIDS algorithms. Their summary figure is in table 2.1.

Dataset	Year	Modern Attacks	Duration of Data Collected	Number of Features	Number of Attacks
KDD	1999	No	7 weeks	41	4
NSL-KDD	2009	No	16 hours and 15 hours	41	4
KYOTO 2006+	2009	No	3 years	24	3
ISCX2012	2012	Yes	7 days	14	7
UNSW-NB15	2015	Yes	16 hours and 15 hours	49	9
CIDDS-001	2017	Yes	4 weeks	14	5
CICIDS-2017	2017	Yes	5 days	86	14
CSE-CIC-IDS2018	2018	Yes	10 days	80	7

Table 2.1: Comparison of benchmark datasets for NIDS [23]

The work [23] by these authors is from 2020. However, in a preliminary literature search for this work where the keywords "NIDS dataset" and "Anomaly detection dataset" were used, no significantly new datasets have been found. According to the authors in [23], most works use the KDD, NSL-KDD and UNSW-NB15 dataset. From the literature research for the present work, it also followed that the CICIDS-2017 is commongly being used. Interestingly, the KDD dataset from 1999 is still being used in modern literature. What furthermore appears from the summary is that the different datasets differ in terms of features available, attacks available and especially the time period in which data has been collected. Furthermore, the different datasets present the data in different formats. Mostly, the data consists of network flows, either in pcap files or in csv format.

2.8 Conclusion

The research area, in which this work is presented, is relatively small and relatively new. This means, that there is not much literature available on the topic of unsupervised NIDS using autoencoders and in particular with the addition of continual learning.

While the work by Faber et al. [20, 21] is limited to the use of synthetic datasets, the work in this thesis will be evaluated also on data from a commercial SOC, which is a unique contribution to the state of the art. Furthermore, the proposed continual learning approach in the work by Faber et al [21] is very elaborate, whereas the work in this thesis tries to make the approach as straightforward as possible. Finally, as Faber et al indicate, there is very little literature available on this exact topic. Therefore, the work presented in this thesis will contribute to the general state of the art of continual learning in unsupervised anomaly-based NIDS.

From the analyzed literature, hereafter the most important take-away messages are written:

1. There is no conclusive answer whether continual learning actively contributes to the performance of unsupervised NIDS using autoencoders.

2. Autoencoders are a suitable tool for performing unsupervised NIDS.

3. An important method of implementing continual learning is by implementing experience replay and online learning.

4. The most used metric in literature for assessing the performance of continual learning is the area under the ROC curve (AUROC).

5. There are multiple datasets available, but some of them are significantly dated.

Chapter 3

Methodology

The research presented in this thesis consists of multiple subquestions, where the methodolody will be discussed for each research question individually. To understand the position of this research within the academic field, a literature review has been performed in chapter 2. The methodology for **RQ1** and **RQ2** will be discussed hereafter. For structure, the methodology for each research question is treated in a different section.

3.1 RQ1: How can continual learning improve the performance of autoencoders on academic anomaly-based NIDS datasets?

To answer this research question, a simple autoencoder will be built and evaluated on datasets that are available within the academic community. Within this section, the design choices for the experiments will be expanded on.

3.1.1 Dataset

To answer the research question, two goals have been set. First, an experiment is designed using a relatively recent dataset that can be compared to literature. Second, an experiment is designed using a dataset that spans a large amount of time, to assess the performance of continual learning on adapting to changing environments. From the datasets that are available and described in chapter 2, no dataset has been identified that satisfies both criteria. Therefore, two experiments have been designed. For the first experiment, CICIDS-2017 is used [24]. This dataset is from 2017, which is relatively recent and contains recent attacks. Furthermore, it is used in various scientific literature which allows for validation. The collected data consists of 5 working days of traffic where Monday is the only day with benign traffic. The remaining 4 working days contain attack traffic. It contains data using all common protocols such as HTTPS, SSH and FTP and has multiple attack types such as DoS and Scan attacks.

To assess the second objective, the Kyoto2006 dataset [25] is used. This dataset contains data that is collected between 2006 and 2009. The authors have later also published a follow-up dataset that contains data until 2015. The Kyoto2006 dataset consists also of real network traffic, but does not have a distinct day of only benign traffic. The benign and attack traffic are mixed in the dataset.

Both datasets consist of network flows and are published in CSV format.

3.1.2 Feature selection

CICIDS-2017 The dataset consists of network flows with an extensive number of features available per flow. Because of the scope of this research, features have been selected according to previous work done by Verkerken et al. [26]. The total number of features therefore comes to 66, excluding the label. The remaining features are all numeric features.

Kyoto2006 In line with the approach for the CICIDS-2017 dataset, features have been selected based on similar work from the literature. In line with the work by Protic et al [27], 9 features have been selected for the Kyoto2006 dataset as baseline, excluding the label. The numerical feature *duration* is not included by the mentioned authors. However, from personal experience, it is expected that this feature might have some contribution. Therefore, a correlation matrix has been generated. This matrix shows the contribution of the individual features to the label. The results can be seen as a feature importance plot in figure 3.1.

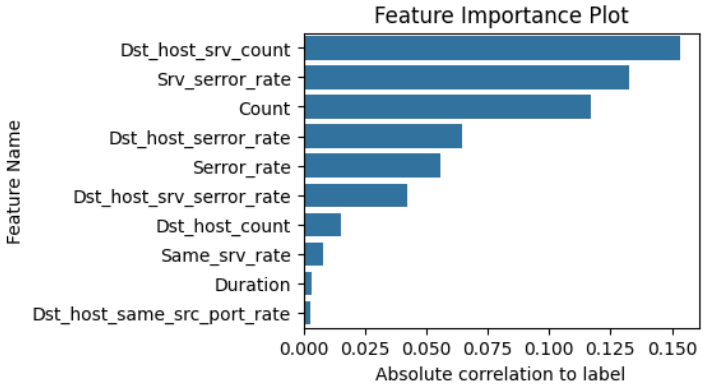


Figure 3.1: Feature importance plot for Kyoto2006

From the figure it followed that *Duration* has a small contribution to the label. Since this contribution is higher than *Dst_host_same_src_port_rate* and that feature has been included by Protic et al [27], *Duration* is also included. Therefore, the total number of features is 10, excluding the label.

3.1.3 Preprocessing

The preprocessing pipeline of the experiment can be described by the following: First, the relevant features are extracted from the dataset. Then, rows with infinity values and/or nan values are dropped from the dataset. Finally, the labels are extracted and stored separately, after which the labels are dropped from the original dataset. For the training set of CICIDS-2017, these steps are explicitly described in table 3.1. For the training set of Kyoto2006, these steps are described in table 3.2.

Description	**Shape**
Shape before any preprocessing	(529918, 79)
Shape after removing nan	(529481, 79)
Shape after selecting features	(529481, 66)

Table 3.1: Data shapes in preprocessing for CICIDS-2017 training data

From this table, it follows that there are 437 rows containing infinity or nan values in CIDIDS-2017. These rows have been dropped, because it is only a very small amount of the

dataset. In Kyoto2006, there are no infinity or nan values. Therefore the shape of Kyoto2006 is only influenced by the feature selection:

Description	**Shape**
Shape before any preprocessing	(249956, 24)
Shape after removing nan	(249956, 24)
Shape after selecting features	(249956, 10)

Table 3.2: Data shapes in preprocessing for Kyoto2006 training data

An important part of the preprocessing pipeline is feature scaling [28]. When this part of the pipeline is not executed and unscaled features are used, this can result in a slow or even failed learning process [28]. In line with the conclusions in [28], determining the right methodology and type of scaling is difficult and there is no universal or straightforward method to determine the best approach. Multiple scalers from scikit-learn [29] have been tested in this research, including MinMaxScaling, QuantileTransformers, RobustScalers and StandardScaler.

CICIDS-2017 In line with the work by Verkerken et al [26], QuantileTransformer has been chosen as scaling method for CICIDS-2017.

Kyoto2006 For the Kyoto2006 dataset, experiments were performed to determine the optimal scaling method. Keeping the data and other optimization parameters equal, the training and validation loss have been observed for the aforementioned four different scalers. In figure 3.2, the resulting plots are shown.

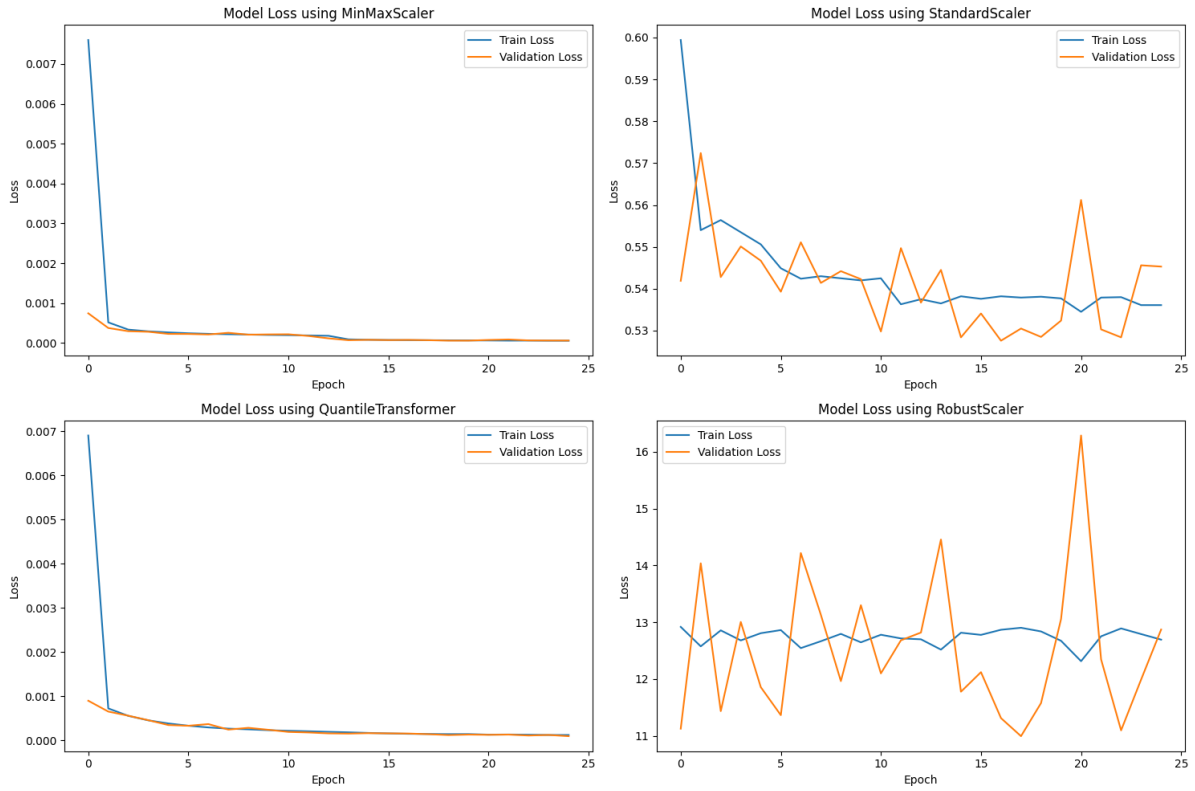


Figure 3.2: Training losses with different scalers

From the plots it can be observed that not all scalers perform equal. Starting from the

top-left plot, MinMaxScaler shows a smooth line for both the training and validation loss. MinMaxScaler transforms the data to a fixed range, usually between 0 and 1, keeping the relation between data points [30]. This makes it suitable for machine learning approaches. In the bottom-left plot, QuantileTransformer shows a smooth line for both the training and validation loss. QuantileTransformer transforms each feature to a uniform distribution [31]. This makes it very robust and usable in this context. However, the correlations between features are distorted [31]. In the top-right plot, StandardScaler does not show smooth lines for both the training and validation loss. StandardScaler transforms features to a mean of 0 and a standard deviation of 1, which makes it very sensitive to outliers. Furthermore, it assumes a normal distribution [30]. Finally, in the bottom-right plot, RobustScaler also does not show smooth lines for both the training and validation loss. RobustScaler removes the influence of outliers, but still relies on a normal distribution [30]. Since a normal distribution cannot be assumed and the relations between features are expected to be important, MinMaxScaler has been chosen as scaler for this experiment.

Finally, since the Kyoto2006 dataset does not provide the benign traffic separately from the attack traffic, the benign traffic is extracted and stored as a separate dataset.

For both datasets, the scalers are stored separately and only fitted on the training data. This ensures that the scaler only learns the distribution of the training data, and does not learn the distribution of unseen data. Finally, a 80/20 train/test split is created in order to validate the training performance.

3.1.4 Autoencoder

For creating the autoencoders, Keras [32] has been used.

CICIDS-2017 To focus on the implementation of continual learning and not the network design itself, the network architecture as proposed by Verkerken et al [26] has been copied. This also allows for validation of their work. Where this work differs from the work by Verkerken et al [26] is the lack of regularization, to keep the network as simple as possible. This resulted in a network consisting of 5 hidden layers with 56,54,51,54 and 56 hidden units. No further research has been done to explore variations of this architecture. ReLu activation has been used in the hidden layers and sigmoid in the output layer.

Kyoto2006 To start with designing the network architecture, the number of hidden layers is the same as for the experiment on CICIDS-2017. Some experiments have been done with the number of hidden units, which can be seen in figure 3.3.

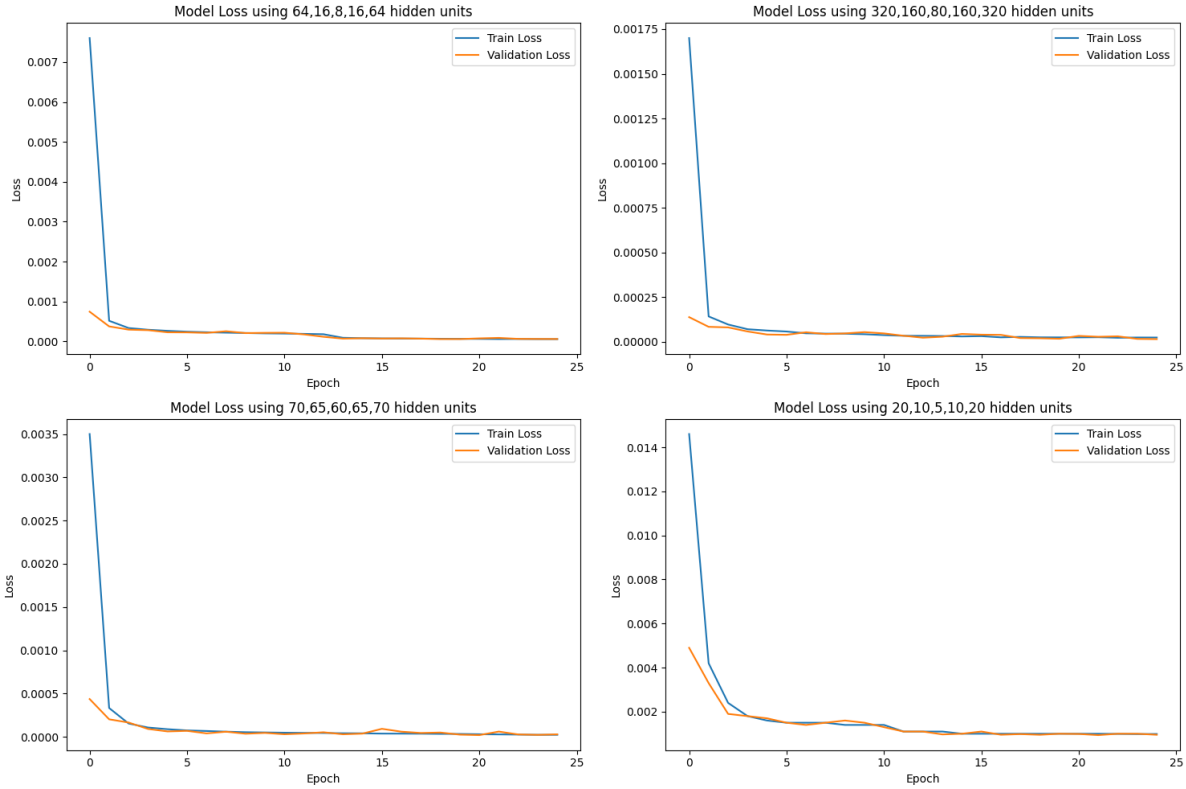


Figure 3.3: Experiments using different structures

The structure 64, 16, 8, 16, 64 has been observed as a suitable architecture, without overfitting on the validation split. In this architecture, there is more variation between the individual layers as compared to the setup for CICIDS-2017. From the downleft plot it followed that an architecture with less variation between the layers performed worse.

3.1.5 Continual learning

The approach taken is an unsupervised learning approach, for which a replay approach is the most suitable, as has been determined in chapter 2. As compared to, for example, the approach by Faber et al [21], a more simple approach is thought of as baseline. The general design consists of a replay buffer and a replay method during the training phase. This approach is inspired by the work of Rolnick et al [33], where it is concluded that a simple buffer provided promising results. The buffer is implemented by means of a simple class that contains an array to store previously seen examples. The class is complemented by helper functions to add and retrieve experiences. Furthermore, the design of the class is in such a way that there is a limited capacity, and the oldest sample is overwritten when the buffer is full.

As a proof of concept, the hyperparameters with respect to the buffer are hand-picked based on intuition. To determine the capacity of the buffer, the base value is picked by intuition based on the size of the dataset. The capacity has been validated using simple experiments, the results of which can be seen in figure 3.4.

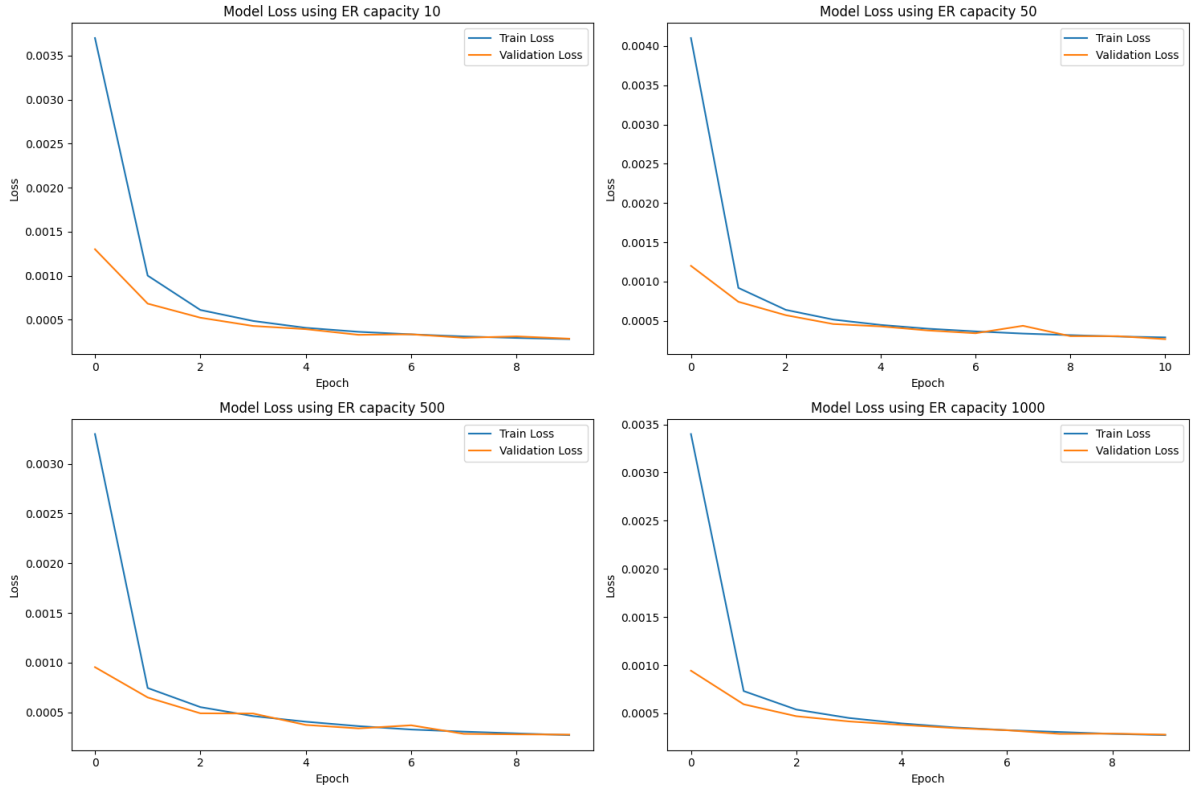


Figure 3.4: Different capacities of ER buffer on CICIDS-2017

From the figure it follows that ER capacity 1000 has the smoothest line and the largest correlation between train and validation loss, which makes it less susceptible to overfitting.

For Kyoto2006, a similar experiment has been done. The results of this experiment can be seen in figure 3.5

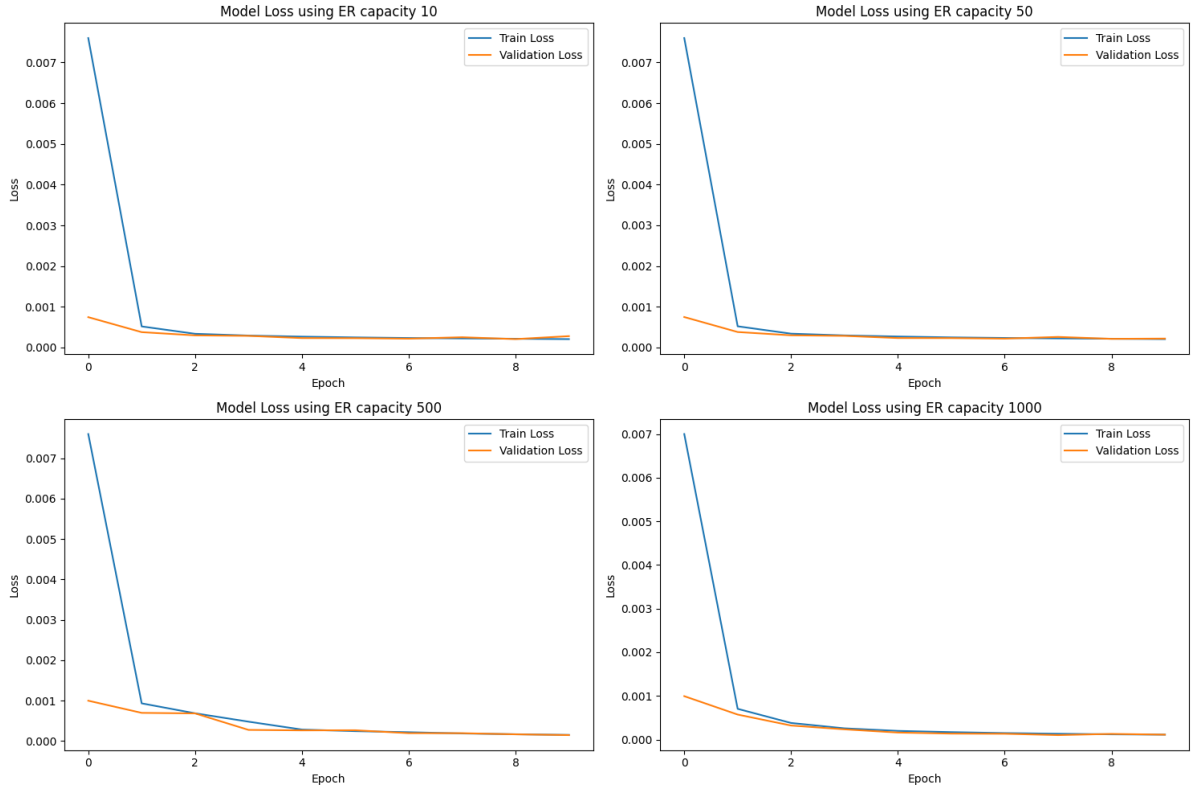


Figure 3.5: Different capacities of ER buffer on Kyoto2006

From this plot it follows that there is less variation in performance with different capacities. It can be seen that with a capacity of 500, the performance has a spike between train and validation loss. Therefore it has been chosen to keep this capacity at 50.

As a summary, the hyperparameters that can be optimized are written below, including the value that they are currently set to:

	CICIDS-2017	**Kyoto2006**
Capacity of the buffer	1000	50
The number of samples to be added to the buffer each epoch	10	10
The number of samples retrieved from the buffer each epoch	5	5
The heuristic on what samples to add to the buffer	random	random
The heuristic on what samples to retrieve from the buffer	random, reservoir sampling	random, reservoir sampling

Table 3.3: Initialization of experience replay buffer

3.1.6 Optimization

Experiments were performed on Kyoto2006 to determine the optimal learning rate. The results can be seen in figure 3.6. From the left plot it followed that LR = 0.05 did not allow any learning. From the right plot it followed that the three smallest learning rates provided similar results. Therefore, 0.005 has been chosen as learning rate. This learning rate has also been used for the experiments on CICIDS-2017.

The network was optimized using an Adam optimizer. As a loss function, the mean square error has been chosen and 25 epochs have been used for training. With continual learning enabled, samples were taken from the buffer and added to the current batch of training data. When training without continual learning, only the training data was used. During training, a validation split of 10% was used.

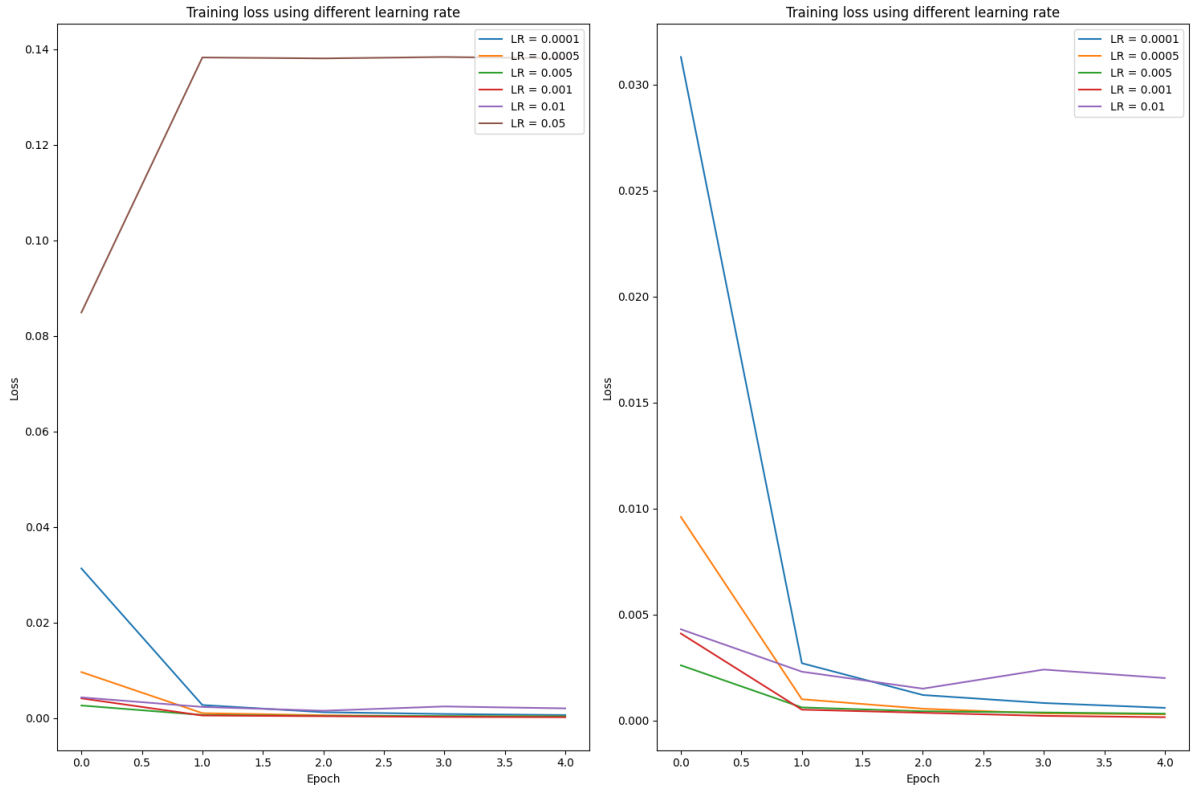


Figure 3.6: Different learning rates on Kyoto2006 training data

3.1.7 Evaluation

All random states have been fixed to a seed, to ensure results are reproducible.

CICIDS-2017 For evaluation, the files of Thursday and Friday have been randomly selected and used. These files contain benign as well as attack traffic and are labeled as such. The labels are extracted and stored separately. As with the training, the same preprocessing is being applied to the evaluation data. An important difference with respect to the training pipeline is the fitting of the scaler, where the same scaler from the training pipeline is fitted to the evaluation data. It is however not modified. The evaluation data is fed to the network and predictions are determined. Using the MSE, the ROC curve and therefore the Area Under the Curve are determined.

Kyoto2006 For training, the benign data from November and December 2006 is used. This data is first split in a 80/20 train/test split and then ultimately the 80% train split is being used for training. As a test set in all instances, the remaining 20% benign traffic is combined with the attack traffic from November and December 2006.
In the first experiment, no extra training is done. In the second experiment, the network is presented and trained with extra data of March and April 2007. This set only contains benign traffic. For the third experiment, the network is presented and trained with the same extra data of March and April 2007, but now also experience replay is activated.
For all three experiments, the performance is tested on data from December 2007. This data is unseen and contains both benign as well as attack data. This measures the performance on unseen data.
The choices to only work with subsets of data is because of a limit of processing power. Therefore, these periods have been selected at random.

The above methodology is also demonstrated in table 3.4.

	2006		**2007**	
	Nov + Dec	**Nov + Dec**	**Mar + Apr**	**Dec**
	80% benign	**all attack + remaining 20% benign**	**benign**	**All**
1	Training	Test	-	Test
2	Training	Test	Training	Test
3	Training	Test	Training + Replay	Test

Table 3.4: Data used for the three experiments on Kyoto2006

The feature averages for both benign and attack data are plotted for all features in figure 3.7.

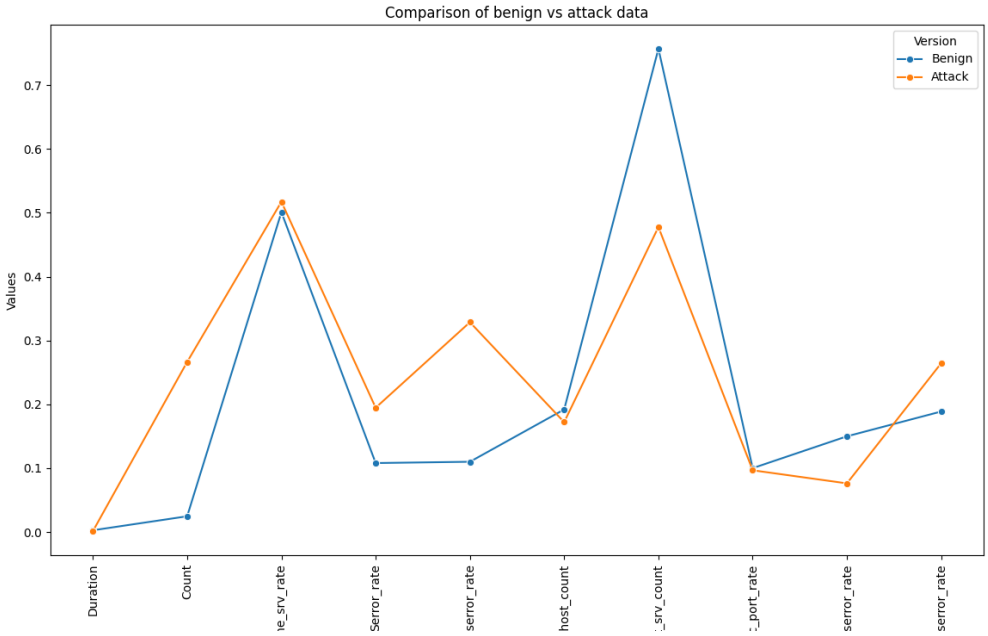


Figure 3.7: Benign versus attack traffic in Kyoto2006

3.2 RQ2: How can continual learning improve the performance of autoencoders on anomaly-based NIDS using real network data?

Because this thesis was written in collaboration with Northwave Cyber Security, there was the unique opportunity to perform experiments on data that is not captured in a benchmark dataset, but in a real network instead. Furthermore, there are experts available that can give more insight in what would be important to deploy such a system in practice. To answer this corresponding research question, the methodology can be split into two main goals: Performing experiments on real data instead of benchmark data and getting to know what is necessary from a company perspective to implement such a solution. The methodology for the first experiment is discussed hereafter, the second goal is discussed using expert knowledge.

3.2.1 Dataset

The data at Northwave is collected using Zeek, formerly called Bro [34]. Zeek is an open source network security monitoring tool which can log, among other usecases, connections made on the network. This connection data is used in the experiment. It consists of data that is captured from the network of a customer of the Northwave SOC and is free of attacks as determined by the signature-based NIDS. Therefore, there is only one class of data available. The attack class will be simulated by manually altering flows present in the dataset. The log is available as a .log textfile and is parsed into a Pandas DataFrame using a package called Zeek Analysis Tools (ZAT) [35]. In the remainder of this thesis, this dataset is referred to as NWDATA. In principle, there is data on three days available. However, because of the size of the dataset, only a random subset is used.

3.2.2 Feature selection

The dataset consists of connection logs/flows with multiple features available. Most features that are being used are numerical features and have been selected because of this property. Choosing not to use the IP addresses was a very explicit choice, to not link the traffic to the IP address, but instead analyse the properties of the network flow itself. The same holds for the exclusion of the vlan.

Numerical	**Categorical**
duration	proto
orig_bytes	service
resp_bytes	conn_state
missed_bytes	
orig_pkts	
orig_ip_bytes	
resp_pkts	
resp_ip_bytes	

All the categorical features have been one-hot encoded using the Pandas get_dummies function. As a result, the flows have 38 features which are used for learning. The experiment on NWDATA is of a similar structure and setup as that for Kyoto2006. In the remainder of this chapter, the differences per corresponding section will be listed.

3.2.3 Preprocessing

The only difference with respect to the preprocessing of Kyoto2006 is that NWDATA only contains benign data. Therefore, there is no separation of attack and benign traffic. The used scaler, MinMaxScaler, is the same.

3.2.4 Autoencoder

The same autoencoder structure as for Kyoto2006 has been chosen: 64, 16, 8, 16, 64 for the hidden layer sizes. Because of rapid overfitting, L2 regularization has been added.

3.2.5 Continual learning

The same replay method and parameters have been chosen as for Kyoto2006.

3.2.6 Optimization

Optimizing the autoencoder for use on NWDATA has been a challenge. The network was not able to learn when performing the initial training. For this reason, l2 regularization has been added and the learning rate has been lowered to 0.00005. These two modifications compared to the training of Kyoto2006 resulted in viable results. This effect can be seen in figure 3.8.

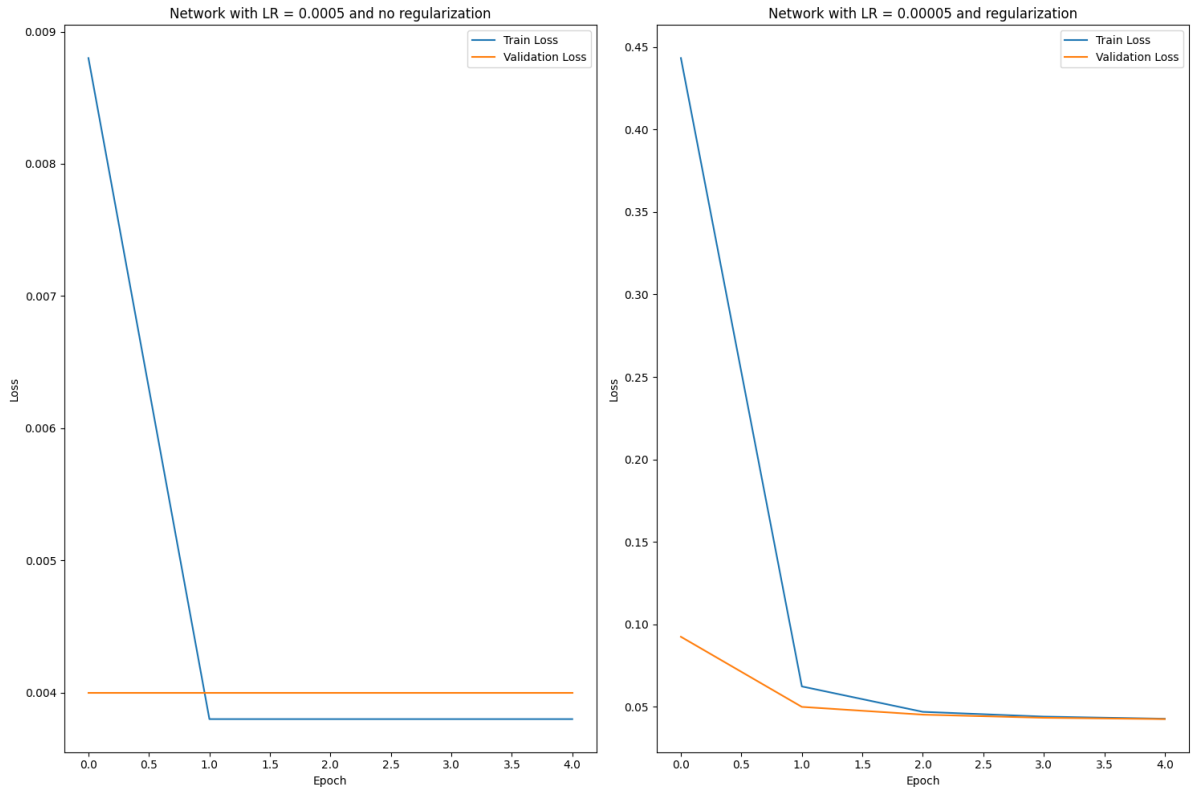


Figure 3.8: Different optimization for NWDATA

3.2.7 Evaluation and attack data generation

The structure for the experiments is similar to the experiments on Kyoto2006. This means, that experiments will be done with and without extra training on unseen data and with and without continual learning. In summary, the experiments can be found in 3.5.

Experiment	Extra Training	Continual Learning
1	No	No
2	Yes	No
3	No	Yes
4	Yes	Yes

Table 3.5: Experiments on NWDATA

In NWDATA, only benign traffic is available. To assess the performance on attack traffic, a part of the test set will be randomly modified by adding a random uniform value between -0.3 and 0.3 to random columns of the sample. While this procedure does not necessarily resemble attack data, it does represent anomalous data, which should be detected. Therefore, for this experiment, this data will be treated as attack data.

For an example data point, the result of this procedure is demonstrated in figure 3.9.

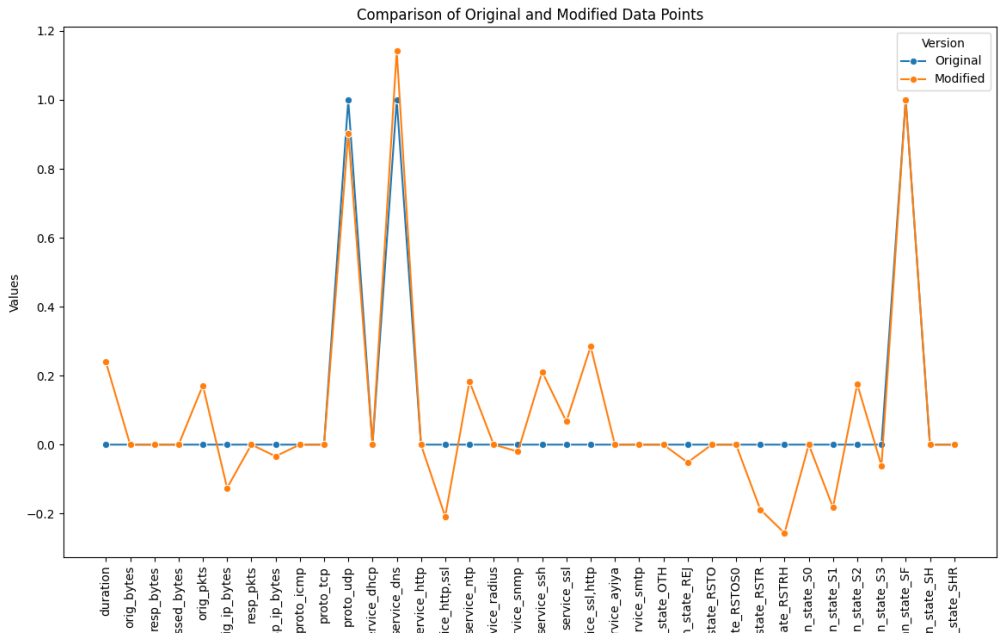


Figure 3.9: Attack generation for NWDATA

Chapter 4

Results

In this chapter the results will be discussed. In line with the conclusions made by Zavrak et al [22], the ROC curve is used as main evaluation metric. As concluded by the authors, this is also the standard method of reporting performance when dealing with skewed datasets.

4.1 RQ1: How can continual learning improve the performance of autoencoders on academic anomaly-based NIDS datasets?

To answer this research question, experiments have been done on the CICIDS-2017 and Kyoto2006 datasets.

4.1.1 CICIDS-2017

The experiment on CICIDS-2017 is taken as a baseline, to verify performance against literature. The network has been trained on the data from Monday, which is benign. The evaluation happened on the files for Thursday and Friday, which have been randomly selected. Experiments were ran five times with random seeds. On average, an AUC of 0.9199 has been recorded without continual learning enabled. With continual learning enabled, the AUC was 0.9271. The corresponding ROC curves can be seen in figure 4.1.

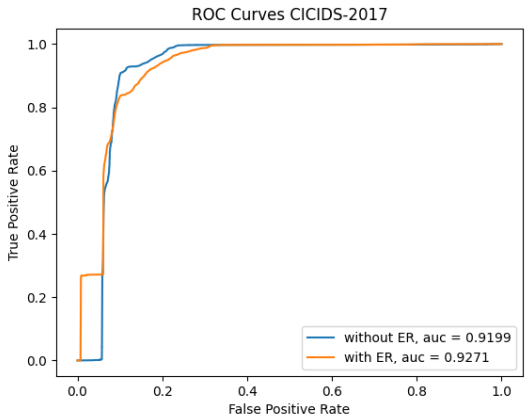


Figure 4.1: ROC curves for CICIDS-2017

From the plot it follows that while the AUC value is almost identical, the curves are not. For the method with experience replay, the TPR first increases without increasing the FPR.

Furthermore, the average reconstruction errors per attack type have been calculated with and without experience replay. These can be found in table 4.1.

Attack type	Count	Avg MSE with ER	Avg MSE without ER	Difference (with - without)
BENIGN	1231773	0.001666	0.001547	0.000119
Bot	1956	0.000252	0.000244	0.000007
DDoS	128025	0.002684	0.003161	-0.000477
Infiltration	36	0.001529	0.001628	-0.000098
PortScan	158804	0.019433	0.010692	0.008741

Table 4.1: Average reconstruction error per attack type in CICIDS-2017

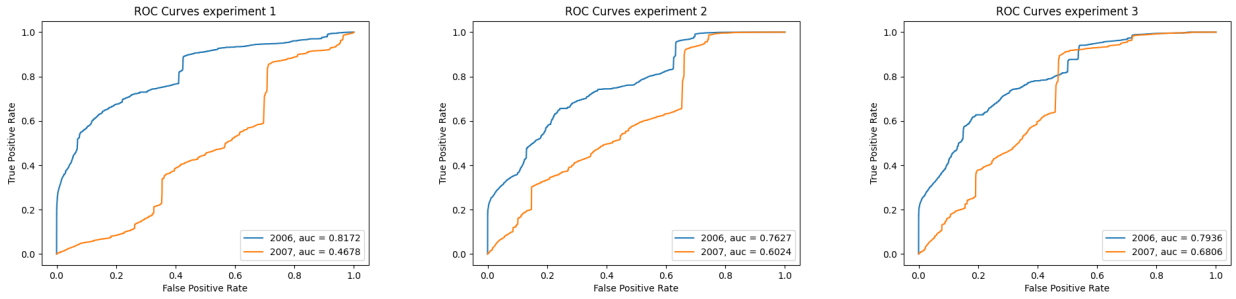
4.1.2 Kyoto2006

For the Kyoto2006 dataset, a more extensive set of experiments has been performed. In table 4.2, the results are reported corresponding to the experiments as mentioned in 3.4. In this case, all experiments were also ran five times and the average AUC is reported.

	AUC	
	2006	**2007**
1	0.8172	0.4678
2	0.7627	0.6024
3	0.7936	0.6806

Table 4.2: Results for experiments on Kyoto dataset

The ROC curves are plotted in figure 4.2.



(a) Experiment 1 (b) Experiment 2 (c) Experiment3

Figure 4.2: ROC curves for experiments on Kyoto2006

As a reference for the experiments, table 3.4 can be consulted. As a summary: Experiment 1 is an experiment without extra training and without experience replay. Experiment 2 is with extra training, without experience replay and experiment 3 is with extra training, with experience replay. It can be seen that the curves for all experiments differ.

4.2 RQ2: How can continual learning improve the performance of autoencoders on anomaly-based NIDS using real network data?

As described in the Methodology, the evaluation of this experiment is a bit different from the experiments done on synthetic datasets. In line with the experiments mentioned, the results can be found in 4.3.

Experiment	Extra Training	Continual Learning	AUC
1	No	No	0.757
2	Yes	No	0.756
3	No	Yes	0.749
4	Yes	Yes	0.750

Table 4.3: Experiments on NWDATA

The corresponding ROC curves are plotted in figure 4.3.

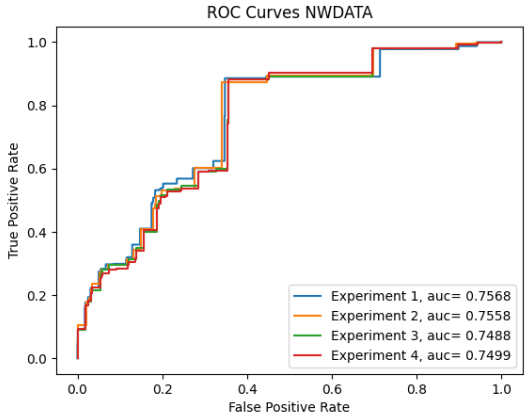


Figure 4.3: ROC curves for NWDATA

From this plot it can be seen that all curves and AUC scores are almost identical.

For the second aspect of this research question, expert knowledge has been collected from Northwave. From this, it followed that a high TPR and a low FPR is important. A high TPR ensures that all alarms that are generated resemble actual attacks. A low FPR ensures that not many alarms are generated which do not neccesarily resemble actual attacks.

Chapter 5

Conclusion and discussion

The final chapter of this thesis discusses the conclusions, limitations and directions for possible future work.

5.1 Conclusions

Using the results on both the CICIDS-2017 and Kyoto2006 datasets, **RQ1** has been answered: How can continual learning improve the performance of autoencoders on academic anomaly-based NIDS datasets?

5.1.1 RQ1: How can continual learning improve the performance of autoencoders on academic anomaly-based NIDS datasets?

On the CICIDS-2017 dataset, an AUC of 0.9199 has been observed without continual learning and an AUC of 0.9271 has been observed with continual learning. The performance of the autoencoder with continual learning is in line with the reference study by Verkerken et al [26], where an AUC of 0.978 was achieved with the same number of hidden units.
In terms of AUC, no significant improvements have been observed when assessing the AUC between evaluating the network with and without continual learning enabled. However, based on expert knowledge at Northwave, it became clear that the ROC curve is important as well. In an ideal scenario, a solution should only provide true positives, which means that everything that is being reported as suspicious is an actual attack. In practice, false positives are also reported. This means that an analyst has to investigate the alarm and then has to conclude that there is no real attack. With experience replay, the ROC curve first shows approximately 25% of true positives. Therefore, this saves valuable resources.

Finally, the experiment showed that there is a correlation between the type of attack and the effect of experience replay on the MSE. From a preliminary conclusion based on the average reconstruction errors per attack type as shown in table 4.1, PortScan might benefit the most from experience replay.

The research on Kyoto2006 was more extensive, and therefore more results have been obtained. It is clear that when there is no extra training involved (experiment 1), the network performs relatively poorly on unseen data from December 2007. Since the AUC dropped significantly, even under chance level, it is expected that a distribution shift has occurred between 2006 and 2007. This hypothesis is preliminary investigated by plotting the feature averages for the 2006 and 2007 testing set. The results can be seen in figure 5.1.

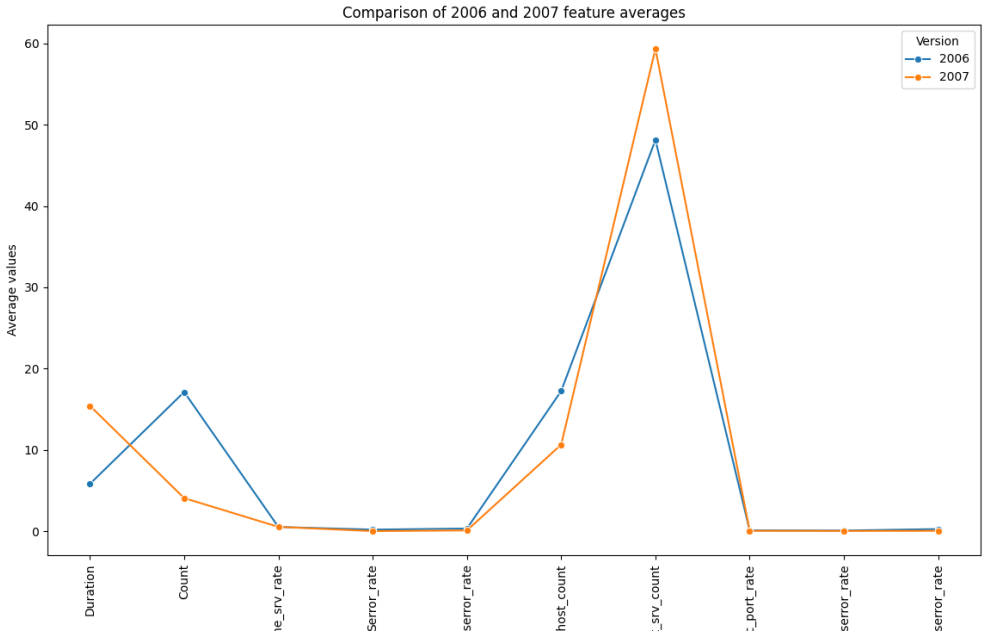


Figure 5.1: Feature averages for 2006 and 2007 on Kyoto2006

Based on this plot, it appears that there are some features that have changed between 2006 and 2007.

Experiment 2 involved the extra training of data from March and April 2007. This did have a positive effect on the AUC score while only slightly dropping the performance on the 2006 dataset. Experiment 3, using continual learning / replay, improved even upon that. This experiment showed good results on both unseen data as well as data from the initial year the network was trained on.
Based on these experiments, it is concluded that a replay approach to implement continual learning has shown potential on academic datasets when purely assessing the performance by means of the AUC score. This is also in line with the work by Faber et al [21]. In their work, they achieved an even greater improvement in terms of AUC. However, the chosen continual learning approach as well as the chosen dataset differed. The results can therefore not be compared directly. A possible first explanation for the difference in performance is that VLAD [21] includes *lifelong change point detection*, which can distinguish between new data for existing knowledge or entirely new knowledge in the network.

5.1.2 RQ2: How can continual learning improve the performance of autoencoders on anomaly-based NIDS using real network data?

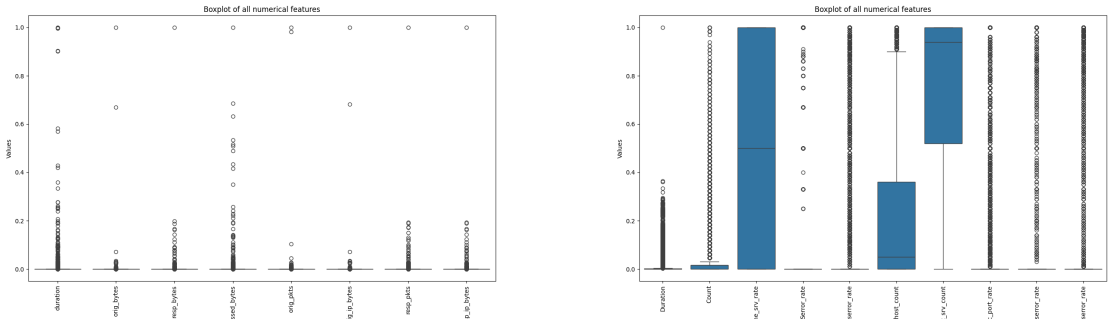
To answer this research question, experiments have been performed with data from a commercial network. The resulting dataset has been called NWDATA. The evaluation of this dataset was less straightforward as compared to the academic datasets, due to the lack of a predefined test set that contains both benign and attack data. Attack data was not present in general. Therefore, synthetic data has been generated using a custom data modification procedure as described in the Methodology. While not strictly representing actual attack traffic, this traffic does reflect anomalous traffic and should therefore be detected. From the results it followed that for this dataset, continual learning did not have a significant influence on the performance. This conclusion follows from both the almost identical AUC scores as well as the ROC curves themselves.

5.1.3 General conclusion

The research question cannot simply be answered by one answer as the conclusions for **RQ1** and **RQ2** contradict each other. While **RQ1** shows improved results for a relatively simple approach when evaluated on Kyoto2006, the conclusion for **RQ2** is that these findings not necessarily directly translate to real network data. Furthermore, the results on CICIDS-2017 were not conclusive either. While not improving performance purely assessed from the AUC, it did have a positive impact on the ROC curve, possibly saving valuable resources. Therefore, the general conclusion is that continual learning can be implemented in a practical use case, but more work needs to be done in order to conclusively evaluate its performance and the possible added benefits.

5.1.4 Possible reasons for performance difference

This section will present preliminary findings on the performance difference between NWDATA and Kyoto2006. As described in Methodology, it was difficult to get the network for NWDATA to train. In figure 5.2, a boxplot is shown for the numerical features of both datasets.



(a) NWDATA (b) Kyoto2006

Figure 5.2: Boxplots for numerical features of Kyoto2006 and NWDATA

From the figure, it follows that Kyoto2006 has a lot more variation in the data than NWDATA. This is a possible explanation for the training difficulties. Since there are very small differences between data points, it is expected that continual learning will cause the network to quickly overfit. Furthermore, from figure 3.7, it stood out that there is a noticeable difference in feature averages for benign and attack data in Kyoto2006. While generating attack traffic for NWDATA, the added noise was kept low. Therefore, the difference between benign and attack data is smaller. This can be seen in figure 3.9.

5.1.5 Comparison to reference studies

In Literature Review, some studies have been mentioned that perform similar research. For example work by Verkerken et al [26] and Faber et al [21]. This thesis differed from the mentioned work by focusing on a simple design, using only experience replay and assessed performance on data from a commercial SOC.

5.2 Limitations

This thesis was written during a period of 6 months. Within this period, the topic of cybersecurity as well as the concepts of continual learning had to be familiarised. When writing the proposal and starting the project, the work seemed more straightforward then it turned out to

be. Within this section, the most important limitations will be listed, which will be divided in limitations for general limitations, limitations for **RQ1** and limitations for **RQ2**.

5.2.1 General limitations

As a general limitation, it can be said that this thesis turned out to be more exploratory than expected when starting the project. Some important limitations will be mentioned below.

Choice of evaluation metric

In line with related literature, the AUC has been chosen as metric in order to assess the performance of continual learning on anomaly-based NIDS using autoencoders. The AUC has been chosen because it does not rely on thresholds and it can work with a skewed dataset.

Tuning of hyperparameters and autoencoder structure

Because of the scope of the research and the time constraints, most hyperparameters and the structure of the autoencoder have been either determined by related work, by a limited grid search or by manually determining them. There is a chance that the chosen parameters are not optimal and therefore the outcomes may differ. This includes that no further research has been performed on determining the best optimization strategy for both CICIDS-2017 and NWDATA. These have been based on the experiments done on Kyoto2006.

Feature selection

The feature selection has, like the tuning of hyperparameters, largely been based on literature or by intuition. This approach delivered feasible results. However, not all possible options have been explored.

Continual learning

The approach for continual learning is very basic, only incorporating an experience replay buffer that has parameters that have been self-determined. This holds for the capacity of the buffer, as well as the sampling procedures and the heuristics.

Size of dataset

Datasets have been chosen in order to be processable on a local computer. Therefore, only small subsets of all datasets could be used.

5.2.2 RQ1: How can continual learning improve the performance of autoencoders on academic anomaly-based NIDS datasets?

Datasets

There are limited academic datasets available, two of them have been chosen for the work in this thesis. More experiments could have been done on analyzing the specific contents of the dataset as well as using different datasets.

Data for extra training

The data that has been chosen for the extra online learning has been determined randomly. It has not been explored in depth what possible shifts are present in the data.

5.2.3 RQ2: How can continual learning improve the performance of autoencoders on anomaly-based NIDS using real network data?

Free of attacks

The data that was collected has been collected in a relatively quiet network, meaning it is almost guaranteed to be free of attacks. There is, however, never a 100% guarantee that this indeed is the case.

Non-public dataset

While being a unique contribution, the usage of a confidential dataset also entails that the findings cannot be validated or improved upon by other research.

Attack generation procedure

No attack traffic has been captured in NWDATA, therefore attack traffic had to be synthesized in order to produce anomalies that the network could detect. Without actual attack traffic, there can not be determined for sure whether the synthesized traffic is indeed similar to actual attack traffic.

5.2.4 Conclusion

In conclusion, it can be said that there are some significant limitations to the work described in this thesis. This thesis therefore has served mostly as exploratory work, with opportunities for future research. These will be listed in the section hereafter.

5.3 Future work

5.3.1 Optimization

As mentioned, no extensive research has been performed on determining the best optimization strategy. Further research may develop a method of determining this optimal strategy.

5.3.2 Types of attacks

While there is small evidence that the performance of experience replay is correlated to the type of attack, as shown by the experiments on CICIDS-2017, this hypothesis has not been investigated in depth. Future research may further investigate the influence of attack type on the performance of experience replay.

5.3.3 Data shifts

Small evidence has been presented that a distribution shift occurred in Kyoto2006 between 2006 and 2007. Future research may further investigate this effect and further explore how this can be dealt with.

5.3.4 Business use case

No significant attention has been given to the implications of using such a replay approach in a commercial setting. For example, the storage of samples costs disk space and therefore money. Future work may further investigate the implications of deploying such a solution.

5.3.5 Validating design choices

Because of the exploratory nature of the work in this thesis, the design choices have not been motivated and validated to a large extent. Future work may validate the chosen parameters and may develop a framework for assessing these values.

5.3.6 Attack data generation

The procedure to generate attack data for NWDATA using a small parameter has been chosen by intuition. From this work, it followed that in Kyoto2006 there was a larger difference between attack and benign data. Future research may investigate what attack traffic in a real network would look like, and whether the attack data generation procedure was indeed a feasible approach.

5.3.7 Continual learning approach

Using the experience replay approach that was chosen in this work, this theses provided small evidence that this is a useful technique that at least works on academic datasets. For future work, research may be performed in including other continual learning approaches.

5.3.8 Size of datasets

Because of computing power constraints, subsets or not extremely large datasets have been selected for this work. Future work may look into the usability and generalization of the conclusions when applied to larger datasets.

5.3.9 Using more real datasets

From the experiments on NWDATA, it followed that a real network does not have many interesting patterns in it, making it difficult to train a network on this data. In benchmark datasets, there is a lot more variation in the data, which might explain a positive performance. It would be interesting to do more research on real data, thus without many interesting patterns. This also translates to expanding the work by Faber et al [21] and Verkerken et al [26] by using their approach on real data. Based on the conclusions in this work, it is expected that their performance on real data will be lower as compared to the used datasets.

Bibliography

[1] A. M. Tonge, S. S. Kasture, and S. R. Chaudhari, "Cyber security: challenges for society-literature review," *IOSR Journal of Computer Engineering*, vol. 12, pp. 67–75, 2013.

[2] S. Mukkamala, A. Sung, and A. Abraham, "Cyber security challenges: Designing efficient intrusion detection systems and antivirus tools," *Enhancing Computer Security with Smart Technology.*, 2005.

[3] S. Sharma, K. K. Sharma, A. K. Jha, D. Tiwari, A. K. Jain, and Vikas, "Advancements in machine learning for intrusion detection in cloud environments," *INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT*, vol. 07, 7 2023.

[4] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, vol. 114, pp. 3521–3526, 3 2017.

[5] G. M. van de Ven, N. Soures, and D. Kudithipudi, "Continual learning and catastrophic forgetting," 3 2024.

[6] J. Keijer, "Unsupervised anomaly-based network intrusion detection using auto encoders for practical use," 2021. [Online]. Available: https://essay.utwente.nl/86521/1/Julik_Keijer_MA_EEMCS.pdf

[7] S. K. Amalapuram, A. Tadwai, R. Vinta, S. S. Channappayya, and B. R. Tamma, "Continual learning for anomaly based network intrusion detection," *2022 14th International Conference on COMmunication Systems and NETworkS, COMSNETS 2022*, pp. 497–505, 2022.

[8] A. P. Singh and M. D. Singh, "Analysis of host-based and network-based intrusion detection system," *International Journal of Computer Network and Information Security*, vol. 6, pp. 41–47, 2014.

[9] S. Parhizkari, *Anomaly Detection in Intrusion Detection Systems*, 10 2023.

[10] S. Omar, A. Ngadi, and H. H. Jebur, "Machine learning techniques for anomaly detection: An overview," *International Journal of Computer Applications*, vol. 79, pp. 33–41, 10 2013.

[11] U. Michelucci, "An introduction to autoencoders," vol. 1, 2022. [Online]. Available: http://arxiv.org/abs/2201.03898

[12] J. Riebesell and S. Bringuier, "Collection of standalone tikz images," 2020, 10.5281/zenodo.7486911 - https://github.com/janosh/tikz. [Online]. Available: https://github.com/janosh/tikz

[13] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 5 2019.

[14] R. Hadsell, D. Rao, A. A. Rusu, and R. Pascanu, "Embracing change: Continual learning in deep neural networks," *Trends in Cognitive Sciences*, vol. 24, pp. 1028–1040, 12 2020.

[15] M. Mermillod, A. Bugaiska, and P. Bonin, "The stability-plasticity dilemma: investigating the continuum from catastrophic forgetting to age-limited learning effects," *Frontiers in Psychology*, vol. 4, 2013.

[16] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," 6 2016.

[17] D. Turner, P. J. S. Cardoso, and J. M. F. Rodrigues, "Modular dynamic neural network: A continual learning architecture," *Applied Sciences*, vol. 11, p. 12078, 12 2021.

[18] F. Wiewel and B. Yang, "Continual learning for anomaly detection with variational autoencoder." IEEE, 5 2019, pp. 3837–3841, https://github.com/satolab12/Continual-Learning-for-Anomaly-Detection-with-Variational-Autoencoder.

[19] S. Prasath, K. Sethi, D. Mohanty, P. Bera, and S. R. Samantaray, "Analysis of continual learning models for intrusion detection system," *IEEE Access*, vol. 10, pp. 121 444–121 464, 2022.

[20] K. Faber, R. Corizzo, B. Sniezynski, and N. Japkowicz, "Active lifelong anomaly detection with experience replay." IEEE, 10 2022, pp. 1–10.

[21] ——, "Vlad: Task-agnostic vae-based lifelong anomaly detection," *Neural Networks*, vol. 165, pp. 248–273, 8 2023.

[22] S. Zavrak and M. Iskefiyeli, "Anomaly-based intrusion detection from network flow features using variational autoencoder," *IEEE Access*, vol. 8, pp. 108 346–108 358, 2020.

[23] R. Alshamy and M. Ghurab, "A review of big data in network intrusion detection system: Challenges, approaches, datasets, and tools," *International Journal of Computer Sciences and Engineering*, vol. 8, pp. 62–75, 7 2020.

[24] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," vol. 2018-January, 2018.

[25] J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue, and K. Nakao, "Statistical analysis of honeypot data and building of kyoto 2006+ dataset for nids evaluation." ACM, 4 2011, pp. 29–36.

[26] M. Verkerken, L. D'hooge, T. Wauters, B. Volckaert, and F. D. Turck, "Towards model generalization for intrusion detection: Unsupervised machine learning techniques," *Journal of Network and Systems Management*, vol. 30, p. 12, 1 2022.

[27] D. Protić and M. Stanković, "Anomaly-based intrusion detection: Feature selection and normalization influence to the machine learning models accuracy," *European Journal of Engineering and Formal Sciences*, vol. 2, p. 101, 12 2018.

[28] V. Sharma, "A study on data scaling methods for machine learning," *International Journal for Global Academic & Scientific Research*, vol. 1, 2 2022.

[29] F. G. Pedregosa, Varoquaux, A. V. Gramfort, Michel, B. O. Thirion, Grisel, M. P. Blondel, Prettenhofer, R. V. Weiss, Dubourg, J. A. Vanderplas, Passos, D. M. Cournapeau, Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in {P}ython," *Journal of Machine Learning Research*, vol. 12, 2011.

[30] H. H. Coşgun, "Which data scaling technique should i use ?" 8 2023. [Online]. Available: https://medium.com/@hhuseyincosgun/which-data-scaling-technique-should-i-use-a1615292061e

[31] J. Dancker, "A brief introduction to feature scaling," 10 2022. [Online]. Available: https://medium.com/@jodancker/a-brief-introduction-to-feature-scaling-e396356937b8

[32] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[33] D. Rolnick, A. Ahuja, J. Schwarz, T. P. Lillicrap, and G. Wayne, "Experience replay for continual learning," 11 2018.

[34] V. Paxson, "Bro: A system for detecting network intruders in real-time," *Computer Networks*, vol. 31, 1999.

[35] B. Wylie and M. Eriksson, "Zeek analysis tools (zat)," 2024. [Online]. Available: https://supercowpowers.github.io/zat/