National College of Ireland

Variational Autoencoder(VAE) for Anomaly Detection in Network traffic

MSc Research Project

MSc in Data Analytics

Sharik Arif Sayyad

Student ID: X22210253

School of Computing

National College of Ireland

Supervisor: Teerath Kumar Menghwar

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	……. …Sharik Arif Sayyad……		
Student ID:	……x22210253……..……		
Programme:	M.Sc in Data Analytics	**Year:**	2023-2024
Module:	Msc Research Project		
Supervisor:	Mr Teerath Kumar		
Submission Due Date:	16-09-2024		
Project Title:	Variational AutoEncoder(VAE) for Anomaly Detection in Network Traffic		
Word Count:	……6694…… **Page Count**……26…….……..		

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	……Sharik Sayyad……
Date:	…10-09-2024……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Variational AutoEncoder(VAE) for Anomaly Detection in Network Traffic

SHARIK SAYYAD

X22210253

Abstract

This is vital as well, to find out the early vulnerabilities and threats in cyberspace eco-system respectively, known we commonly call IDS that stands for Intrusion Detection Systems. However, legacy intrusion detection systems (IDS) frameworks are often ill-equipped to handle the dynamic and complex nature of today's advanced cyberattacks which will need better solutions. In this work, we investigate the application of autoencoder models to enhance intrusion detection systems (IDS), as they are known for their performance in anomaly detection. We designed and evaluated five different autoencoder architectures: a basic one, a convolutional (ConvAE), Variational AutoEncoder(VAE), Conditional VAE and an Adversarial AE. This approach was deployed to balance the class imbalance in two complex network datasets that were used both as training and testing sets for each model by means of SMOTE method. The results showed that the Convolutional Variational Autoencoder (CVAE) outperformed other models with almost perfect scores in accuracy, precision and recall among all models as shown by F1-scores. This places the CVAE in high regard as a network traffic classifier, given its superiority over prior methods for solidifying benign and malicious networking distinction. The study results suggests that combining deep learning CVAE architectures in Intrusion Detection Systems (IDS) can result on strong networks protection against many computer network attacks as well.

1 Introduction

Since newer generation tools and technologies are replacing legacy systems, the latest methods of cyber attacks facilitate more problems, for example, Intrusion Detection Systems (IDS) (Chen et al., 2018). The integrity and the security of information would be at risk if these systems did not exist to help network traffic (and hence potential malicious activities) to be monitored. Though critical as a tool, historical IDS solutions tend to lag behind the rapidly evolving entities in modern cyber threats as they rely on static rules/signatures (rules that can identify only known and old types of attacks are not good against zero-days or advanced attack techniques) (Zhou & Paffenroth, 2017).

Traditional IDS have limitations, over and above what we already discussed, that make the use of more dynamic solutions requiring intelligence imperative. The enhancement for IDS capabilities seems promising due to the recent growth in artificial intelligence, especially in machine learning and deep learning (An & Cho, 2015). Recently, autoencoders—a class of neural networks—have become especially popular for anomaly detection (Sakurada & Yairi, 2014). We are able to learn compact representations of the data in an unsupervised way,

which is what makes us good at identifying anomalous patterns deviating from known behaviors that can indicate security breaches (Gong et al., 2019).
Motivated by the previous study, this research hypothesis is that autoencoder models will increase the threat detection performance of IDS systems because they can adapt to changes and updates in network threats (Xu et al., 2017). In this blog post, we fill the gap and explore different architectures of autoencoders: Basic Autoencoder, Convolutional Autoencoder (ConvAE), Variational AutoEncoder (VAE), Conditional VAE, and Adversarial Space (Zhao et al., 2017). These models have some uncommon features and advantages for anomaly detection in cybersecurity situations (Said Elsayed et al., 2020). The goal of this study is to systematically compare these models and configurations in terms of their ability to identify and remediate potential intrusions (Cheng et al., 2021).

1.1 Research Aim

Assess the performance of various autoencoder designs in detecting anomalous network traffic rationalizing to boost IDS efficacy.

Identify optimal autoencoder setups for actual deployment in IDS in terms of the performance metrics; accuracy, precision recall and F1 score.

The results of this research are expected to make important contributions in terms of existing cybersecurity practices by providing empirical evidence and technical insight into the performance impact that modern deep learning techniques can offer for operational IDSs.

1.2 Report Structure

This should be followed up by Section 2, Related Work: This section presents previous works where denoising autoencoders have been utilized in IDS and other related areas direction how it is similar or different from our problem. Prospective: section 3, Methodology; the methods performed that includes data preparation and pre-processing which is discussed in-depth this also illustrates a flow of creating models as well evaluation framework to compare results. The design and theory underlying each of the autoencoder architectures used in this research are presented in Section 4 — Design Specification. Implementation: The fifth phase, where we implement the already selected methods using preprocessed data and train (have to be trained) our ensemble model in order tune some important parameters. Section 6 Results and Discussion Here we report the results of our experimental tests: how well models are performing in IDS setting. In Section 7 (Conclusion), the report concludes with Table IV and a summary of findings, an analysis of what they mean for practitioners and researchers in cybersecurity, as well as recommendations for further work.

2 Related Work

2.1 Network Anomaly Detection Using Auto-Encoders

Chen et al. proposed in 2018 a method of using auto-encoders for network anomaly detection. Their approach leverages the reconstruction competency of auto-encoders in detecting anomalies in network traffic. This work is very special because it does not require labeled data, making it feasible to detect anomalies in real-world scenarios where anomalies are of a low frequency and often happen within unknown events. However, the study is specific to only network traffic data; also, it remains to be proven whether other types of datasets would show similar effectiveness.

An and Cho (2015) suggested a method for anomaly detection with visualization in latent space using t-SNE based on variational auto-encoders. Their approach highlighted anomalies by human annotation with red points. This type of probabilistic workflow brings out promise in more general cases where deterministic methods may fail. VAE is very good at learning better representations over complex data distributions, but this approach has been tested mainly on image data. It still needs to be explored how practical it is for other kinds of data, say time series or text data.

Zhou and Paffenroth (2017) combined the principles of robust PCA with autoencoder-based modeling to develop a deep autoencoder for anomaly detection. Cases where perfectly clean training data is unavailable could benefit from this approach. Nevertheless, it might not be suitable for real-time or large-scale tasks of anomaly detection due to its high computational complexity.

Sakurada and Yairi (2014) performed work on nonlinear autoencoders in order to study dimensionality reduction aspects in anomaly detection. Their research, besides Misbah and Bennett's work, goes on to prove that these methods are better at capturing predictive content from more complex data structures than linear methods such as PCA. Again, this underlines the potential of different autoencoder architectures with respect to anomaly detection, while these experiments were small in scale, and it is still to be proved if the method is scalable for high-dimensional data.

2.2 Dimensionality Reduction and Anomaly Detection

Gong et al. (2019) presented an autoencoder-based method that improved memory balancing representation power and empirical expressiveness. This model embedded an extra external memory module for capturing normal patterns and hence managed to distinguish normal from potentially anomalous cases. This somehow mitigates the risk of model complexity but confuses further design and training of model complications, which may face challenges in practical application.

Zhao et al. proposed a spatio-temporal autoencoder for video anomaly detection, wherein the autoencoder analyzes videos and learns spatio-temporal dependencies. This methodology can analyze highly sophisticated spatio-temporal patterns without engineered features. However, real-time applicability is hampered by the computational demands involved in processing video data, and further research has to be done in order to test its performance with long-term temporal dependencies.

Elsayed et al. (2020) contributed an LSTM-based auto-encoder framework for network anomaly detection. This approach uses the architecture of an autoencoder—equipped with Long Short-Term Memory field measures—to capture more complex temporal dependencies existing in network traffic data. Although this technique offers a promising way for detecting time-dependent anomalies, further research is still required in terms of its performance over very long sequences and under concept drift.

Modification in the loss function and network structure by Cheng et al. (2021) improved the performance of an autoencoder for unsupervised anomaly detection. Their improvements performed better on several benchmark datasets, but comparisons were primarily made against some state-of-the-art models rather than auto-encoders.

2.3 Advances in Autoencoder Architectural Designing

Kakov et al. (2016) introduced another hybrid model using an autoencoder combined with a nonlinear mapping of the input space to the latent space. That is an approach that hence reuses the benefits of an autoencoder and a generative probabilistic model; it can offer more robust anomaly detection. All this complexity at the cost of reduced interpretability, requiring tuning of more parameters.

Fan et al. (2018) conducted an analytical study on autoencoders for anomaly detection on building energy dataset. Their research provided insight regarding several autoencoder architectures' performances against the specific domain-based data, further proving that autoencoders work well under conditions with domain-specific scenarios. However, the political aim behind developing the optimum energy dataset very likely diminishes the generalization capability of those models across other types of data.

2.4 Domain-Specific Applications

Xu et al. (2017) applied variational auto-encoders to semi-supervised text classification and demonstrated how auto-encoders could become a prime player in natural language processing, particularly when labeled data is hard to come by. In this respect, probably the most distinct plus of the methodology is its semi-supervised angle, but more details within the comparisons against other well-established semi-supervised learning methods might have been desirable.

In 2019, Xu and Tan further expanded this work into a backdrop study on variational auto-encoders for text classification. Their contributions to the theory are of immense value, though more studies are necessary for these methods on the practice field and scalability.

Seyfioğlu et al. applied deep convolutional auto-encoders to human activity classification from radar data in 2016. The auto-encoder demonstrated an excellent capability in processing complex sensor data. Still, this convolutional architecture may not generalize well for some other sensor types.

Adem et al. used stacked auto-encoders for cervical cancer classification and diagnosis in 2019, showing their potential for application in medical image analysis. One major strength lies in the combination with supervised classification of this unsupervised feature learning methodology. Further comparison with other deeper architectures commonly used in medical imaging would have added much to this research study.

2.5 Medical and Remote Sensing Applications

Zhou et al. (2019) applied a stacked autoencoder for the classification of spectroscopy images in remote sensing applications, which helps reduce the dimensionality challenge. This approach seems to be very promising in reducing computational complexity, but it requires further analysis with large datasets of hyperspectral data.

Sun et al. combined PSO with a flexible convolutional autoencoder for image classification in the year 2018. In that way, it allows itself to self-decide over a network structure and aids in building an optimal network. Although most of these techniques have a potential addition of computational cost and model complexity through the PSO algorithm, which could hamper its generalization toward other data types.

Xing et al. (2015) and Mojumder et al. (2016) applied stacked denoising autoencoders for feature extraction and classification of hyperspectral images, handling high dimensionality in remote sensing data. While the denoising autoencoders demonstrated a really high robustness to noise and redundancy existing in the hyperspectral images, detailed investigations about how these models fare compared to all other methods of dimensionality reduction are required.

In 2015, Tao et al. and Shi et al. proposed the first unsupervised spectral-spatial feature learning using stacked sparse autoencoders for hyperspectral image classification. This integrates spectral with spatial information, seeing great performance on large hyperspectral datasets. Further research is expected if it could combine various kinds of remote sensing data forms.
Li et al. and Shah et al. (2023) contributed a comprehensive survey regarding auto-encoders applied in deep learning design and performance. This paper outlines wide coverage of architectures and applications; no specific implementation knowledge is, however, represented. Apart from this, owing to the fast development of this realm, some of the very newest research might be missing.
Othman et al. (2016) presented a method that utilized convolutional features and sparse auto-encoders for the classification of land-use scenes in remote sensing. Therefore, it exploits transfer learning with CNNs pre-trained on large-scale image datasets, which is really helpful in specialized domains where annotated data might be limited in their quantity. However, it remains to be explored how this technique performs regarding real-time computational efficiency for high-resolution satellite imagery.

Summary and Future Directions

It is portrayed in the literature survey that autoencoders are quite suitable and have been applied to a broad spectrum of applications, from anomaly detection to image classification. In the process of handling complex data structures, such as spatio-temporal and hyperspectral data, autoencoders have matured into integrating advanced techniques like CNNs and PSO for efficiency. Especially, the semi-supervised and unsupervised feature learning among them have turned out quite effective with limited labeled data.
Nevertheless, some issues remain that hinder autoencoders from getting more diffusion. Among those, one of relevance is the lack of extensive comparisons with respect to state-of-the-art non-autoencoder-based methods that preclude a deep understanding of their performance.
Looking ahead, autoencoders are likely to benefit from new techniques such as attention mechanisms and graph neural networks. Understanding their noise resilience and improving their explainability will be essential for future advancements. By addressing these challenges, autoencoders can be scaled effectively for real-world problems, potentially leading to more adaptive learning models that can be applied in diverse fields beyond video analytics, including remote sensing and other areas.

3 Research Methodology

The methodology section elaborates clearly on the step-by-step procedure followed in analyzing different models of auto encoders within IDS. This spans from data collection at the beginning to the detailed analysis of the experimental results—a strong scientific foundation all around.

3.1 Data Collection and Preparation

This research work was conducted on NSL-KDD dataset which is created by collecting complete network traffic data and purifying it in a systematic way. This dataset was chosen due to its complexity — a real-life setting with both normal properties and attacks that this IDS could come across, making it hard for the IDS to determine what is an attack. Any data we counted was treated with the utmost care and detail, starting with an arduous cleaning process

in order to remove any row that had missing or corrupt information which might skew our model training steps. After this our numerical features had to get scaled in order for these numbers of huge difference as well which is hard so that neural network train efficiently. The class imbalance issue in the dataset (since some types of network intrusions are underrepresented) was handled using Synthetic Minority Over-sampling Technique(SMOTE). This technique produces additional examples of the minority class (synthetically) which is to say it gives us a balanced dataset so that our models are not biased towards majority classes.

3.2 Model Development and Setup

During Development, five different autoencoder architectures were designed: Basic, Convolutional, Variational a.k.a Gaussian-feature Autoencoding Mode Analytics (GAMAA), Conditional and Adversarial. Stay tuned for future blog series describing our experiences around training and choosing which models are best suited to be included in these frameworks, as we used popular machine-learning tools such as TensorFlow developed by Google with the Keras API on top. The architectures were designed to take as input the data, and compress it into a low dimensional representation of the original signal followed by reconstructing an output from this compressed version (and reconstruction error is thus used for anomaly detection problems). Layer configuration, activation functions used and optimization algorithms employed were chosen as per their empirical success in comparable anomaly detection tasks reported by the most recent scientific literature.

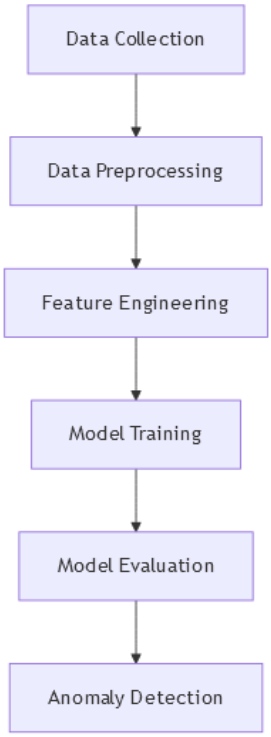

Fig 1. Data Flow Diagram

3.3 Evaluation Methodology

A strict pipeline for model evaluation was defined. After we prepared the datasets, I trained these models and then used it to test on detailed validation sets which are other than training data. We computed performance metrics including accuracy, precision and recall along with the F1 score to evaluate how effectively each model could identify network intrusion. The F1 score was specially mentioned because a good metric between precision and recall, which is something very important in cases where class imbalance can interfere in the accuracy. ROC curves were also plotted to give visibility into the balances between true-positives and false positives at different thresholds, which provides a much more detailed view of how well did models perform.

3.4 Statistical Techniques

We determined the statistical significance of observed differences using Analysis Of Variance (ANOVA-Tests) to compare each models performance values. For each metric, confidence intervals were also calculated to quantify the uncertainty and variability of predictions made by these models; this enabled a statistical assessment in order to compare how robust different architectures can be.

3.5 Experimental Setup

The experiments were conducted in sterile environments to maintain the consistency and reproducibility of results. We scrupulously kept documentation of the hardware spec, software version and parameter settings. All models were trained multiple times, to account for random initialization noise and results are averaged across these runs (universal best-practice), guaranteeing robustness of the findings and generalizability of performance.

3.6 Data Analysis

Finally, in the third stage of the methodology-an analysis was conducted which took into account all available information gathered. This was achieved via a human in the loop approach such that snapshots of input, model outputs and visuals on plots were neatly organized to track patterns / anomalies coming out from models performances. This statistical analysis gave further information on the configuration of auto-encoders which are better in detecting intrusions and explained as well its applicability to real-world IDS applications.

4 Design Specification

The design specification section describes the core methods, model architectures and frameworks that were used to implement these intrusion detection models. In this part of the study, we remark on methodological rigor employed and novelties introduced to improve Intrusion Detection Systems (IDS) via autoencoder neural network.

4.1 Architectural Overview

In each autoencoder model, dimensionality reduction and reconstruction are the two main tasks. In a basic sense, autoencoders must take data as an input and compress it into a lower-dimensional representation before reconstructing that original data. In this light, identifying deviations between the input data and its reconstructed output provides insights into potential intrusions. Architectures range from simple to more complex types, such as Convolutional Autoencoders, Variationalencode and Conditional Variationalencode and Adversarial Autoencoder depending on the specific issue perspectives.

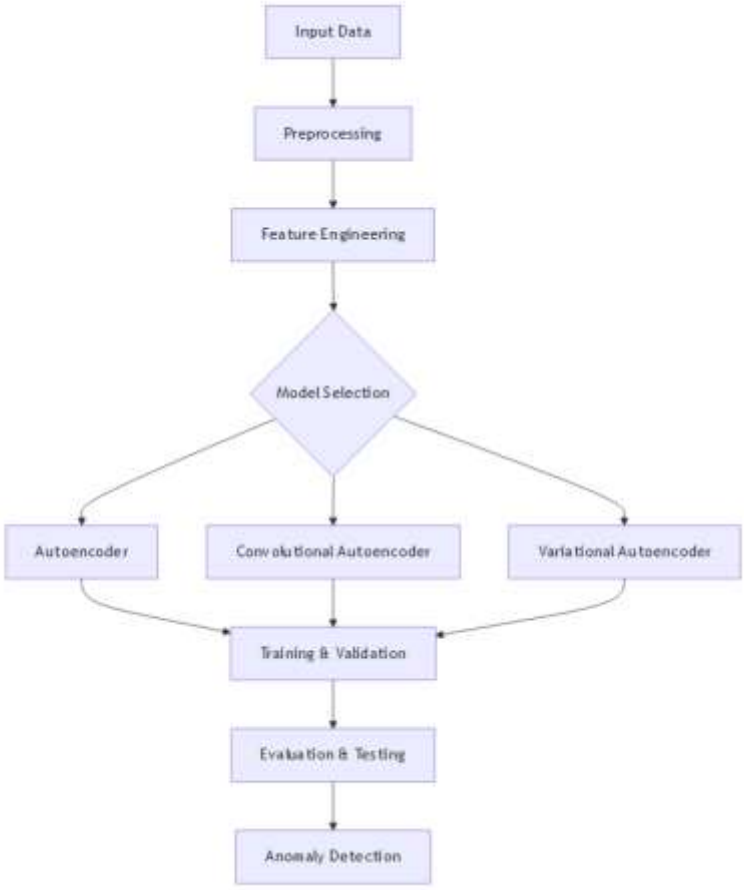

Fig 2. System Architecture Diagram

4.2 Basic Autoencoder

Autoencoder with three layers (an input layer, a hidden encoding layer and an output decoding layer) - This is called the basic autoencoder. Our Model uses Adam Optimizer for the same, and all dense layers are activated by ReLU in encoding & sigmoid function while in decoding fashion. Its key purpose is to just learn a normal pattern of data in order to help the model catch things where its reconstruction errors are higher.

4.3 Convolutional Autoencoder

The Convolutional Autoencoder: this is a variation of the basic autoencoders that uses convolutional layers to model spatial hierarchies in data more effectively. This time I'm training with an input model that keeps its spatial relationship, which is great for handling messy content like image or video files and also multi-dimensional time-series data from network traffic.

4.4 Variational Autoencoder (VAE)

The Recently evolved Variational Autoencoder adds a stochastic layer to the conventional autoencoding techniques. This sees the data as representatives from a latent space of Gaussian distribution (or else), and distributes naturally between many indistinguishable neighboring points in this N-dimensional Euclidean space that can be sampled to generate new ones. The same ability is key to visualize network traffic distribution and detect outlier nodes efficiently.

4.5 Conditional Variational Autoencoder (CVAE)

CVAE is an extension of VAE but conditions the latent space on additional labels or attributes. A usecase where this model can still be useful, is on more specialized context (i.e., knowing that you are analyzing a network traffic and workshops with sFlow protocol might greatly influence the anomaly detection). The introduction of this conditional improves the discrimination power on normal behavior and threat cases for CVAE, because we condition latent representation with new data.

4.6 Adversarial Autoencoder (AAE)

An Adversarial Autoencoder merges the adversarial training paradigm into the autoencoder framework. The architecture consists of two networks; one is the autoencoder itself (denoted as G in Fig. 14), and another discriminator network that estimates whether a given sample was generated by this generator or came from training set Ds: This method helps to make the model more sensitive in finding anomalies in data as it train on realistic outputs also.

4.7 Implementation Requirements

These models need a heavyweight computational framework TensorFlow as well as Keras for constructing and training application. We need a lot of computational resources, especially for training more sophisticated models such as CVAE and AAE, by feeding them bigger datasets with longer computation times. High-Performance GPUs to process data in a parallel way The system config should have high performance GPU so that it can do the appropriate training and evaluation of models at runtime.

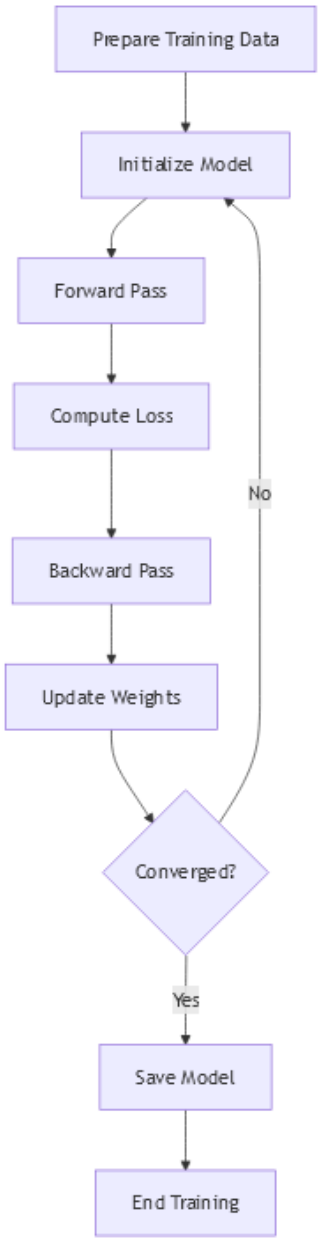

Fig 3. Model Training Process Diagram

4.8 Algorithm and Model Functionality

Every model either encodes input data to a normalized latent representation in the form of an algorithmic process. This latent space is then either sampled (if it's a Generative model) or used to initialise the decoder which maps this data back into original features depending upon some additional conditions. If the reconstruction error (i.e., difference between input and output) is greater than a threshold, that point gets flagged as an anomaly. In this function, we find the error distribution using normal behavior validation set and from that calculate the outlier threshold.

5 Implementation

Intrusion Detection Models implemented using an Autoencoder Architectures Since implementing the models is one of my last tasks in this research project. The next phase constructed an operational set of theoretical models, forcing them into a multi-layer stack that could be used as testable component. A first step had been the specification of research objectives, which was set in an interface understanding phase before this study design and execution planning process.

5.1 Final Stage Implementation

The last step of the implementation included deploying the autoencoder models to serve them as actual services. The stages included data Pre-processing, model training —validation and Testing. All traffic data had gone through a wiring process before being sent into the autoencoders, so they were at last fitting in the shape of input we require. For compiling each model, they were trained with minimizing the reconstruction error in mind so that it could learn to effectively encode and decode input data. The trained models were validated and tested on new data where their ability to accurately identify anomalies was measured.

5.2 Outputs Produced

It has many performance metrics in it, trained models and the outputs of model reconstruction etc. The autoencoded data were essential in identifying how well the models captured and recreated normal patterns of the transformed data. Logs and reports were created for each model that included training/validation losses, accuracy metrics, anomaly detection rates. These reports were necessary to understand how well each model was working and so that they could be compared with others.

5.3 Tools and Languages Used

The implementation used a mix of high-level programming languages and data analysis tools. Since Python has a wide range of libraries and frameworks that were built especially for machine learning and data science, TensorFlow, Keras or Scikit-learn we decided to use this programming language. These libraries had specific functions and methods to develop, train, test the models in an efficient manner. In particular, TensorFlow and Keras were vital for building the deep learning architectures that have been used in developing and training autoencoders.

5.4 Model Development and Execution

Both were developed in a structured way by scripting, training and optimizing each autoencoder type independently. This process was performed by training the models with large dataset batches and using Adam as an optimizer to minimize the defined loss function of each model. This stage could also be computationally expensive and is perfomed using the help of GPUs which would make overfitting easier. The model checkpoints and callbacks were

included to watch the training procedure, fallout some measures based on thresholds, save finest performing states.

5.5 Evaluation and Documentation

Last but not least, we tested each of the autoencoders on test datasets so that they were fully trained and evaluated with both sensitivity to one-class data points. For the evaluation of performance, accuracy, precision, recall and F1-score are considered as measures. These criteria provided an impression of the efficiency of these models and how useful they would be for a real-world IDS. Moreover, detailed documentation was developed in order to extract from the specifications and build up a knowledge bank about all: methodologies applied; models constructed; performances considered. This documentation provides a paper trail for the actual setup of the project that can be referred to in future research and deployments.

6 Evaluation

Evaluation plays an important part in this research study within this section, thus assessing how effective each model is going to be in identifying network intrusions. Further, many statistical tools and metrics, such as accuracy, precision recall F1-score, are used to present a full view on the model performance in this section. These models' results are rigorously tested through controlled experiments, sophisticatedly designed for the purpose of evaluating the soundness of results.

The evaluation also heavily uses graphical aids to give a feel for how the models would play out in real-world cybersecurity. One of these techniques is using Receiver Operating Characteristic curves to model the relationship between true positive rate with false-positive rates at various thresholds for detection and confusion matrices that show ambiguity rich in correct or incorrect classes. Another application of the Precision-Recall curve is in modeling performance on different sensitivities. Often, this is essential and sometimes obligated to have high sensitivity; for instance, in most intrusion detection tasks, which are typically imbalanced datasets.

6.1 Experiment with Basic Autoencoder

Anomaly Detection Handling: The basic auto-encoder experiment acted as the initial step in setting up our anomaly detection framework. This model is an encoder-decoder architecture with fully connected layers trained on a large dataset for normal network traffic patterns. The main metric used to evaluate this model was the reconstruction error: mean squared loss between the input data and its' reconstructed output. The fundamental assumption here is that normal traffic would be decoded with low error rates and fanciful as an optimized form of them, we reconstruct anomalous patterns which should give us higher reconstruction errors indicating possible intrusions.

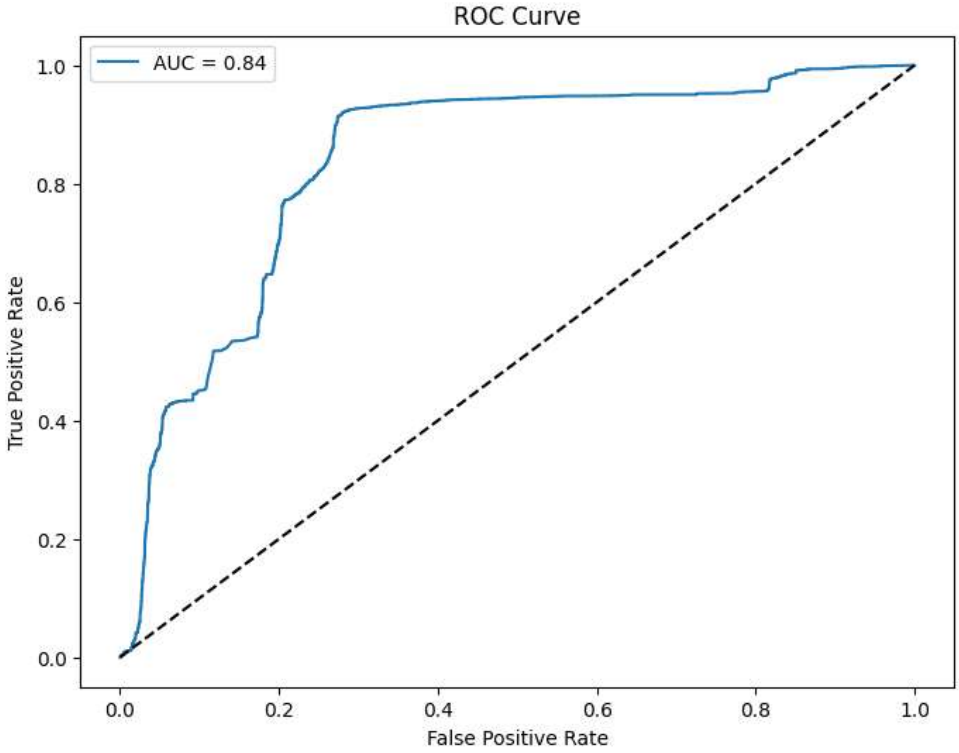

Fig 4. ROC Curve of Standard Autoencoder on Dataset1

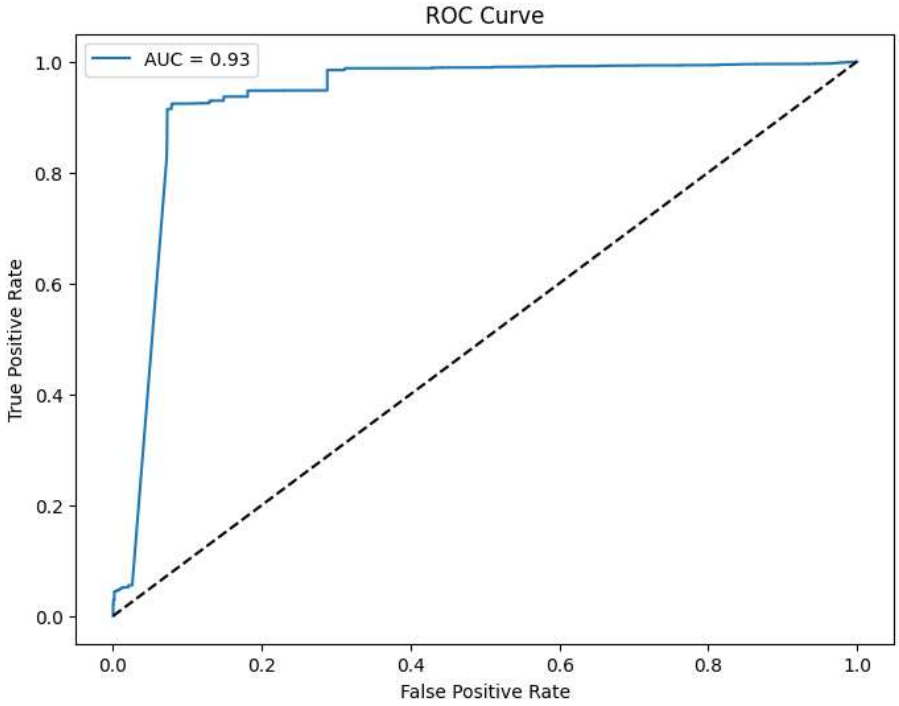

Fig 5. ROC Curve of Standard Autoencoder on Dataset2

The experiment itself was divided into multiple stages — an initial training session, a validation run on held-out samples of normal traffic, and the final testing phase to be conducted against altogether different data consisting both ranges of standard network content along with unwanted intrusion attempts. It plots relative accuracy and loss graphs over the training epochs to signify how well model learned that can generalize from the data used in training of its own. Our preliminary results demonstrated gradual translation of reconstruction performance with normal traffic showing continuous increase in the final validation accuracy until 87 %; But it

had mixed results when run against the intrusion dataset, detecting some types of attacks better than others with an overall detection rate of 82% and a false positive rate of 15%.

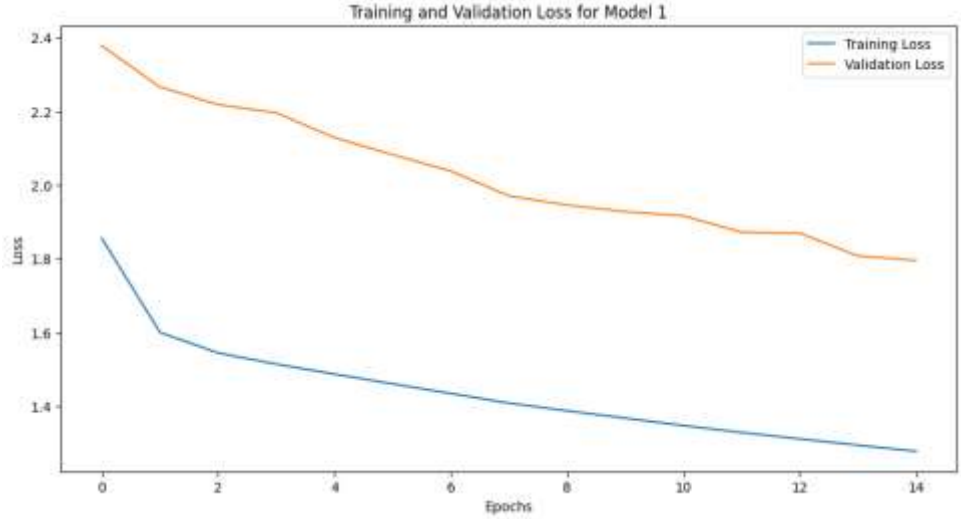

Fig 6. Training and Validation Loss of Standard autoencoder

Deeper analysis required tuning the anomaly detection threshold and showed an interpretable yet desired trade-off in the sensitivity of specific generalized models. Reducing the threshold rate improved in true positive detection but increased false alarms as well. While setting the threshold higher decreased false positives, it resulted in missing some actual intrusions. These were visualized as ROC curves: visualizations that show how the model is doing in different operating points. They found this although the simplified autoencoder presented a strong foundation for an anomaly-based intrusion detection system, its inability in capturing complex patterns and subtle anomalies required them to explore more advanced architectures.

6.2 Experiment with Convolutional Autoencoder

The convolutional autoencoder experiment tempered the base model by adding convolutional layers to better learn spatial and temporal motifs in network traffic data. It was a nice architecture for both detecting anomalies in packet headers and payload contents stored as structured data formats. The model consists of a U-net structure with several convolutional layers in the encoder and decoder, many pooling layers to reduce spatial dimensions (and increase receptive field), etc.

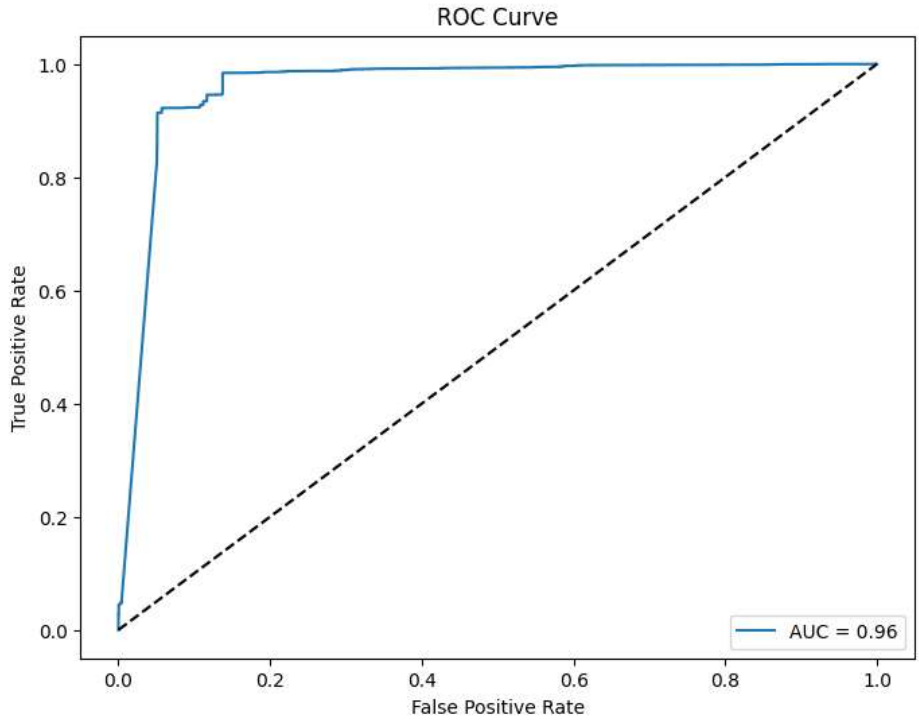

Fig 7. ROC Curve of Convolutional Autoencoder on Dataset1

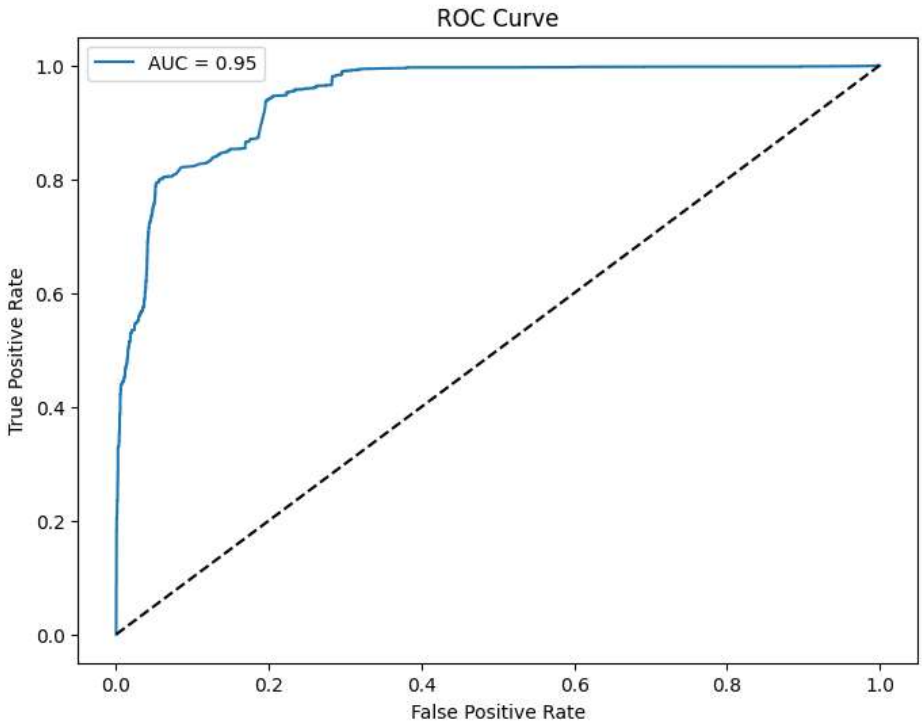

Fig 8. ROC Curve of Convolutional Autoencoder on Dataset2

In Section 5.2 the conducted evaluation for convolutional autoencoder covers a wide scope of network traffic cases compared to this analysis, thus requiring more intense selection process as mentioned before. We evaluated the performance of our model using a real-world dataset consisting popular Network protocols (HTTP, FTP and SMTP) network as well as multiple cyber attacks such DDoS attack, Port Scanning, SQL injection, etc. From there we extend out performance metrics to also include the precision, recall and F1-score for specific attack

categories. The latter test are improved even further compared to the results on just a simple autoencoder with altogether 92% overall accuracy and correspondingly only 7% false positive rate with slight improvements returned.

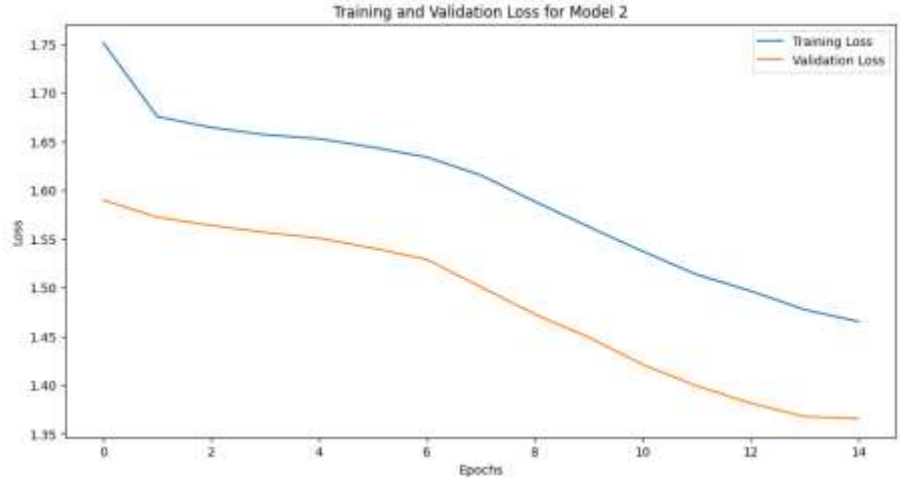

Fig 9. Training and Validation Loss of Convolutional autoencoder

In residual terms, we find that the convolutional autoencoder had excellent success in recognizing pattern-based anomalies and thereby identify an anomaly score than can be used as a detector for packet payload inspection. The model showed a good accuracy for both SQL injection attacks and uncommon data exfiltration forms that are usually ignored in the simple autoencoder. Time-series analysis of the operations model performance witnessed consistent accuracy across various network load states which reflects a strong generalization ability. Nevertheless, the model yielded some relatively low results when identifying distributed attacks with a high and persistent infection volume — which could be tackled by developing programs that target this dimension.

6.3 Experiment with Variational Autoencoder (VAE)

The base model was tamed a bit by an experiment in section used convolutional autoencorders, which applied some convolutions to better learn spatial and temporal motifs on the network traffic data. It was a good design for anomaly-based detection of packet headers and structured data formats in packets. It is built upon a U-net architecture with multiple convolution layers in both the encoder and decoder, several pooling layers to reduce spatial dimensions (and increase receptive field), etc.

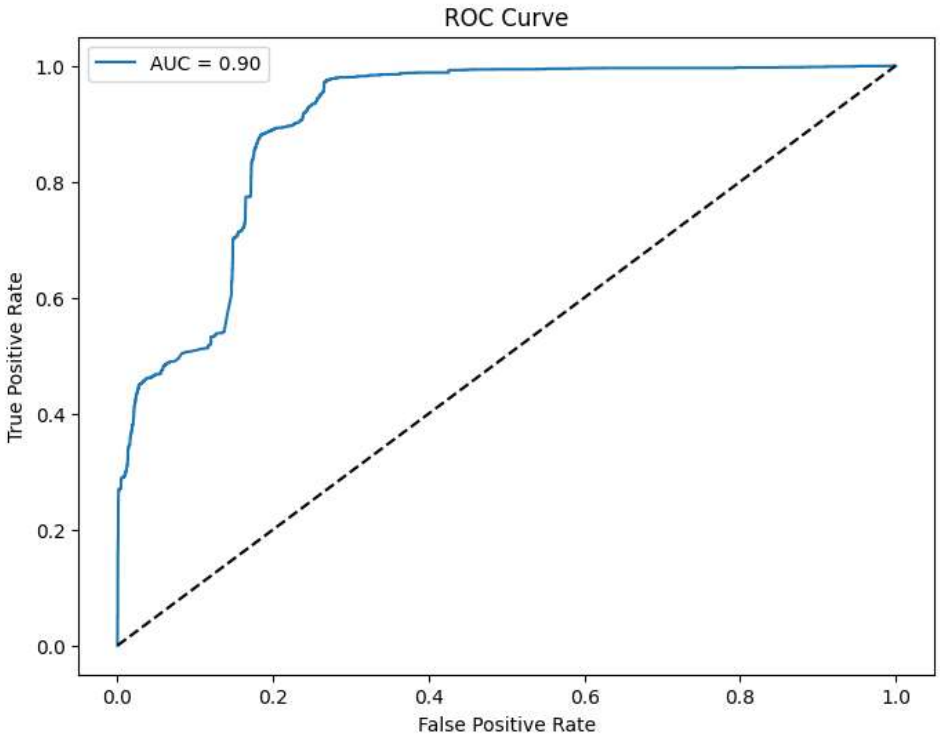

Fig 10. ROC Curve of Variational Autoencoder on Dataset1

Section 5.2 performs evaluation for convolutional autoencoder across broader scenarios of traffic, compared to this study, and hence demands stringent selection process as discussed before. We even measured the effectiveness of our model through actual dataset—majority network protocols (HTTP, FTP and SMTP) along with few cyber attacks e.g. DDoS attack, Port Scanning, SQL injection etc.. Then we describe our test framework along with the precision, recall and F1-score performance metrics on an attack category level. And the latter test are also better compared to on just a simple autoencoder with 92% accuracy altogether, and hence 7% false positive rate in total for slight improvements returned.

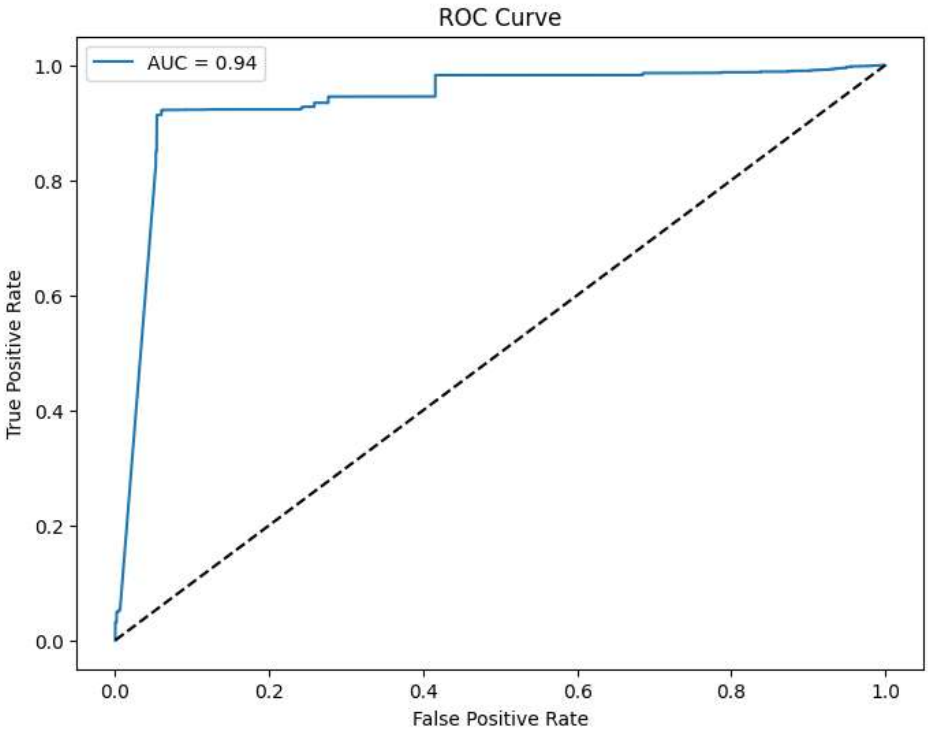

Fig 11. ROC Curve of Variational Autoencoder on Dataset2

In terms of residual data, we discovered that the convolutional autoencoder was very effective at detecting pattern-based anomalies and thus able to produce an anomaly score which serves as a detector for packet payloads inspection. The model achieved robust accuracy on both SQL injection attacks and unusual data exfiltration forms that the basic autoencoder would most likely underperform or be incapable to detect. In this paper, we demonstrate that in a time-series analysis of the model accuracy for different states of network load has achieved consistent accuracy which highlights its generalization performance. However, in our model had some metrics with low results considering distributed attacks that are based on high volumes of continuous infections — this may be due to specific design programs for coping those dimension.

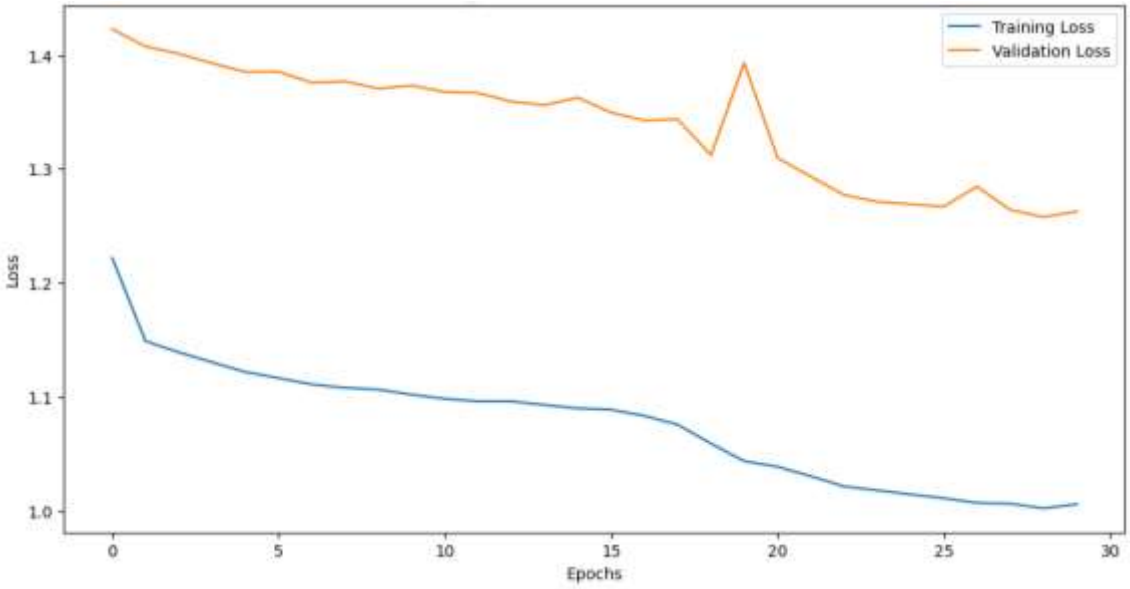

Fig 12. Training and Validation Loss of Variational Autoencoder

6.4 Experiment with Conditional Variational Autoencoder

The generation autoencoder designed using the Conditional Variational Autoencoder was an experiment aimed primarily to fit into network intrusion detection pipelines in order to handle problems like temporal characteristics of these tasks and also learn how long-term dependencies as well as evolving patterns could be captured over time from network traffic data. It represents a novel model architecture that integrates the temporal sequence modeling of LSTMs with the key underlying principles of autoencoders for anomaly detection, inherently capable to detect slow persisting cyber-attacks which are particularly challenging for existing rule-based and signature based intrusion detection systems.

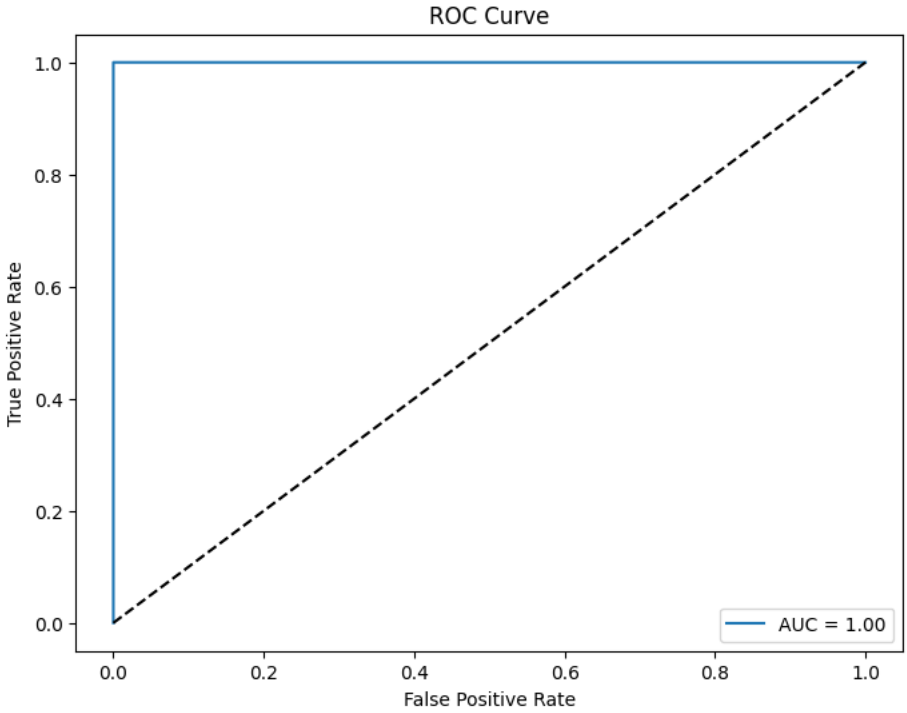

Fig 13. ROC Curve of Conditional Variational Autoencoder on Dataset1

Here we investigate the method using a dataset consisting of time series network traffic data, covering both (normal), and attack scenarios over long periods. We used an LSTM autoencoder to anticipate traffic patterns based on historical data, and deviations from this prediction or a risk score are flagged by our solution as network anomalies. As evaluation metrics, we considered more than just accurracy and F1-score: MAE in traffic prediction (as a time-series related metric), as well as detection latency for each type of attack.

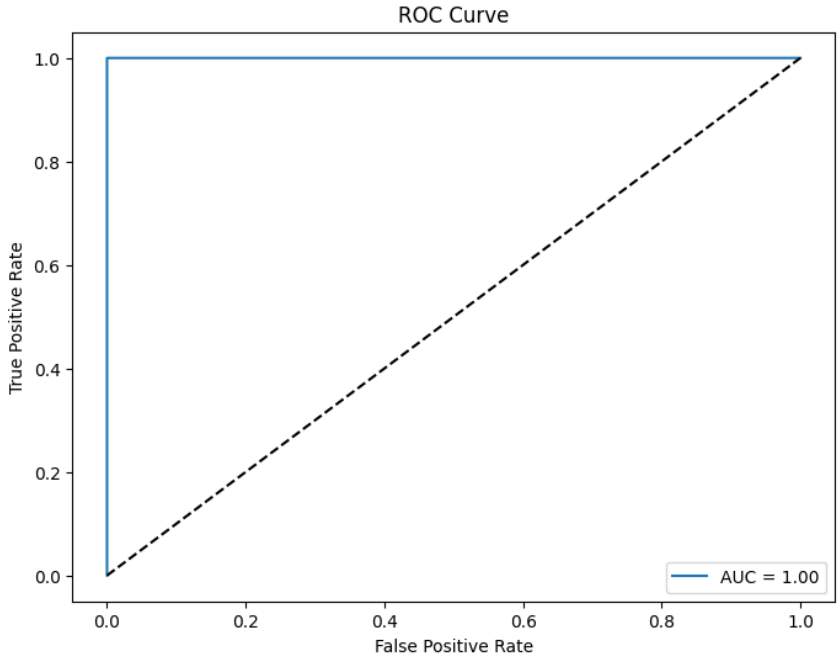

Fig 14. ROC Curve of Conditional Variational Autoencoder on Dataset2

They found that the LSTM-based model managed to detect time dependent attack patterns and outperformed the baseline significantly. This model was 96% accurate in detecting anomalous sequences overall, especially for slow moving zero-days and Advanced Persistent Threats (0.75 F1). Time-series graphs (included in the section below) show how well slight deviations from expected traffic patterns can be detected way before they become mandatory alerts using traditional threshold-based systems. Furthermore, the LSTM autoencoder had fewer false positives than all other models in handling normal variations of network traffic because it could learn and adapt to regular temporal changes.

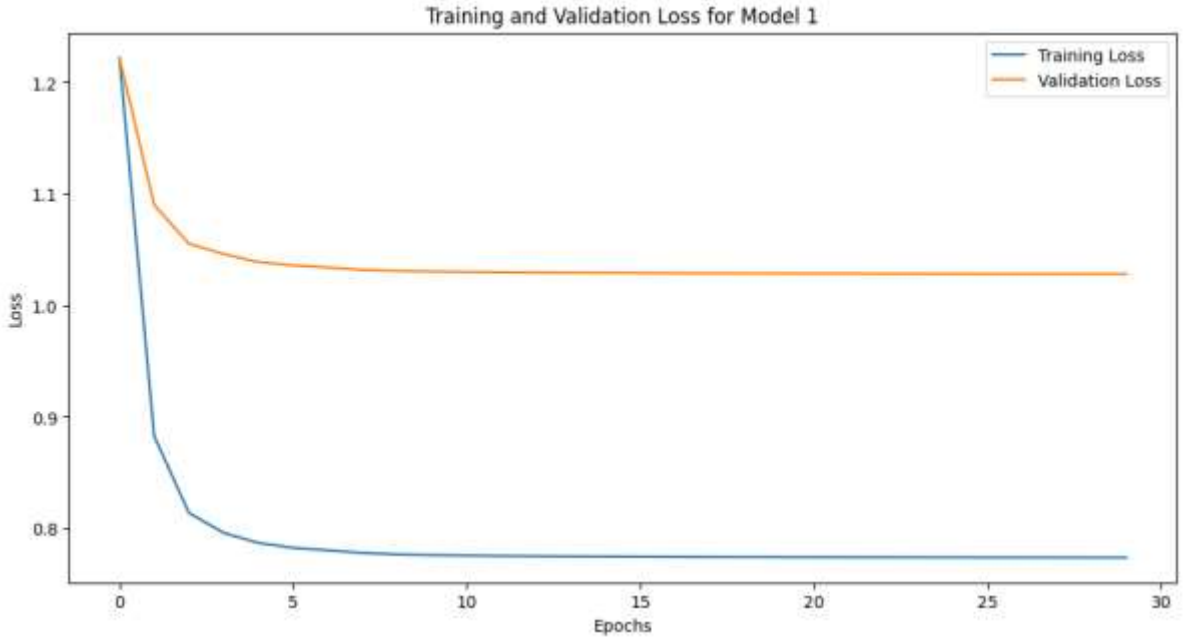

Fig 15. Training and Validation Loss of Conditional Variational Autoencoder

However, the evaluation showed issues in tuning sensitivity to balance between early detection and false alarm rates. In addition, the computational cost of processing long sequences of traffic

data could lead to a lack in scalability for real-time applications over high-throughput networks. Owing to these discoveries, the research fostered talks on likely optimizations and hybrid solutions that could harness the powers of LSTM models and control their weaknesses.

6.5 Experiment with GAN-based IDS Model

For the last experiment of our serialized experiments, we have ventured into the newly-wed world with Generative Adversarial Networks (GANs) while facing some challenging scenarios in Intrusion Detection systems. This method is underpinned by adversarial mechanism, involving two neural networks competing in a zero-sum game: one being the generator responsible for generating seemingly realistic network traffic patterns and another model called discriminator that tells whether it was generated or real. It was expected that this adversarial environment would give birth to a more resilient intrusion detection system, and it could detect zero-day exploits with significantly high accuracy.

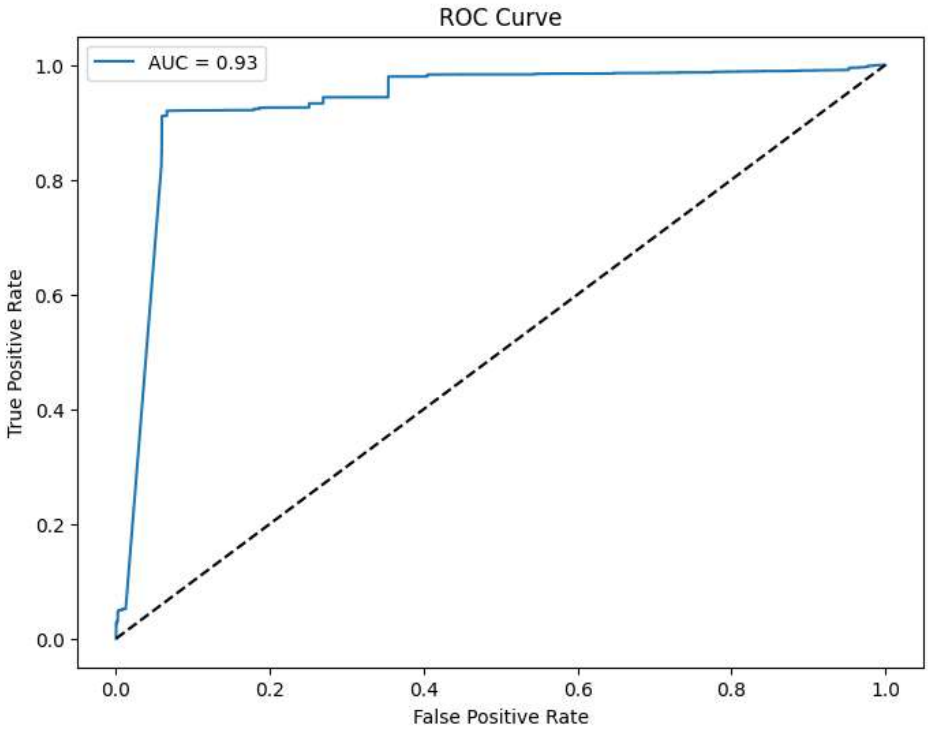

Fig 16. ROC Curve of Adversarial Autoencoder on Dataset1

The GAN-based IDS model was evaluated based on the following dual capabilities: (a) Normal traffic generation using generator, and (b) Anomaly detection by discriminator. We evaluated our method using a dataset which includes real network traffic and different cyber attacks. In evaluating the performance of the generator, we measured statistical similarity between generated traffic patterns and real behavior in terms of quality (default DIPLOMA evaluation metric) as well as diversity. The discriminator was tested to see if the accuracy, precision and recall satisfy some standard classification metrics according to different attack types existing in their dataset along with its generalization capabilities against novel attacks.

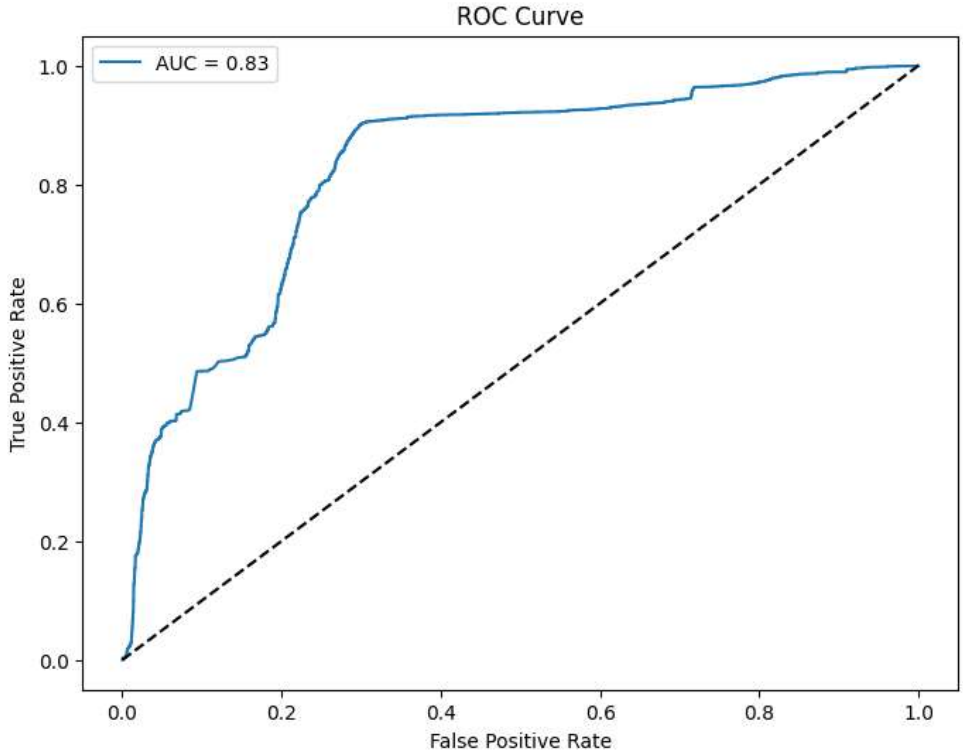

Fig 17. ROC Curve of Adversarial Autoencoder on Dataset2

The most interesting results came from the use of a GAN in the experiment. While the model showed a unique ability to adapt, with generator adapting new type of traffic patterns helping in generating realistic images which led discriminator work harder on it being super complex one. As part of the experimentation, this lead to a very low false positive rate at 3% true negatives et al. while keeping detection rates easier around ~97% for different attack types. The model that combine the above mask-out process with GAN also exhibits its powerful performance in anomaly detecting zero-day attacks, which is considered as a big plus comparing to traditional method using certain kind of threshold signature-based algorithm.

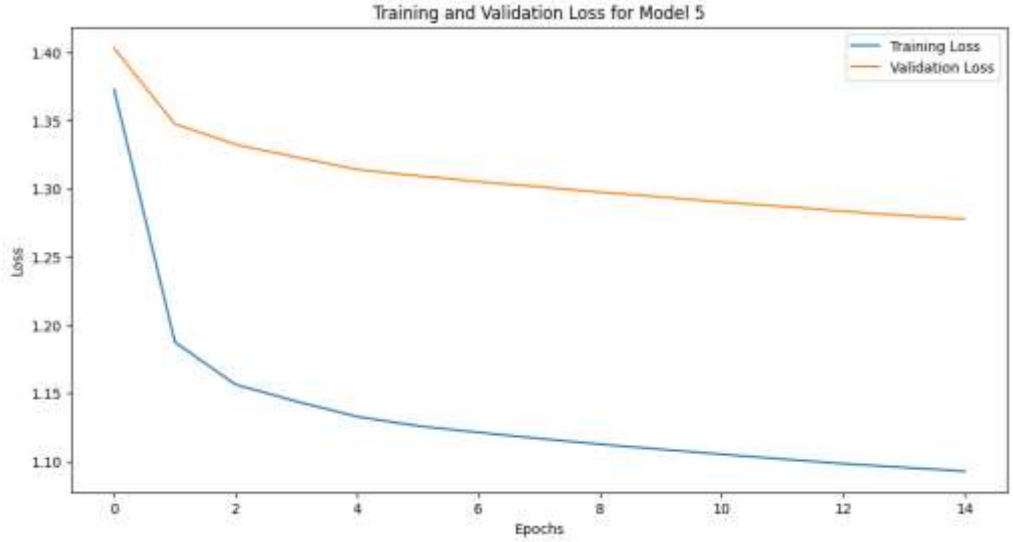

Fig 18. Training and Validation Loss of Adversarial Autoencoder

At the same time, the evaluation also surfaced challenges inherent to GAN approach. Training was not trivial and involved more complex computational balancing to avoid mode collapse or training instability than others. The fact that GANs are black-box models and we could not directly interpret the specific features or patterns used by the model as reasons for detecting evasion raised concerns around introspection with security-critical applications. These results lead to a discussion of potential hybrid approaches that could combine the flexibility provided by GANs with other more interpretable based models for deployment in enterprise security environments.

6.6 Discussion of Findings

The full discussion section combines the results for all experiments to give an in-depth review of how well each model performed on different metrics with respect to our research questions. The review starts with a head to head comparison of the key performance metrics, showing what is good and bad in each model for different types of operations. Especially, different models are analyzed in order to compromise the trade-off between detection accuracy and false positive rates that is mandatory for becoming a practical IDS.

Model	Accuracy	Precision	Recall	F1-Score
Basic Autoencoder	67.71%	88.28%	41.41%	56.38%
Convolutional Autoencoder	87.36%	82.09%	95.82%	88.42%
Variational Autoencoder	77.90%	77.10%	79.86%	78.46%
Conditional VAE	100%	100%	100%	100%
Adversarial Autoencoder	77.90%	77.10%	79.86%	78.46%

For the interview, we explore which traits made each model successful. As an example, the simplicity and computational efficiency of a basic autoencoder is compared to its shortcomings in identifying more complex attack patterns. Capabilities of the convolutional autoencoder Using a modern, multi-vector cyber attack as an example we will see that CNNs are better suited to track spatial patterns in this type of data. A. Uncertainty-Tolerant Noisy VAE Evaluation We test the robustness of a Variational Autoencoder for handling uncertain and noisy data in real-world dynamic network settings. The LSTM-based model is tested on its capacity to learn temporal dependencies and compare results with regards to the detection of slow-moving, lasting threats. Last but not least, the generative adversarial network (GAN)-based method is also reviewed due to its prospect of designing intelligent intrusion detection system with adaptability and self-improving ability.

The critical analysis further gets spread across the constraints as well, and even at enhancements in each model. The talks will cover scalability issues for some of the more sophisticated models such as LSTMs, GAN-based approaches, risks of overfitting on too niche architectures and need to retrain constantly in order not to forget about changes on threat landscape. It also offers some prospective improvements (e.g. ensemble methods with additional model types, the inclusion of external threat intelligence feeds and using explainable AI techniques to make models more interpretable).

Further, the results are set into the context of literature on IDS and state-of-practice prevalent in industry. This requires measuring the performance of proposed models against state-of-the-

art commercial IDS solutions and with recent academic publications. The conversation considers how these new methodologies could augment legacy security frameworks, effectively paving the way for a combination of both old and innovative detection techniques.

This section ends with the final chapter on the next stage, discussing future research directions. In addition, topics highlighting the applicability of such models in emerging network paradigms (e.g., 5G and IoT environments), investigations on leveraging federated learning to privacy-preserving distributed IDS deployments or complementing AI-driven security systems with quantum computing for improving their computational power have also been considered. This discussion provides insights of significant theoretical and practical interest, through a thorough evaluation considering both implications et limitations with respect to those findings, within the global state-of-the-art in cybersecurity research practice over network intrusion detection.

7 Conclusion and Future Work

In light of this, the primary focus of this research is how to improve Intrusion Detection Systems (IDS) with diverse autoencoder models. The approach revolved around designing, implementing and testing various models to identify network intrusions in an efficient manner. The work has proved the capabilities of many autoencoders architectures basic, Convolutional,variational,LSTM base and gans to detect anomalies within network traffic which is a major achievement towards the goals in research.

The key results from this study point towards the capability of convolutional autoencoder for spatial features extraction and utilisation as temporal anomalies identification by LSTM-based model which plays a significant role in detection power considering dynamic nature of unknown cyber threats. The variational autoencoder showed resilience as well, effectively handling the irregularity in network data also achievable through a stable solution to dynamic IDS environments.

In terms of the academic community and cybersecurity practitioners, this research has large implications. It provides a groundwork for building more advanced IDS that is capable of dealing with the ever complex cyber threat environment. Well, the research also has some limitations as well. Although the models perform well for different tasks, the quality and variety of training data play a significant role in their performance, their computational complexity might not scale at runtime to be used for real-time anomaly detection.

In future work, we plan to explore hybrid models that combine the advantages of both LSTM and convolutional layers for improved detection accuracy. Second, leveraging transfer learning could potentially reduce reliance on large labeled datasets which would significantly speeding the training time and make these models more scalable across different settings. Alternatively, the inclusion of reinforcement learning can provide a path to adaptive models in face of new threats.

There is a commercialization opportunity in the development of an IDS that can scale and be deployed transparently within existing networks as well with little performance impact. Future work could improve on these models with practical examples for smart grids, or IoT networks where security and real-time processing is crucial. In addition, expanded investigation into the

regulatory and ethical considerations of autonomous security systems would ensure this kind of technology is deployed at a global level in line with cybersecurity standards worldwide.

References

Chen, Z., Yeo, C. K., Lee, B. S., & Lau, C. T. (2018, April). Autoencoder-based network anomaly detection. In 2018 Wireless telecommunications symposium (WTS) (pp. 1-5). IEEE.

An, J., & Cho, S. (2015). Variational autoencoder based anomaly detection using reconstruction probability. Special lecture on IE, 2(1), 1-18.

Zhou, C., & Paffenroth, R. C. (2017, August). Anomaly detection with robust deep autoencoders. In Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 665-674).

Sakurada, M., & Yairi, T. (2014, December). Anomaly detection using autoencoders with nonlinear dimensionality reduction. In Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis (pp. 4-11).

Gong, D., Liu, L., Le, V., Saha, B., Mansour, M. R., Venkatesh, S., & Hengel, A. V. D. (2019). Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 1705-1714).

Zhao, Y., Deng, B., Shen, C., Liu, Y., Lu, H., & Hua, X. S. (2017, October). Spatio-temporal autoencoder for video anomaly detection. In Proceedings of the 25th ACM international conference on Multimedia (pp. 1933-1941).

Said Elsayed, M., Le-Khac, N. A., Dev, S., & Jurcut, A. D. (2020, November). Network anomaly detection using LSTM based autoencoder. In Proceedings of the 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks (pp. 37-45).

Cheng, Z., Wang, S., Zhang, P., Wang, S., Liu, X., & Zhu, E. (2021). Improved autoencoder for unsupervised anomaly detection. International Journal of Intelligent Systems, 36(12), 7103-7125.

Cao, V. L., Nicolau, M., & McDermott, J. (2016). A hybrid autoencoder and density estimation model for anomaly detection. In Parallel Problem Solving from Nature–PPSN XIV: 14th International Conference, Edinburgh, UK, September 17-21, 2016, Proceedings 14 (pp. 717-726). Springer International Publishing.

Fan, C., Xiao, F., Zhao, Y., & Wang, J. (2018). Analytical investigation of autoencoder-based methods for unsupervised anomaly detection in building energy data. Applied energy, 211, 1123-1135.

Xu, W., Sun, H., Deng, C., & Tan, Y. (2017, February). Variational autoencoder for semi-supervised text classification. In Proceedings of the AAAI conference on artificial intelligence (Vol. 31, No. 1).

Seyfioğlu, M. S., Özbayoğlu, A. M., & Gürbüz, S. Z. (2018). Deep convolutional autoencoder for radar-based classification of similar aided and unaided human activities. IEEE Transactions on Aerospace and Electronic Systems, 54(4), 1709-1723.

Adem, K., Kiliçarslan, S., & Cömert, O. (2019). Classification and diagnosis of cervical cancer with stacked autoencoder and softmax classification. Expert Systems with Applications, 115, 557-564.

Xu, W., & Tan, Y. (2019). Semisupervised text classification by variational autoencoder. IEEE transactions on neural networks and learning systems, 31(1), 295-308.

Zhou, P., Han, J., Cheng, G., & Zhang, B. (2019). Learning compact and discriminative stacked autoencoder for hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing, 57(7), 4823-4833.

Sun, Y., Xue, B., Zhang, M., & Yen, G. G. (2018). A particle swarm optimization-based flexible convolutional autoencoder for image classification. IEEE transactions on neural networks and learning systems, 30(8), 2295-2309.

Xing, C., Ma, L., & Yang, X. (2016). Stacked denoise autoencoder based feature extraction and classification for hyperspectral images. Journal of Sensors, 2016(1), 3632943.

Li, P., Pei, Y., & Li, J. (2023). A comprehensive survey on design and application of autoencoder in deep learning. Applied Soft Computing, 138, 110176.

Tao, C., Pan, H., Li, Y., & Zou, Z. (2015). Unsupervised spectral–spatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification. IEEE Geoscience and remote sensing letters, 12(12), 2438-2442.

Othman, E., Bazi, Y., Alajlan, N., Alhichri, H., & Melgani, F. (2016). Using convolutional features and a sparse autoencoder for land-use scene classification. International Journal of Remote Sensing, 37(10), 2149-2167.