Detecting Network Intrusion Anomalies with RNN (LSTM)- Based Deep Learning Models

Mr. Rohith D, Mr. T Abid Hussain, Mr. K Harshavardhan,
Mr. Sharath PD, Mr. Mohammed Jawad Ahmed
Under the guidance of,
Mr. Yamanappa
Assistance Professor
School of CSE & IS ,
Presidency University Bengaluru

Abstract: As more and more network devices and services are used, there is a growingneed for security measures as hackers attempt to take down or steal data from target systems. One of the key components of network perimeter security is the intrusion detectionsystem (IDS), which looks through operating system logs or network traffic packets to identify threats. Although previous research has shown the effectiveness of several machine learning algorithms, relatively few of these studies have made use of the time-series information included in network traffic data. Neural network-based methods have notincorporated category data either. In this paper, we offer models for network intrusion detection based on categorical information using the embedding technique and sequential information using the long short-term memory (LSTM) network. Using the extensive networktraffic dataset KDD CUP 99, we have tested the models. The findings of the trial confirm that the suggested strategy improves performance, with a 99.72% binary classification accuracy.

Keywords: Machine Learning, Deep Learning ,Network Intrusion Detection, Long Short-Term Memory ,Feature Selection

Introduction

The relevance of network security has increased due to the growing number of network devices and services. The need for preventative measures is growing ashackers attempt to cripple or steal data from network-connected computer systems. These attacks include, for instance, sending harmful files and taking advantage of targets' security flaws. Hackers engage in network activity by interacting with the target system in such assaults. One of the key components ofnetwork perimeter security is the Network Intrusion Detection System (NIDS), which monitors these actions and sounds an alarm. Specifically, when it detects malicious network activity, NIDS triggers alarms by analysing the header and payload data of incoming and departing network packets.

level of feature extraction, but differ in their ability to recognise malevolent behaviour. Individual packets in network activities are summarised into high- level events like sessions using the feature extraction technique. The feature values that define each summarised record are what make up the high level event. Next, in the conventional NIDS development process, security specialists determine attack patterns and determine the threshold ranges for each feature.In contrast, a model is automatically trained to identify patterns ofmalicious activity from a dataset in the machine learning technique.Because machine learning-based techniques have the potential to identify more complexpatterns in a large-scale dataset, they have recently garnered more attention than traditional techniques.

Although a lot of academics have experimented with different machine learningtechniques, there hasn't been much focus on time-series information of network traffic data.As network activity happens in real time, using sequential data in machine learning models ought to produce more thorough analyses—that is, if themodel has the computational power to handle this kind of extra data. Long short- term memory (LSTM) or gated recurrent units (GRU) are common RNNs. Recurrentneural networks (RNNs) are able to capture temporal dependence in data, which has led to significant advancements in the fields of speech recognition and machine translation.

Neural network based network intrusion detection systems have overlooked categorical information in addition to time dependence. Non-numeric (or symbolic) characteristics seen in network trace data, such as protocol type, state, and service, are referred to as categorical information. Even though thesecharacteristics are essential for identifying harmful pattern activity, they could not be accepted as input by conventional neural network techniques. Becausewords are symbols, categorical features are frequently encountered in natural language processing (NLP). Various feature embedding (or word embedding) strategies, such as language models and neural machine translation, are available to handle symbolic words in NLP applications.

In this paper, we propose to use LSTM and feature embedding to create in- trusion detection models. Specifically, we use feature embedding to use categorical features and LSTM to capture sequential information of network trace data. We use the KDD(Test&Train) dataset, which is an up-to-date dataset for network intrusion detection, for evaluation after reviewing open datasets. To capture temporal dependence for intrusion detection, we assume that records are organised in a timely manner. Through experiments, we demonstrate that LSTM can effectively describe sequential structures for NIDS and that feature embedding can enable theneural network models to access category features. LSTM with feature embedding,as compared to other machine learning approaches, finally produces a significant improvement in detection performance after numerous experiments with different options and hyperparameters. Our binary classification accuracy on the UNSW- NB15 dataset is 99.72%.

The rest of this paper is organized as follows. In Section 2, relevant works and contexts are discussed. We provide novel models for network intrusion detection inSection 3. The experiment results are presented and analyzed in Section 4, followed by Section 5 where we conclude the paper.

Background

2.1 Network Intrusion Detection Data

A popular dataset for identifying network intrusion assaults is KDD Cup ’99. It includes subsets for testing and training, making it possible to create and assess intrusion detection systems. The dataset is an important tool for teaching computers to identify and react to possible security risks because it captures a range of network activity, both benign and malevolent. This datasetis frequently used by academics and industry professionals to benchmark andimprove intrusion detection techniques’ efficacy in the ever-changing field of network security.

2.2 Network Intrusion Detection Method

IDS has two detection mechanisms according to de nitions of malicious activity Signature-based detection mechanism detects malicious activities, and rec-ognizes behaviors that match the attacks. In contrast, anomaly-based detection
mechanism de nes normal activity, and recognizes behaviors that deviate from
the normal ones. The former mechanism is more compatible with attacks that are already known and shows

low false-positive rate compared to the latter. Onthe other hand, the latter has potential to recognize unknown attacks, but it cansure from high false-positive rate.

IDS offers two detection methods based on how malicious activity is defined.Malicious actions are detected via a signature-based detection technique, which also identifies behaviours consistent with attacks. Anomaly-based detection mechanisms, on the other hand, distinguish between normal activities and abnormal behaviours. Compared to the latter, the former approach exhibitsa lower false-positive rate and is more compatible with known assaults. Conversely, the latter may be able to identify unidentified attacks, but it may also suffer from a high false-positive rate.

Expert-centered and machine learning-based NIDS development methodologies correspond to the two detection processes. Experts in security writesignatures in an expert-centered manner. For instance, the well-known open- source NIDS project Snort allows users to build rules that dictate how it analyses network packets and generates alarms. This method necessitates specialistknowledge or sets of rules. In contrast, machine learning models automatically learnsignatures in the latter method. Additionally, a dataset with a large volume of data and labels indicating the type of assault on each datum is needed.

After publication of KDD Cup 99' there have been many research works to applymyriad of machine learning techniques to the dataset. Suleiman et al. applied various classical machine learning algorithms such as Random Forest,
K-nearest neighbor, and Support Vector Machine .Among the experiments, J48 andK-NN algorithms were proposed as the most suitable models with high efficiency and accuracy. Moustafa et al. experimented anomaly-based detection method based on geometric area analysis using trapezoidal area estimation . Numerous studies have been conducted to apply a wide range of machine learning techniquesto the UNSW-NB15 dataset since its publication. Suleiman et al. used a variety of traditional machine learning techniques, including Support Vector Machine, Random Forest, and K-nearest neighbour.The experiments revealed that the J48 and K-NN algorithms were the most accurate and efficient models. Using trapezoidal area estimation and geometric area analysis, Moustafa et al. experimented with an anomaly-based detection approach.

In the meantime, Papamarztivanos et al. presented a novel decision tree and genetic algorithm-based method to NIDS. In their work, they generated detection rules that make up a decision-tree model using a genetic approach. After testing the final model on UNSW-NB15, it demonstrated good performance in identifying both common and uncommon assaults within the dataset.

While earlier research has shown a variety of machine-learning-based NIDS, the majority of it ignored sequential data. As an exception, Staudemeyerused the KDD99 dataset and a long short-term memory (LSTM) network to enhance classification performance. Furthermore, Kim and colleagues tested an LSTM-RNN model on KDD99 and observed a significant improvement in performance. Nevertheless, the KDD99 dataset—which reflect modern assaultbehaviors—was used in their trials. Furthermore, categorical features were employed in earlier LSTM models as though they were continuous ordinal data. Similar to word embedding in NLP, categorical characteristics require feature embedding.

2.3 Long Short-Term Memory

We briefly examine LSTM and recurrent neural networks (RNN) in this section.To obtain additional details, readers are directed to

An RNN is a modified neural network that continuously modifies its internal state. RNNs areable to capture temporal features in sequential data and memorise previous inputs by creating circular connections within the network. However, because of the disappearing gradient issue, RNNs are only able to hold onto memory for a limited number of time steps. The LSTM introduces three gates—input, forget, and output—around unique memory units known as cell states (ct) in order to overcome the vanishing gradient problem. As seen in Fig., the gates regulate how the cell states are updated.

In LSTM, inferences for the cell states ct and hidden states ht are given by

$$ct = ft \;\; ct \;\; 1 + it \;\; \tanh(U\,ht \;\; 1 + W\,xt + b); \quad (1)$$

$$ht = ot \;\; \tanh(ct); \quad (2)$$

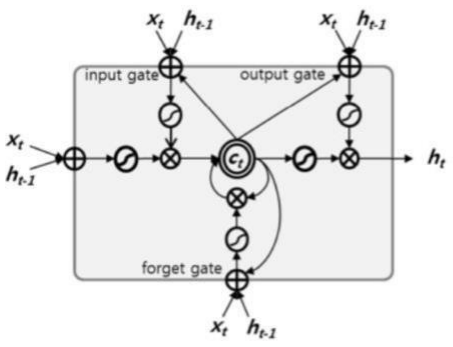


Fig. 1. Structure of LSTM Cell.

Adopted from where indicates the element-wise multiplication operation, and the three gates are de ned by

$$it = \;(Wixt + Uiht \;\; 1 + bi); \quad (3)$$

$$ft = (Wf\ xt + Uf\ ht\ 1 + bf); \qquad (4)$$

$$ot = (Woxt + Uoht\ 1 + bo): \qquad (5)$$

The sigmoid function is shown here, with the parameters being U, W, and b. Peephole connections are not added for implementation convenience without performance impact.

Intrusion Detection based on LSTM

Depending on their nature, network intrusions follow certain patterns. These patterns typically emerge across numerous packets rather than in a single one.Unfortunately, the majority of the earlier machine learning techniques fornetwork intrusion detection systems were unable to address this feature and were unable to identify patterns that appeared in several packets.

Multi-layer Perceptron (MLP), for instance, ignores temporal dependency and uses only one packet to identify intrusions. Actually, it would be very difficult toidentify a denial-of-service (DoS) attack using machine learning (MLP) since DoS attacks aim to bring down a server by sending large numbers of packets that are not all that different from regular packets. This problem may not only apply to DoS attacks but also to other kinds of attacks. Therefore, dealing withmany packets instead of a single packet is important for more precise intrusiondetection.

In this study, we employ LSTM to determine if the current packet, in light ofthe preceding packets, is normal. As shown in Fig., the inputs of the LSTM arethe current packet and the packets from the past.

Still, there are a number of different approaches to building our network designor training the network. Initially, either the current packet's original label or all of the labels for both previous and current packets can be used to train the network. Secondly, the network can be built for many classi cations ('normal' or various attackkinds) or binary classi cations ('attack' or 'normal'). We can train a model for severalclassifications, including binary classification, and categorise all attack kinds as a single class "attack."

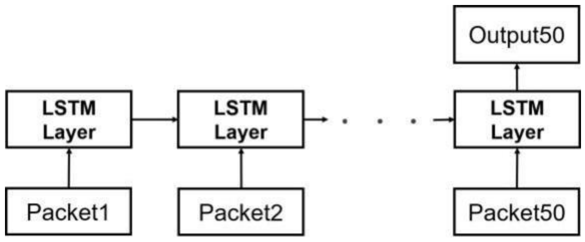


Fig. 3. Intrusion detection through multiple packets

Further, the network can be constructed with or without embedding layer whichis for categorical features.

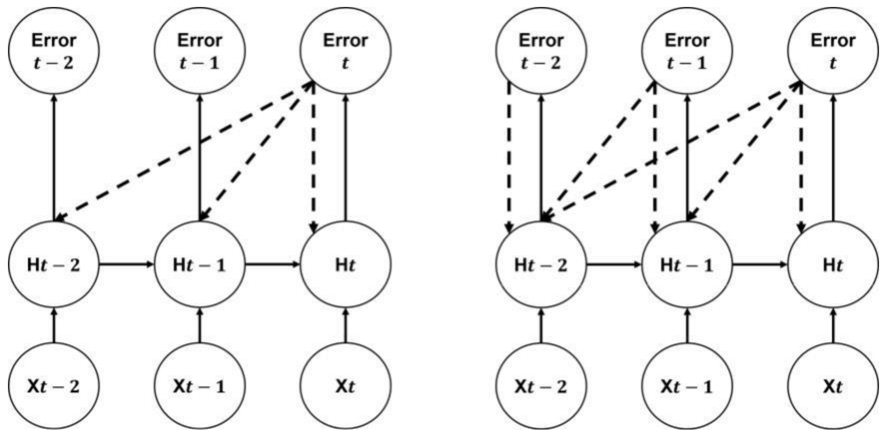
3.1 Many-To-Many Train vs. Many-To-One Train

As shown in Fig. 3, NIDS is to perform many-to-one classification given sequentialpackets, where the current input is classified using sequential packets. In other words, just the last step's output is decided given several input steps. RNNs, such as LSTM, function by processing one input packet at a time and producing a prediction output at each time step. As a result, many-to-one (M2O) training—alsoknown as training the model just using the final error—is a natural approach in many-to-one classification, as shown in Fig. 4(a). However, if the labels areaccessible, all of the errors can be used for training as shown in Fig. 4(b), since the labels for the earlier packets contain some information that speeds up the training process. It is known as instruction from many to many (M2M). In other words, the M2M method learns both the attack type of the target packet and the attack type ofthe packets that came before it. In experiments, we compare two training strategies.

3.2 Multi Classes to Binary Class Detection

Network intrusions can take many different forms, hence multi-classification may beuseful. However, there are situations when it would be fascinating to categorise a packet as abnormal or normal. There are two methods for that. Prior to training, several attack types of packets can be transformed into "attack." Subsequently, a packet is classified into binary outcomes by training the model. Alternatively, the model can be trained for multiple classification and the prediction results can be combined into binary classification results, as shown in Fig., without combining several attack kinds into a single class "attack."In other words, the model essentiallyconducts multi-classification; nonetheless, it is classified as a "attack" if the prediction corresponds to one of the attack types. We refer to this classification asmulti-to-binary.

3.3 Detection With Feature Embedding



(a) (b)

Fig. 4. Two learning methods: (a) M2O training learns only the last output, and (b) M2Mtraining learns all the outputs in the sequence.

Many nominal (or categorical) properties, such as protocol and state, are present in network packets (or connections). Every nominal characteristic describes the status and function of the packet. Different packets with differentnominal values can be distinguished from one another by the characteristics ofeach attribute. Nonetheless, a feature's functions can cause different values tobehave quite similarly. Consequently, it might not be sufficient to represent packets by merely substituting the one-hot encoding vector for the nominal features. In order to make each nominal feature a suitable vector in a continuous vector space according to the attack kinds, we utilise the feature embedding technique.

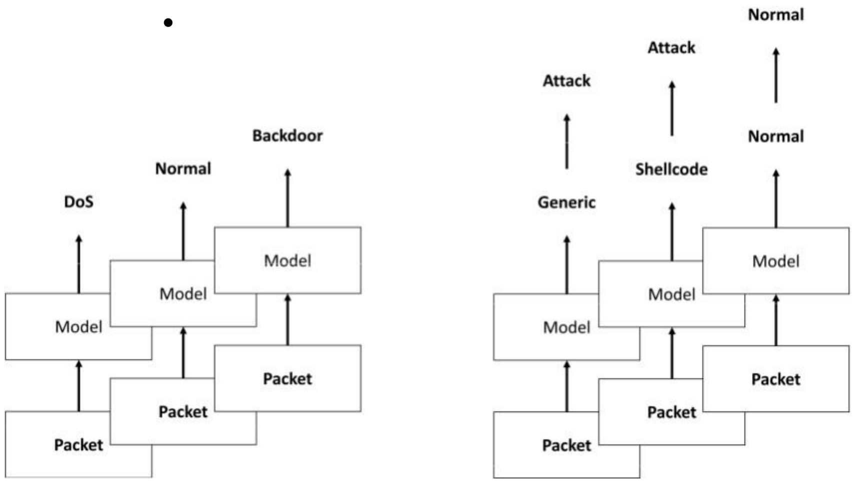
Every nominal feature is initialised to random vectors, much like in NMT. Upon training, the vectors converge at the right locations based on the types ofattacks on the packet. For instance, after training, TCP and UDP are situated near to one other in the vector space since they frequently occur in the same attack type. By using relationships between nominal features, our model mightenhance the detection performance with all embedded category features.

Experiments

4.1 Dataset

A popular dataset for identifying network intrusion assaults is KDD Cup '99. It includes subsets for testing and training, making it possible to create and assess intrusion detection systems. The dataset is an important tool for teaching computers to identify and react to possible security risks because it capturesa range of network activity, both benign and malevolent. This dataset is frequently used by academics and industry professionals to benchmark and improve intrusion detection techniques' efficacy in the ever-changing field of network security.

•



(a) (b)

Fig. 5. M2B classi cation: (a) The model is trained to perform multi-classi cation, (b) Theprediction results are merged into binary classi cation results.

Table 1 lists the nine attack types and normal type of UNSW-NB15. There are37 numerical features, 2 binary, and 3 nominal features in the dataset. The dataset is divided into two subsets: 82,332 packets for testing and 175,341 packets for training. Ten percent of randomly chosen samples from the trainingset are set aside and utilised for validation.

Table 1. KDD Cup 99' ataset Attack Type

Category	Train	Test
Total Records	175,341 (100%)	82,332(100%)
Normal	56,000(31.94 %)	37,000(44.94 %)
Analysis	2,000(1.14%)	677(0.82%)
Backdoor	1,746(1.00%)	583(0.71%)
Dos	12,264(6.99 %)	4,089(4.97%)
R2L	33,393(19.04 %)	11,132(13.52 %)
Probe	18,184(10.37 %)	6,062(7.36%)
Generic	40,000(22.81 %)	18,871(22.92 %)
U2R	10,491(5.98 %)	3,496(4.25%)
Shellcode	1,133(0.65%)	378(0.46%)
Worms	130(0.07%)	44(0.05%)

Advantages of KDD(test and train) Dataset:-

KDD, which stands for Knowledge Discovery in Databases, typically involves creatingseparate training and test datasets to develop and evaluate machine learning models. Hereare the advantages of having distinct training and test datasets:

1. **Model Evaluation:** The test dataset allows you to assess how well your model performs on unseen data. This evaluation is crucial to understanding how your model might perform inthe real world.
2. **Prevention of Overfitting:** Having separate datasets helps prevent overfitting, where a model learns to perform well on the training data but fails to generalize to new, unseen data.The test dataset ensures you can assess generalization performance accurately.
3. **Hyperparameter Tuning:** The training dataset is used to train the model, and you can use

techniques like cross-validation to optimize hyperparameters. The test dataset remains untouched during this process to avoid information leakage.

4. **Bias and Variance Estimation:** By comparing the performance of a model on the training and test datasets, you can gain insights into issues related to bias (underfitting) or variance (overfitting) and adjust your model accordingly.

5. **Enhanced Model Robustness:** A model that performs well on both the training and test datasets is likely more robust and better equipped to handle new, unseen data, increasing its real-world utility.

6. **Unbiased Performance Estimation:** The test dataset provides an unbiased estimate of the model's performance on new data, helping in making informed decisions about model deployment or improvements.

7. **Quality Assurance:** Having separate datasets ensures the quality of the model by confirming that it's not simply memorizing the training data but genuinely learning patterns and features that generalize well to new instances.

8. **Iterative Model Improvement:** As you iterate through model development, having separate datasets allows you to retrain and test the model iteratively, improving its performance over time.

4.2 Model Architecture

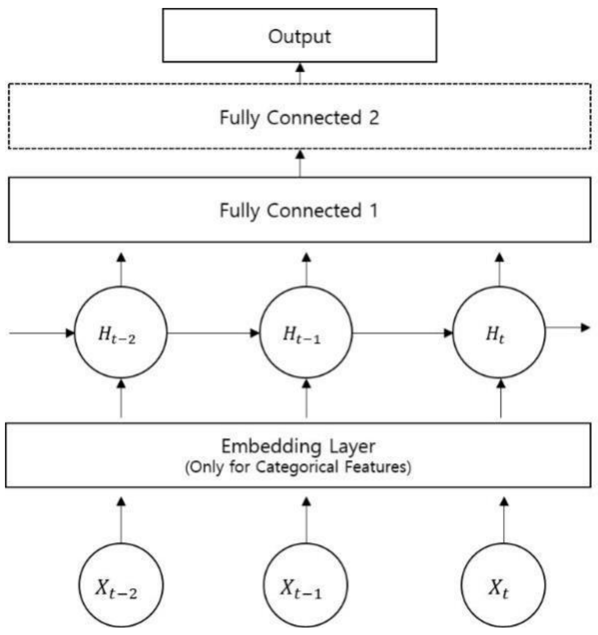


Fig. 6. Model Architecture: embedding, LSTM, and fully connected layers. ‘Fully Con- nected 2’ is used only for binary classi cation.

Three different kinds of layers make up our model: fully linked, LSTM, and embedding layers. Only nominal features of an input are used in the embedding layer; continuous features are ignored. Three nominal characteristics correspond to vectors in dimensions of three, three, and two, respectively. Concatenating these output vectors with continuous features allows them to move to the next layer of the model. There are 100 nodes in the hidden state that make up the LSTM layer. With dropout, the completely connected layer hasa size of 50. For non-linear transformation, leaky ReLU is used as the activationfunction. A second fully linked layer with the same number of nodes is added inthe event of binary classification. The layer is shown in the dotted line as only functioning in binary classification scenarios.

4.3 Evaluation Metrics

As evaluation metrics, we used accuracy (AC) and F1-score (F1). Given true positive (TP), true negative (TN), false positive (FP), and false negative (FN),

AC and F1 are respectively calculated by

TP + TN

AC = TP +TN +FP +FN

F1= $\underline{\quad 2P\ R}$

R

where P and R stand for precision and recall, respectively as follows.

T P

P = TP+FP

T P

R= TP +FN

(6)

(7)

(8)

(9)

As the harmonic mean of precision and recall, F1-score provides a better evalu-ation measure than accuracy especially for imbalanced data.

4.4 Experiment Results

We assess a variety of training configuration combinations on LSTM using feature embedding. First, as previously mentioned, the LSTM model is trainedin two different methods. One is learning from each output's errors (M2M), whilethe other is simply learning from the most recent output's errors (M2O).Furthermore, we incorporate "multi-classi cation to binary-classi cation" (M2B)into binary-classi cation. This involves training a multi-classi cation model and converting all harmful labels and model outputs to a single label, or "attack." Finally, each model is subjected to feature embedding (EMB).

Table 2. Binary-classi cation LSTM Model results for test data. Validation results are in the parenthesis.

Model	Sequence Length	Accuracy	F1 Score
ANN	-	81.91	95.2
RepTree	-	88.95	-
Random Forest	-	90.3	92.4
MLP	-	83.55 (94.00)	86.89
LSTM(M2M)	110	98.68 (99.88)	99.16
LSTM(M2O)	310	98.49 (97.99)	98.90
LSTM(M2M M2B)	130	98.29 (99.84)	98.43
LSTM(M2O M2B)	210	99.42 (98.07)	99.47
LSTM(M2M + EMB)	270	99.72 (99.97)	99.75
LSTM(M2O + EMB)	90	99.52 (97.82)	99.56
LSTM(M2M M2B + EMB)	110	99.53 (99.93)	99.67
LSTM(M2O M2B + EMB)	110	98.83 (98.02)	98.93

MLP model and LSTM models have apparent di erences in terms of perfor-mances as summarized in Tables .The MLP model shows the accuracy o

Table 3. Multi-classi cation LSTM Model results. Validation results are in the paren- thesis.

Model	Sequence Length	Accuracy
Random Forest	-	75.5
RepTree	-	81.28
MLP	-	72.81 (79.32)
LSTM(M2M)	20	84.78 (85.52)
LSTM(M2O)	250	83.45 (82.72)
LSTM(M2M + EMB)	30	86.98 (88.50)
LSTM(M2O + EMB)	150	85.93 (83.00)

83.55% and 72.81% for multi-class and binary classes, respectively. For the binaryexample, the comparable F1 score is 86.89%. The LSTM models exhibit accuracylevels of 83% in multi-classification and over 98% in binary classification (F1 scoreof 99.75%). Because LSTM can capture the temporal dependency presented in a packet sequence that MLP cannot, the LSTM models perform better. Furthermore,our LSTM models perform better than the earlier studies.

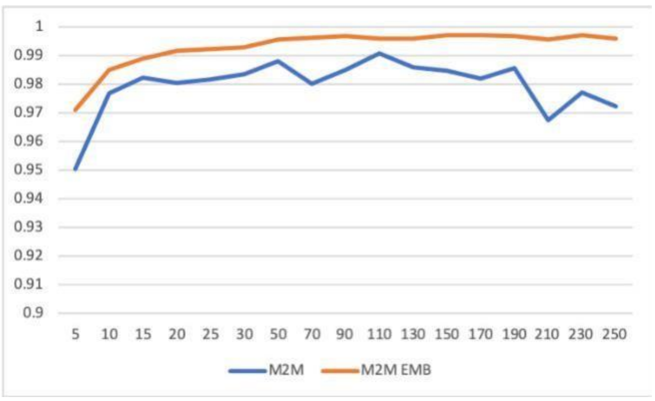


Fig. 7. Binary-classi cation accuracy graphs on the validation data: M2M, and M2M withembedding. The horizontal axis indicates the length of sequence.

The M2M+EMB model outperformed the other LSTM models in both binary and multiple classification tests. The reason for this is that feature embedding effectivelycaptures the information for neural networks, and categorical features contain distinct information. In fact, as demonstrated in Figs., EMB models perform better than similar non-EMB models (around 1% higher for binary classification and 2% higher for multi-classification) and produce more consistent results.

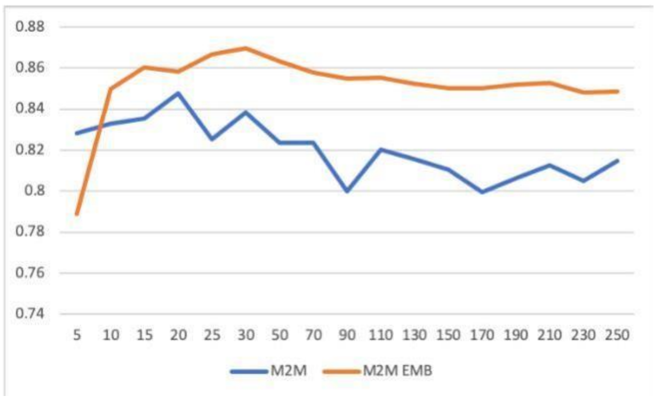


Fig. 8. Multi-classi cation accuracy graphs on the validation data: M2M, and M2M with embedding. The horizontal axis indicates the length of sequence.

Furthermore, M2B can be used for binary classification, although it has no discernible effect on performance. M2B and non-M2B model outcomes are nearly identical.

We examined the prediction time with various sequence lengths in the model for practical considerations, and the results are summarised in Fig. 9, which shows that the forecast time is linear with the sequence length.

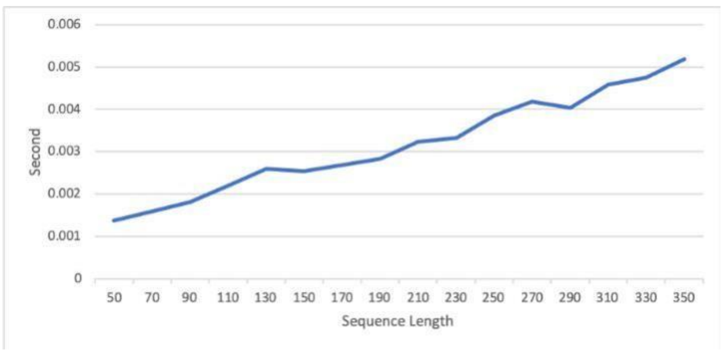


Fig. 9. Prediction time in seconds per sequence with various sequence lengths.

Conclusion

In conclusion, this project underscores the vital role of deep learning in addressing the evolving challenges of network attack detection in social networking security. As societal integration with the Internet deepens, security threats rise, necessitating innovative solutions. The Intrusion Detection System (IDS) stands as a notable achievement, capable of discerning ongoing or pastintrusions. The research focuses on classifying network traffic into five categories, differentiating normal behavior from various attack types for enhanced information security.

6 References

[1] Network Intrusion Detection Using Deep Neural Networks M.Ponkarthika1 and Dr.V.R.Saraswathy2 (Open Access Quarterly International Journal) Volume 2, Issue 2, Pages 665-673, April-June 2018

[2] Host Based Intrusion Detection System with Combined CNN/RNN
ModelAshima Chawla(B), Brian Lee, Sheila Fallon, and Paul Jacob
On the Effectiveness of Machine and Deep Learning for Cyber Security 2018 10th International Conference on Cyber Conflict

[3] Collective Anomaly Detection Based on Long Short-Term Memory Recurrent Neural Networks Lo¨ıc Bontemps, Van Loi Cao(B), James McDermott, and Nhien-An Le-Khac A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection Anna L. Buczak, Member, IEEE, and Erhan Guven, Member, IEEE

[4] Application of Neural Networks for Intrusion Detection in Tor Networks Taro Ishitaki∗, Donald Elmazi†, Yi Liu ∗, Tetsuya Oda ∗, Leonard Barolli‡ and Kazunori Uchida‡2015 29th International Conference on Advanced Information Networking and Applications Workshops