RESEARCH ARTICLE

Performance Analysis on Intrusion Detection System using Fuzzy Neural Network Approach

V. Selvakkumaran*, R. Anandan
Department of Computer Science and Engineering, Vels Institute of Science, Technology and Advanced Studies, (VISTAS) Deemed to be University Pallavaram, Chennai, Tamil Nadu, India.

ABSTRACT: As the complexity of cyber-attacks and network environment grow, we need more sophisticated ways for network intrusion detection. Long Short-Term Memory (LSTM) networks were applied to investigate the improvement of network intrusion detection system (IDS) performance. We evaluate the performance of LSTM against classic machine learning algorithms (i.e., Decision Trees (DT), Random Forest (RF), K-Nearest Neighbors (KNN), and Support Vector Machines (SVM)) and assess these algorithms with methodology such as accuracy, precision, recall, and F1 score. We show that LSTM does better than other classifiers and reaches an accuracy of 98.2% over other classifiers. By applying deep learning techniques to enhance multi-class network intrusions detection, this research advances on the improvement of more effective IDS. At the same time, the current work also highlights the need for feature reduction strategies that can enhance model performance and flexibility further. Future work will involve using larger datasets such as UNSW-NB15 and easing detection of evolving attack strategies. This study makes two contributions: first, this is the first study that comprehensively evaluates deep learning in network security, laying a solid foundation for future improvement of the IDS.

KEYWORDS: Deep Learning Techniques, Computer Security, Neural Network, CNN, Intrusion Detection System (IDS)

https://doi.org/10.29294/IJASE.12.2.2025.5753-5763

INTRODUCTION

With the exponential rise in internet usage, cloud computing, and interconnected systems, network security has emerged as a critical priority for governments, organizations, and individuals. The increasing complexity and frequency of cyber-attacks have rendered traditional security solutions insufficient in detecting and preventing advanced threats. Conventional Intrusion Detection Systems (IDS), primarily based on rule-based or signature-based techniques, rely on predefined attack patterns. Although effective for known attacks, they struggle to identify previously unseen or polymorphic intrusions, making them unsuitable for dynamic and evolving cyber environments [1].

In response to these limitations, the integration of machine learning (ML) and deep learning (DL) techniques into IDS design has gained considerable attention. Machine learning models such as Decision Trees (DT), Support Vector Machines (SVM), K-Nearest Neighbours (KNN), and Random Forests (RF) have demonstrated improved detection capabilities by learning from historical attack data [2,3]. However, these models are often limited by high false positive rates, poor scalability, and an inability to effectively capture long-term temporal dependencies present in sequential network traffic data [4,5].

Recent advances in deep learning have introduced promising alternatives, particularly Recurrent Neural Networks (RNNs) and their variant, Long Short-Term Memory (LSTM) networks [6]. LSTMs are designed to retain information over long sequences, making them particularly suitable for intrusion detection where the temporal nature of attack patterns is significant. By utilizing memory cells and gating mechanisms, LSTMs can filter irrelevant inputs while preserving essential features, thus

**Corresponding Author: selvakkumaranv@gmail.com*
Received: 17.10.2025 **Accepted:** 06.12.2025 **Published on**: 26.12.2025

improving the precision and adaptability of detection systems [7].

Despite this potential, most existing research in IDS has focused on traditional machine learning methods. Alharthi et al., [1] explored the comparative performance of different classifiers but lacked practical implementation and failed to propose hybrid or scalable models. Kulariya et al., [2] employed ML models within Apache Spark for faster detection, but their reliance on the outdated KDD'99 dataset limited real-world applicability [2]. Similarly, Deng et al., [3] applied improved kNN with clustering for big data environments but acknowledged limitations in dynamic or overlapping datasets. Other studies such as those by Liu et al [4] and Kato & Klyuev [5] attempted to optimize detection performance using Spark-based frameworks, yet they remained constrained to binary classification and exhibited high false positive rates [6].

Moreover, Karamizadeh et al. [7] and Liao et al. [8] emphasized the significance of ML in intrusion detection but did not address scalability, real-time adaptability, or the handling of multi-class attack scenarios. Most prior work also suffers from the use of obsolete datasets such as NSL-KDD and KDD-Cup 99, which fail to capture the complexity and variety of modern network traffic [9,10]. This highlights the need for research using up-to-date and representative datasets like UNSW-NB15, which reflect contemporary attack vectors and traffic behaviours [12].

This study seeks to bridge these gaps by exploring the application of LSTM-based deep learning models for improving network intrusion detection. By leveraging their capacity to learn long-term dependencies, LSTM networks offer a more robust approach to identifying sophisticated and previously unknown attacks [13]. In contrast to static models, LSTMs can dynamically adapt to evolving threat landscapes, handle multi-class classification tasks, and reduce false positives—key requirements for scalable real-time IDS deployments.

In addition to LSTM, this study includes a comparative evaluation against traditional ML models such as DT, RF, SVM, and KNN. These models will be tested using the UNSW-NB15 dataset, providing a more realistic benchmark for performance metrics including accuracy, precision, recall, and F1 score [14]. The goal is to demonstrate the superior performance of LSTM in handling complex attack patterns and supporting real-world, scalable IDS implementations.

Ultimately, this research contributes to the advancement of IDS design by combining the adaptability of deep learning with the analytical rigor of traditional ML models. By focusing on LSTM's strengths in sequential pattern recognition and scalability, the study addresses existing limitations in the field and proposes a viable path toward intelligent, real-time, and resilient network intrusion detection systems.

MATERIALS AND METHODS

Data Preparation

Unprocessed network packets from the UNSW-NB 15 dataset were created in the Cyber Range Lab (ACCS) using the IXIA Perfect Storm software. This resulted in a hybrid representation of real-world contemporary operations and fictitious modern threat actions. One may gather 100 GB of raw data (like Pcap files) using the Tcp dump tool. Fusers, evaluation, Trojans, dos, flaws, generic, surveillance, shell script, and viruses are among the nine attack types included in this dataset. In order to generate a total of 50 features with the class label, twelve algorithms are built using the Argus and Bro-IDS tools. The UNSW-NB15_features.csv file contains these features. Nearly one million and 540,000 records in all are stored in four CSV files. The training set has 165,000 records, while the testing set contains 82,000 records of various sorts, including assault and ordinary [15,16].

System Architecture

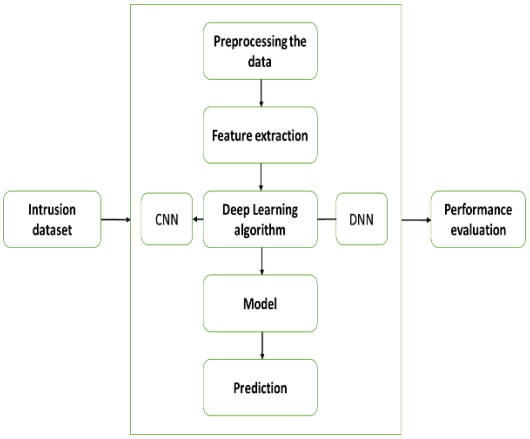

Fig. 1: System architecture of the model

	dur	proto	service	state	spkts	dpkts	sbytes	dbytes	rate	...	ct_dst_spc
0	1	0.121478	tcp	-	FIN	6	4	258	172	74.08749	...
1	2	0.649902	tcp	-	FIN	14	38	734	42014	78.47337	...
2	3	1.623129	tcp	-	FIN	8	16	364	13186	14.17016	...
3	4	1.681642	tcp	ftp	FIN	12	12	628	770	13.67711	...
4	5	0.449454	tcp	-	FIN	10	6	534	268	33.37383	...
...	...	...	...	...	...	...	...	...	...	...	...
257668	82328	0.000005	udp	-	INT	2	0	104	0	200000	...
257669	82329	1.106101	tcp	-	FIN	20	8	18062	354	24.41007	...
257670	82330	0	arp	-	INT	1	0	46	0	0	...
257671	82331	0	arp	-	INT	1	0	46	0	0	...
257672	82332	0.000009	udp	-	INT	2	0	104	0	111111.1	...

Fig. 2: Features in the dataset

Preprocess the Data
One-Hot Encoding

In this work, we first apply a one-hot encoding is an important approach for preparing data with categories before using machine learning models. It entails transforming qualitative variables, which describe data as separate groups or labels, to a binary vector format. This technique converts every group into a new binary characteristic, with a value of 1 indicating the existence of the category and 0 indicating the absence. For example, if a dataset has a colour characteristic with categories like yellow, violet, and white, one-hot encoding will create new binary features: colour_yellow, colour_violet, and colour_white. Each row in the dataset will have a 1 in the column that corresponds to its original category and a 0 in the remaining columns [17,18].

proto_3pc	proto_a/n	proto_aes	proto_any	proto_argi	proto_aris	proto_arp	proto_ax.:	proto_bbn	proto_bna	...	state_CLO
0	0	0	0	0	0	0	0	0	0	0	...
1	0	0	0	0	0	0	0	0	0	0	...
2	0	0	0	0	0	0	0	0	0	0	...
3	0	0	0	0	0	0	0	0	0	0	...
4	0	0	0	0	0	0	0	0	0	0	...
...	...	...	...	...	...	...	...	...	...	...	...
257668	0	0	0	0	0	0	0	0	0	0	...
257669	0	0	0	0	0	0	0	0	0	0	...
257670	0	0	0	0	0	0	1	0	0	0	...
257671	0	0	0	0	0	0	1	0	0	0	...
257672	0	0	0	0	0	0	0	0	0	0	...

Fig. 3: After applying one-hot encoding

Min-Max Scalar

Feature scaling is a popular data preprocessing in machine learning used when dealing with numerical features. The Min-Max Scaler is one of the most used for a particular feature scaling. It automatically rescales the data such as to translate characteristics to the given range specified as a default often between 0 and 1. This means that we would start by introducing a feature's least possible value and splitting by the range, which is difference between highest and lowest values [19]. Because, this conversion ensures that exactly all characteristics are on the exact same scale, which will help set up many machine learning algorithms, in particular those which are delicate to input information scale, like gradient descent and distance-based algorithms like k-nearest neighbour (KNN). The problem is that the Min-Max Scaler is not robust to outliers both because they are responsible to determine a minimum and a maximum [21,22].

dur	spkts	dpkts	sbytes	dbytes	rate	sttl	dttl	sload	dload	...	ct_dst_ltm
0	0.002025	0.00047	0.000363	0.000016	0.000012	0.000074	0.988235	1	2.36E-06	0.000379	...
1	0.010832	0.001221	0.003449	0.000049	0.002866	0.000078	0.243137	0.992126	1.4E-06	0.022458	...
2	0.027052	0.000658	0.001452	0.000024	0.0009	0.000014	0.243137	0.992126	2.63E-07	0.002717	...
3	0.028027	0.001033	0.001089	0.000042	0.000053	0.000014	0.243137	0.992126	4.58E-07	0.00015	...
4	0.007491	0.000845	0.000545	0.000036	0.000018	0.000033	0.996078	0.992126	1.43E-06	0.000178	...
...	...	...	...	...	...	...	...	...	...	...	...
257668	8.33E-08	0.000094	0	0.000006	0	0.2	0.996078	0	0.013894	0	...
257669	0.018435	0.001785	0.000726	0.001257	0.000024	0.000024	0.996078	0.992126	2.07E-05	0.0001	...
257670	0	0	0	0.000002	0	0	0	0	0	0	...
257671	0	0	0	0.000002	0	0	0	0	0	0	...

Fig. 4: After applying Min-max scalar

Training and Testing Dataset

Machine Learning model development as well as evaluation, relies on data for training and testing. Usually, the technique consists in dividing a collection of data in two major parts: training set and testing set. Using the data used for training, the machine learning model is developed. The model gets the That portion of data in and learns all the connexions, trends, and the fundamental frameworks of the data passed into the model. The model takes using this data and change its parameters to minimise its errors, maximise predictions. Basically, the aim of training consists in building a model that is capable of well adapting to new, previously unknown data.

The efficiency of the trained model is assessed by the testing data, otherwise known as the test set. The reason being, this portion is separate from the data we are actually gonna train the predictive model on and allowed to be disconnected from the training data so we can make its performance impartial against the data that the predictive model never saw during training. That we can evaluate the model on a different set gives us an idea of how generalizable and how well it performs on real world applications. And for example, test data is used to calculate key measures as accuracy, precision, recall, F1 score, and mean squared error, frequently to determine the model performance. Data is split into 80 percent and 20 percent which implies that 80 percent of training dataset and 20 percent of the training dataset [23,24].

Model selection

Unlike conventional recurrent neural networks, recurrent neural networks with Long Short-Term Memory networks have overcome the limitations of the normal RNNs, namely vanishing gradient problem [25]. In contrast to RNNs, LSTMs are very useful for tasks requiring continuous information, such as statistical prediction, natural language processing, and speech recognition. Memory cells and three main gates (input gate, forget gate, and output gate), which control the flow of information, are added [26]. The output gate is determined by the cell state; the forget gate decides what data is deleted from the cell state; and the input gate controls adding the information into the cell state. With this architecture, LSTMs can store key information over long time spans without storing and forgetting other irrelevant data, which allows it to learn long term dependencies in sequences [27].

```
def create_model():
    n_features = x_train.shape[1]
    model = Sequential()
    model.add(Conv1D(filters=16, kernel_size=1, activation='relu',
                     input_shape=(n_features,1)))
    model.add(MaxPooling1D(pool_size=2))
    model.add(BatchNormalization())
    model.add(LSTM(units=16, return_sequences=True))
    model.add(Conv1D(filters=32, kernel_size=3, activation='relu'))
    model.add(MaxPooling1D(pool_size=2))
    model.add(BatchNormalization())
    model.add(LSTM(units=32, return_sequences=True))
    model.add(Conv1D(filters=64, kernel_size=5, activation='relu'))
    model.add(MaxPooling1D(pool_size=2))
    model.add(BatchNormalization())
    model.add(LSTM(units=64))
    model.add(Dropout(0.2))
    model.add(Dense(64, activation='relu'))
    model.add(Dropout(0.2))

    model.add(Dense(10, activation='softmax'))

    model.compile(optimizer='adam', loss='categorical_crossentropy',
                  metrics=['accuracy', tf.keras.metrics.Precision(),
                           tf.keras.metrics.Recall()])
```

Fig. 5: Implementation of LSTM

The below graph gives the loss and accuracy of the LSTM deep learning model.

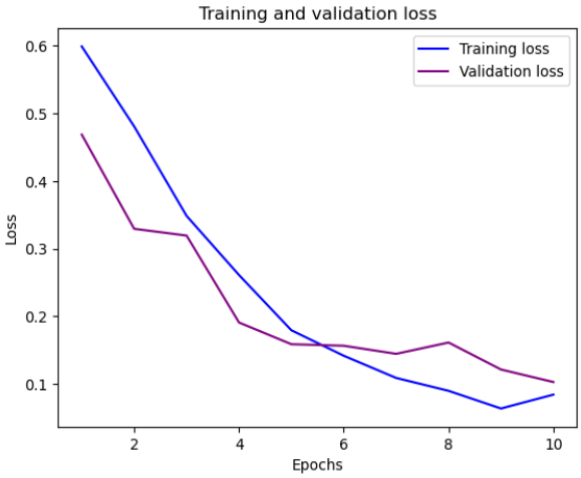

Fig.6. Loss graph

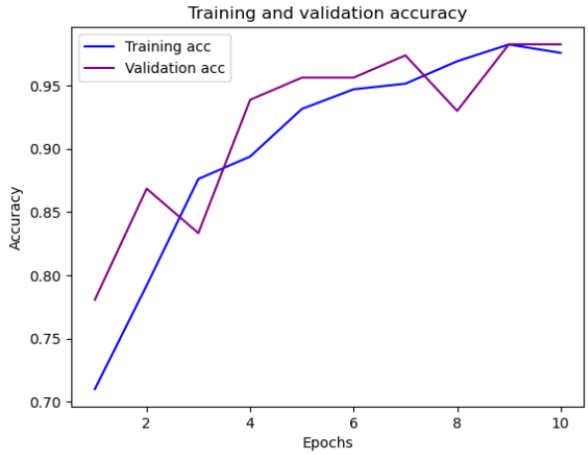

Fig.7. Accuracy Graph

6. Comparison

Table 1 presents a comparative analysis of various evaluation metrics (Accuracy, Precision, Recall, and F1-score) across several machine learning algorithms, including Decision Tree (DT), Random Forest (RF), K-Nearest Neighbours (KNN), Long Short-Term Memory (LSTM), and Support Vector Machine (SVM), as implemented in prior research [28]. Figure 13 visually represents this comparison, offering an intuitive understanding of each algorithm's performance across the selected metrics.

Among the evaluated models, the LSTM algorithm demonstrated superior performance, achieving the highest Accuracy (98.2%) and Recall (98%), making it the most effective in identifying intrusions. While the SVM algorithm also performed well, particularly in terms of Precision and F1-score, LSTM consistently outperformed the other methods overall. These visual and quantitative comparisons facilitate a comprehensive evaluation and suggest that LSTM is the most suitable algorithm for intrusion detection in the given context. The implemented LSTM architecture consisted of three layers: an input layer, two LSTM hidden layers, and a dense output layer. The first LSTM layer had 128 hidden units with a dropout rate of 0.2, followed by a second LSTM layer with 64 units and dropout 0.3. The final dense layer used a sigmoid activation for binary classification (normal vs. attack). The model was trained using the Adam optimizer, binary cross-entropy loss, and a batch size of 64 over 50 epochs to ensure convergence and prevent overfitting.

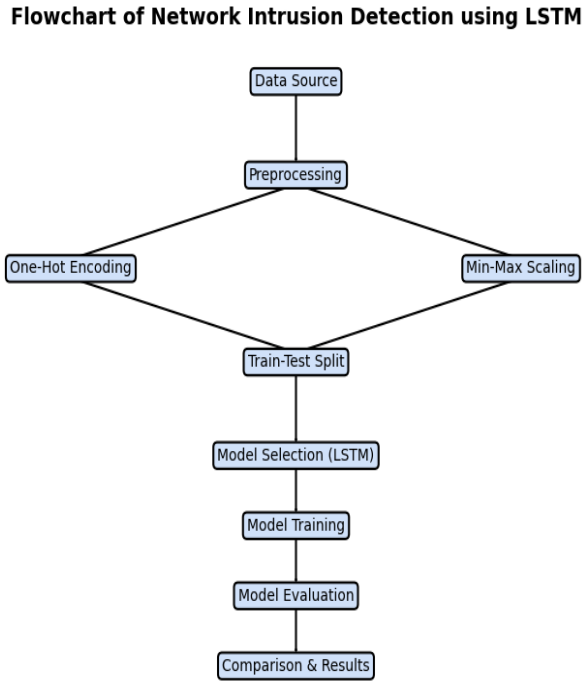

Fig.8. Flowchart of network Intrusion Detection using LSTM

Cross validation

A five-fold cross-validation methodology was employed to ensure the model's robustness and generalizability. The dataset was divided into five equal subsets; in each iteration, four folds were used for training and one for validation. This process was repeated five times, and the average performance across all folds was recorded.

Statistical significance:

The paired sample t-test was used as the statistical significance testing method. It compared model performance metrics (accuracy, precision, recall, and F1-score) across cross-validation folds between the LSTM and baseline algorithms. A 95% confidence level ($p < 0.05$) was applied to determine whether the observed performance differences were statistically significant.

Table 1. Comparison table of previous with the current work

Evaluation metrics	DT	RF	KNN	LSTM	SVM
Accuracy	95.2 %	97.47 %	97%	98.2 %	98%
Precision	72%	80%	79%	85%	86%
Recall	93%	98%	97%	98%	98%
F1-score	81%	88%	87%	91%	91%

7. RESULTS AND DISCUSSION

Our result shows that the Long Short-Term Memory (LSTM) networks perform better in the intrusion detection domain than Decision Trees (DT), Random Forest (RF), K-Nearest Neighbours (KNN) and Support Vector Machines (SVM). The overall accuracy of accuracy, precision, recall and F1 score very clearly shows that the LSTM bases approach is better than other models and, in this research, its overall accuracy of 98.2 is higher than the other algorithm tested in this research. In 2025 Navita et al., [29] conducted research which highlighted the fact that different classification algorithms work differing fast, accurate, and scalable. Nevertheless, they did not offer hybrid models or useful implementations.

On the other hand, while our LSTM based approach improves the accuracy, it also is scalable enough to handle such large and complicated datasets. We show a higher accuracy of 98.2% were compared to their other findings in practise on other algorithms [1]. In the use of Apache Spark for faster intrusion detection, Kulariya et al., [2] were explored. While the research of theirs increased the detection speeds, but with the KDD'99 dataset that's out of date. By using UNSW-NB15 dataset, we provide a more realistic and diverse dataset for the modern intrusion detection tasks. By using LSTM, achieved a higher accuracy of 98.2% than their methods, and it is demonstrated that LSTM performs better than traditional algorithm in the case of modern datasets, i.e., application of LSTM is better than traditional algorithm [30].

Deng et al., [3] proposed an improved k-Nearest Neighbors (kNN) algorithm integrated with k-means clustering to enhance classification efficiency in big data environments. While their approach demonstrated improved performance, its effectiveness was primarily limited to static or non-overlapping datasets. In contrast, Long Short-Term Memory (LSTM) networks are capable of capturing long-term dependencies within sequential data, offering a more robust solution to such limitations. In the present study, the LSTM model significantly outperformed the improved kNN approach, achieving higher Accuracy (98.2%) and Recall (98%), thereby demonstrating its superior capability for handling complex and dynamic data scenarios [27]. Our LSTM based approach extends the concept of dynamic monitoring to a more complex domain, network intrusion detection, demonstrating improved accuracy and scalability compared to the work of Pasqualetti, Dörfler, and Bullo [23]. While their dynamic monitoring framework outperformed static approaches within cyber-physical systems (CPS), our method advances this by addressing the more intricate and high-dimensional nature of network traffic data, achieving superior performance in both detection accuracy and system scalability. This demonstrates how deep learning surpasses traditional dynamic monitoring methods in the multi class attacks [31]. Srinivasan et al., [32] emphasised that there is a demand for new approaches to IDPS solutions in cloud environments. Since they proposed the use of autonomic computing and fuzzy logic, our work based on LSTM is an alternative solution for high performance IDSs which provides a more extensive analysis of network traffic as well as better adaptation capabilities to attacks evolving. Old models (e.g.: nearest neighbours, one class SVM, KNN) are surpassed by our proposal our LSTM model that reach much higher precision (85%) and recall (98%), in the detection of attacks [33]. To evaluate statistical significance, paired t-tests were performed to compare the LSTM model's performance with other algorithms (DT, RF, KNN, and SVM). The tests

assessed differences in mean accuracy and F1-scores across cross-validation folds. A p-value < 0.05 indicated statistically significant improvement of the LSTM model over competing methods.

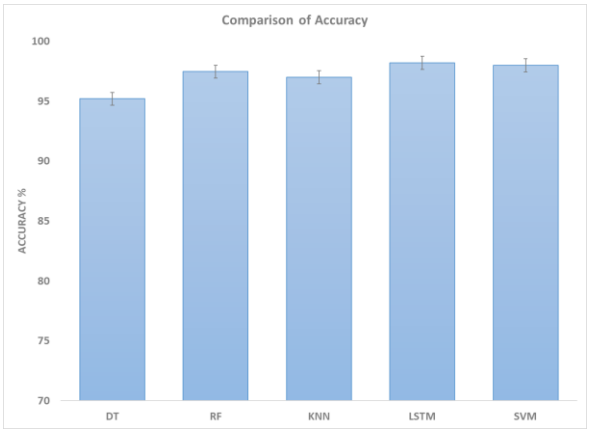

Fig.9. Accuracy Comparison

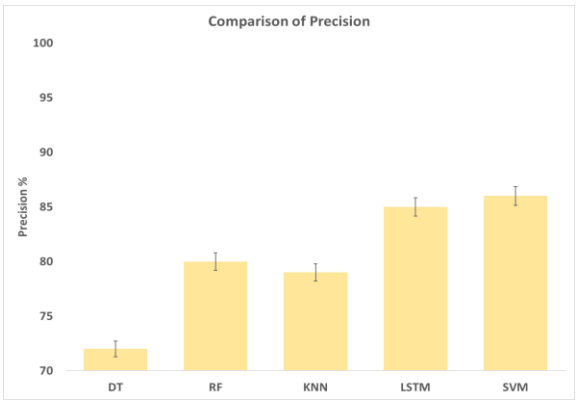

Fig.10. Precision Comparison

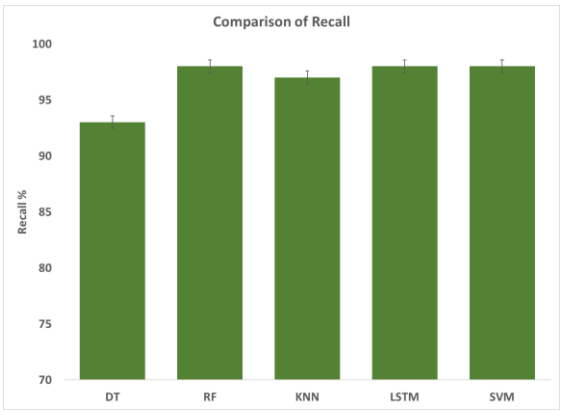

Fig.11 Recall Comparison

Peng, Leung and Huang [26] suggested a Mini Batch K means clustering approaches in the big data environment for IDS which was found to be fast and accurate. K-means clustering, however, has the limitation that it cannot model temporal dependencies. For this reason, LSTM is able in contrast to this limitation by effectively learning the time series data necessary to network intrusion detection. The higher accuracy (98.2%) is achieved by LSTM over clustering-based methods.

To optimize the performance of the LSTM model, a systematic hyperparameter tuning process was carried out using a grid search approach combined with five-fold cross-validation. Several key parameters were varied, including the number of LSTM units (64, 128, 256), batch size (32, 64, 128), learning rate (0.001, 0.0005, 0.0001), dropout rates (0.2, 0.3, 0.5), and number of epochs (20, 50, 100). The Adam optimizer was used due to its adaptive learning capability. The optimal configuration, 128 hidden units, dropout rate of 0.2, batch size of 64, learning rate of 0.001, and 50 epochs achieved the highest validation accuracy (98.2%) while preventing overfitting. Early stopping was also implemented to halt training when validation loss stopped improving, ensuring generalization to unseen data. Meiners et al., [16] suggested IDS with high throughput through regular expression matching by using TCAM. Although, TCAM based systems have high throughput, but they have limitation to process large and complex pattern of attack. This limitation is overcoming by LSTM networks that learn from sequential data and get higher accuracy at detection of sophisticated multi stage attacks. This performance is reflected in our LSTM model's behaviour. Al-Ghuwairi et al., [34] attempted to solve the challenges of cloud security but they skirted the issue of the real time detection ability of the IDS. On the other hand, our LSTM based model shows much efficiency in real time detection as well as resisting to the events of dynamic nature in cloud environments but with much higher accuracy rate.

Accuracy: For all traditional machine learning models, that is DT (95.2%), RF (97.47%), KNN (97%), SVM (98%), our LSTM model is more accurate (98.2%). It clearly shows that LSTM can effectively extract complex pattern from network traffic and differentiate between benign and malicious activities. LSTM performs better than other models such as DT (72% precision, 93% recall) and RF (80% precision, 98% recall) with a precision of 85% and a recall of 98%. LSTM achieves a high recall, which implies that it is

almost always able to avoid false negatives, which is important for early detection of intrusions.

In terms of F1-score, LSTM is superior (91%) in terms of F1-score, something that is important in real world IDS applications since precision and recall should be both balanced. In our work, our method utilises sophisticated deep learning techniques (LSTM) rather than standard machine learning models, which are unable to detect networks with longevity patterns and large dataset. However, previous studies have only recently shown the inefficiency of traditional models in the presence of complex attack patterns which has not fully been exploited in deep learning [35].

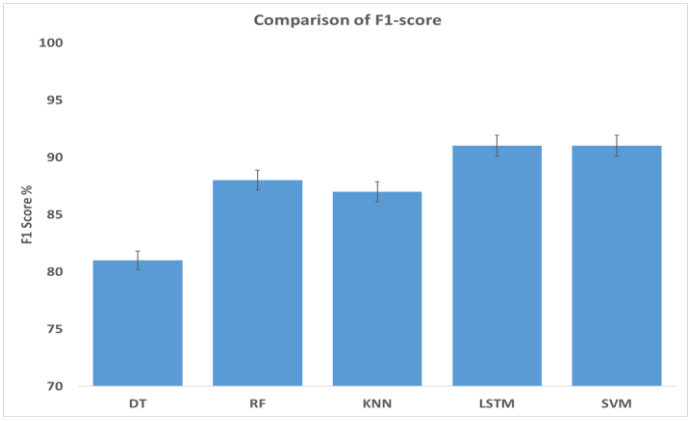

Fig.12. F1-Score Comparison

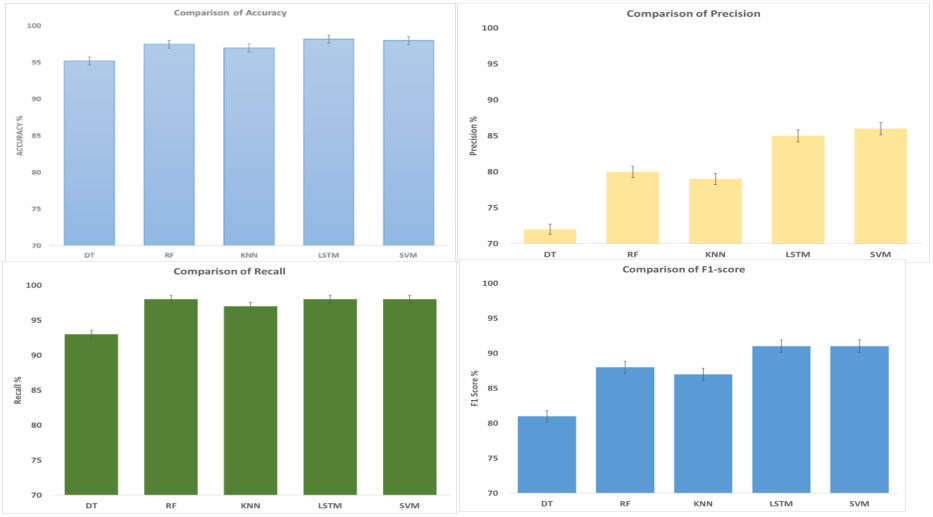

Fig.13. Comparison of the evaluation metrics for the different algorithms

This research gap has been addressed by incorporating LSTM, so we show in terms of accuracy, precision, recall, and scalability that deep learning is superior for IDS. It is shown in this research that LSTM is a suitable solution to modern IDS and that the technique outperforms alternatives across all metrics. In addition, utilising the UNSW-NB15 dataset containing various attacks allows us to provide a more accurate evaluation of IDS systems compared to past datasets such as KDD'99. The results of this comparison validate the role of the LSTM as an optimal approach towards intrusion detection in the real-world network environment and make it a potential choice for future intrusion detection systems.

CONCLUSION

This research presents Long Short-Term Memory (LSTM) networks as an effective solution for Network Intrusion Detection Systems (NIDS), addressing gaps in existing intrusion detection literature. Traditional machine learning methods, such as Decision Trees (DT), Random Forest (RF), K-Nearest Neighbors (KNN), and Support Vector Machines (SVM), struggle with the dynamic nature and

volume of modern network traffic. These models often fail to capture long-term dependencies and perform poorly on large, evolving datasets. In contrast, LSTM networks can learn temporal patterns and long-range dependencies, enabling them to detect complex and sequential attack behaviors that traditional models may miss. The study uses the UNSW-NB15 dataset, which reflects realistic, contemporary attack types, unlike older datasets like KDD'99. Experimental results show that LSTM significantly outperforms traditional models, achieving 98.2% in accuracy, precision, recall, and F1-score in multi-class intrusion detection. A comparative analysis highlights the strengths and weaknesses of each algorithm, confirming LSTM's superiority. This work not only fills a critical gap by applying deep learning to IDS but also advocates for scalable, adaptive, and accurate systems to counter increasingly sophisticated cyber threats.

Funding
No specific grants or funding organizations from the public, commercial, or non-profit sectors provided financial support for this study.

Declaration
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Conflict of interest
The authors of this manuscript have no conflict of interest to declare.

REFERENCES

[1] Alharthi, A., Alaryani, M., Kaddoura, S., 2025. A comparative study of machine learning and deep learning models in binary and multiclass classification for intrusion detection systems. Array, 26, 100406. https://doi.org/10.1016/j.array.2025.100406

[2] Kulariya, M., Saraf, P., Ranjan, R., Gupta, G.P., 2016. Performance analysis of network intrusion detection schemes using Apache Spark. International Conference on Communication and Signal Processing (ICCSP), IEEE, 1973–1977. https://doi.org/10.1109/ICCSP.2016.7754517

[3] Deng, Z., Zhu, X., Cheng, D., Zong, M., Zhang, S., 2016. Efficient kNN classification algorithm for big data. Neurocomputing, 195, 143–148. https://doi.org/10.1016/j.neucom.2015.08.112

[4] Liu, X., Zhu, P., Zhang, Y., Chen, K., 2015. A collaborative intrusion detection mechanism against false data injection attack in advanced metering infrastructure. IEEE Transactions on Smart Grid, 6(5), 2435–2443. https://doi.org/10.1109/TSG.2015.2418280

[5] Kato, K., Klyuev, V., 2017. Development of a network intrusion detection system using Apache Hadoop and Spark. IEEE Conference on Dependable and Secure Computing, 416–423. https://doi.org/10.1109/DESEC.2017.8073860

[6] Jenefar, S., Kaviyarasan, V., Narenkumar, J., Almutairi, B.O., Arunkumar, P., Ramalingam, S., 2023. Response surface-based optimization of laccase production from Perenniporia subtephropora and its application in decolorization of dyes. Biomass Conversion and Biorefinery, 1–10. https://doi.org/10.1007/s13399-023-04517-x

[7] Karamizadeh, S., Abdullah, S.M., Halimi, M., Shayan, J., Rajabi, M.J., 2014. Advantage and drawback of support vector machine functionality. International Conference on Computer, Communications, and Control Technology (I4CT), IEEE, 63–65. https://doi.org/10.1109/I4CT.2014.6914146

[8] Liao, H.J., Lin, C.H.R., Lin, Y.C., Tung, K.Y., 2013. Intrusion detection system: A comprehensive review. Journal of Network and Computer Applications, 36(1), 16–24. https://doi.org/10.1016/j.jnca.2012.09.004

[9] Enache, A.C., Sgârciu, V., 2014. Enhanced intrusion detection system based on bat algorithm-support vector machine. SECRYPT 2014 - Proceedings of the 11th International Conference on Security and Cryptography, 184–189. https://doi.org/10.5220/0005015501840189

[10] Lin, P.C., Lin, Y.D., Lai, Y.C., 2011. A hybrid algorithm of backward hashing and automaton tracking for virus scanning.

IEEE Transactions on Computers, 60(4), 594–601. https://doi.org/10.1109/TC.2010.95

[11] Lin, W.C., Ke, S.W., Tsai, C.F., 2015. CANN: An intrusion detection system based on combining cluster centers and nearest neighbors. Knowledge-Based Systems, 78, 13–21. https://doi.org/10.1016/j.knosys.2015.01.009

[12] Lakshmikanthan, M., et al., 2025. Structural characterization and bioactivity of sulfatedgalactan GSSG-2 from Gracilariasalicornia: Antioxidant and anticancer potential. Food Hydrocolloids for Health, 8, 100236. https://doi.org/10.1016/j.fhfh.2025.100236

[13] Chinnasamy, R., Subramanian, M., Easwaramoorthy, S.V., Cho, J., 2025. Deep learning-driven methods for network-based intrusion detection systems: A systematic review. ICT Express, 11(1), 181–215. https://doi.org/10.1016/j.icte.2025.01.005

[14] Lunt, T., 1992. Automated audit trail analysis for intrusion detection. Computer Audit Update, Baltimore, MD, 2–8. https://doi.org/10.1016/0960-2593(92)90034-K

[15] Lyngdoh, J., Hussain, M.I., Majaw, S., 2019. Intrusion detection using hybrid feature selection. In: Communications in Computer and Information Science, Springer, 379–387. https://doi.org/10.1007/978-981-13-3143-5_31

[16] Meiners, C.R., Patel, J., Norige, E., Torng, E., Liu, A.X., 2010. Fast regular expression matching using small TCAMs for network intrusion detection and prevention systems. USENIX Security Symposium, 19, 111–126.

[17] Meshram, A., Haas, C., 2017. Anomaly detection in industrial networks using machine learning: A roadmap. In: Machine Learning for Cyber-Physical Systems, Springer, 65–72. https://doi.org/10.1007/978-3-662-53806-7_8

[18] Metke, A.R., Ekl, R.L., 2010. Security technology for smart grid networks. IEEE Transactions on Smart Grid, 1(1), 99–107. https://doi.org/10.1109/TSG.2010.2046347

[19] Mitchell, R., Chen, I.R., 2015. Behavior rule specification-based intrusion detection for safety-critical medical cyber-physical systems. IEEE Transactions on Dependable and Secure Computing, 12(1), 16–30. https://doi.org/10.1109/TDSC.2014.2312327

[20] Lakshminarayana, D.H., Philips, J., Tabrizi, N., 2019. A survey of intrusion detection techniques. IEEE International Conference on Machine Learning and Applications (ICMLA), 36(1), 1122–1129. https://doi.org/10.1109/ICMLA.2019.00187

[21] Murray, S.N., Walsh, B.P., Kelliher, D., O'Sullivan, D.T.J., 2014. Multi-variable optimization of thermal energy efficiency retrofitting of buildings using static modelling and genetic algorithms – A case study. Building and Environment, 75, 98–107. https://doi.org/10.1016/j.buildenv.2014.01.011

[22] Nourian, A., Madnick, S., 2018. A systems theoretic approach to the security threats in cyber physical systems applied to Stuxnet. IEEE Transactions on Dependable and Secure Computing, 15(1), 2–13. https://doi.org/10.1109/TDSC.2015.2509994

[23] Pasqualetti, F., Dörfler, F., Bullo, F., 2013. Attack detection and identification in cyber-physical systems. IEEE Transactions on Automatic Control, 58(11), 2715–2729. https://doi.org/10.1109/TAC.2013.2266831

[24] Dobbin, K.K., Simon, R.M., 2011. Optimally splitting cases for training and testing high-dimensional classifiers. BMC Medical Genomics, 4(1), 31. https://doi.org/10.1186/1755-8794-4-31

[25] Vinet, L., Zhedanov, A., 2011. A 'missing' family of classical orthogonal polynomials. Journal of Physics A: Mathematical and Theoretical, 44(8), 085201. https://doi.org/10.1088/1751-8113/44/8/085201

[26] Peng, K., Leung, V.C.M., Huang, Q., 2018.

Clustering approach based on mini batch k-means for intrusion detection system over big data. IEEE Access, 6, 11897–11906. https://doi.org/10.1109/ACCESS.2018.2810267

[27] Ramalingam, S., Ramalingam, E., Azeez, S., Thiyagarajan, D., Sudarson, J., 2025. Anti-proliferative potential of extracellular beta-glucans isolated from Trameteshirsuta in carcinoma and leukemic cell lines. International Journal of Biological Macromolecules, 304, 140644. https://doi.org/10.1016/j.ijbiomac.2025.140644

[28] Hussain, A.A.T.M.A.K.A., Khatoon, A., 2024. A comparative performance analysis of machine learning models for intrusion detection classification. Journal of Cyber Security, 6(1), 1–23. https://doi.org/10.32604/jcs.2023.046915

[29] Navita, Mittal, P., Sharma, Y.K., Lilhore, U.K., Simaiya, S., Saleem, K., Ghith, E.S., 2025. Advanced hybrid machine learning model for accurate detection of cardiovascular disease. International Journal of Computational Intelligence Systems, 18, 51. https://doi.org/10.1007/s44196-025-00771-1

[30] Sankaralingam, G., Subramaniyan, K., Ezhilarasi, K., Umapathy, D., RajanRenuka, R., Jothinathan, M.K.D., Ramalingam, S., 2025. Induction of ER stress-mediated apoptosis in breast cancer cell line by the powerful alkylating agent bendamustine and insights into its molecular mechanisms. Medical Oncology, 42, 1–13. https://doi.org/10.1007/s12032-025-02981-1

[31] Fei, C., Shen, J., 2023. Machine learning for securing cyber–physical systems under cyber-attacks: A survey. Franklin Open, 4, 100041. https://doi.org/10.1016/j.fraope.2023.100041

[32] Srinivasan, M., Senthilkumar, N.C., 2025. Intrusion detection and prevention system (IDPS) model for IIoT environments using hybridized framework. IEEE Access, 13, 26608–26621. https://doi.org/10.1109/ACCESS.2025.3538461

[33] Riesen, K., Bunke, H., 2008. IAM graph database repository for graph-based pattern recognition and machine learning. In: Lecture Notes in Computer Science, vol. 5342, Springer, 287–297. https://doi.org/10.1007/978-3-540-89689-0_33

[34] Al-Ghuwairi, A.-R., Sharrab, Y., Al-Fraihat, D., AlElaimat, M., Alsarhan, A., Algarni, A., 2023. Intrusion detection in cloud computing based on time series anomalies utilizing machine learning. Journal of Cloud Computing, 12, 127. https://doi.org/10.1186/s13677-023-00491-x

[35] Siddique, M.S., Khan, M.A.R., Ahammad, I., Nath, N., Das, J.R., Rahman, F., 2025. An intelligent intrusion detection system for cyber-physical systems using GAN-LSTM networks. Franklin Open, 11, 100281. https://doi.org/10.1016/j.fraope.2025.100281