Bot-Net Access Detection System Based on Deep Learning Algorithm in Hybrid-Cloud Infrastructure

Bari Hade Variant Wahono
Department of Informatics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
6025231016@student.its.ac.id

Ratih Nur Esti Anggraini
Department of Informatics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
ratih_nea@if.its.ac.id

Riyanarto Sarno
Department of Informatics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
riyanarto@if.its.ac.id

Agus Tri Haryono
Department of Informatics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
7025231020@student.its.ac.id

Abdullah Faqih Septiyanto
Department of Informatics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
7025221022@student.its.ac.id

2024 IEEE International Conference on Artificial Intelligence and Mechatronics Systems (AIMS) | 979-8-3503-5052-4/24/$31.00 ©2024 IEEE | DOI: 10.1109/AIMS61812.2024.10512749

***Abstract*—The escalating adoption of hybrid-cloud infrastructure technology across various fields results in a surge in resource utilization, encompassing server instances, databases, microservices, and other hybrid-cloud-based resources, thereby generating diverse log files. These logs serve various purposes, including classifying information based on urgency levels and facilitating data analysis for model training. Moreover, they play a pivotal role in detecting anomalies, such as DDoS attacks, bot-net access, and malware. This study aims to elucidate the process of gathering log files from various sources and processing them to detect bot-net access (formerly known as brute force attacks) across different protocols like SSH, FTP, and Kerberos. One-Hot Encoding is employed for feature extraction to identify bot-net access and distinguish it from other types of access logs. Subsequently, Deep Learning algorithms, including Simple Neural Network (SNN), Deep Neural Network (DNN), Artificial Neural Network (ANN), Convolutional Neural Network (CNN), and Long Short-Term Memory (LSTM) are utilized for detection. Our findings reveal that the Convolutional Neural Network (CNN) emerges as the top performer, achieving the highest average accuracy of 86.681%. These results advance anomaly detection capabilities in hybrid-cloud environments, thereby enhancing overall security measures.**

Keywords*—*Simple Neural Network, Deep Neural Network, Artificial Neural Network, Convolutional Neural Network, Long Short-Term Memory

I. Introduction

The advent of hybrid-cloud infrastructure has brought forth new challenges and complexities in ensuring the security and stability of computer systems and networks. As organizations embrace distributed architectures, the need to safeguard against evolving threats, such as bot-net infiltrations, becomes paramount in research [1]. In the operational landscape of computer systems, substantial data is generated and stored in log files, providing a critical resource for detecting and mitigating malicious activities. Recognizing the significance of this data, our focus shifts towards the development of a Bot-Net Access Detection System, harnessing the power of deep learning algorithms within the dynamic context of hybrid-cloud environments.

Navigating through the vast and intricate landscape of log entry information presents a formidable challenge, especially for non-experts. The identification and characterization of bot-net activities within log data demand sophisticated approaches that transcend traditional methods using machine learning algorithms based on research [2]. To address this, we propose an innovative solution, a classification system powered by a deep learning algorithm. This system deploys client applications across monitored instances or services, transmitting log data to a centralized server. Through the application of deep learning, the server processes and categorizes data, enabling swift identification and analysis of bot-net access patterns. This groundbreaking approach aims to fortify security measures within hybrid-cloud infrastructures and enhance the overall resilience of computer systems against evolving cyber threats.

In developing a robust Bot-Net Access Detection System, we prioritize user empowerment through streamlined retrieval of relevant information on potential bot-net activities. Utilizing deep learning algorithms, our system enables systematic monitoring and expedites root cause analysis, contributing to a proactive defense against dynamic bot-net infiltrations in hybrid-cloud environments.

This research [1] investigates diverse defense strategies against DDoS attacks across different web platforms, addressing unique challenges for ensuring uninterrupted web service availability. This study [2] explores web mining techniques using server log data, employing various classification algorithms and integrating genetic algorithms to enhance classifier performance, ultimately achieving superior predictive accuracy with the RFGA hybrid approach.

In study [3], researchers explore using Convolutional Neural Networks (CNNs) to detect brute force attacks in Brain-Computer Interfaces (BCIs), adapting CNNs from their typical use in image recognition to the nuanced domain of BCI

security. In [4], Md Delwar et al. compares Long Short-Term Memory (LSTM) networks with traditional machine learning algorithms for identifying SSH and FTP brute force attacks, delving into LSTM's strengths and weaknesses in handling sequential data. [5] compares neural network architectures MLP, CNN, LSTM, and SAE for detecting SSH brute force and DDoS attacks, adding complexity with an Intrusion Detection System (IDS) over Software-Defined Networking (SDN) networks.

In the research [6], the focus is on Network Intrusion Detection Systems (NIDS) in Software-Defined Networking environments, utilizing a combination of CNN and Bidirectional Long Short-Term Memory (BiLSTM) to enhance detection accuracy. [7] employs Area Under the Curve (AUC) and Area Under the Precision-Recall Curve (AUPRC) algorithms for detecting SSH and FTP brute force attacks, emphasizing robust performance assessment.

The research [8] introduces the Tree-CNN algorithm for building an IDS based on a Hierarchical Deep Convolutional Neural Network, allowing nuanced feature extraction at different levels. [9] employs CNNs for creating an IDS tailored for Network Traffic Payload Analysis, emphasizing the inspection of network packet content. [10] explores K-Nearest Neighbors (KNN) for detecting DDoS attacks on Resilient Distributed Datasets (RDD), depicted accuracy equation (2) and loss equation (1) as follows :

$$\text{Loss}(y, \hat{y}) = -\sum_{j=0}^{M}\sum_{i=0}^{N} y_{ij} \cdot \log(\hat{y}_{cij}) \tag{1}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

Study [11] introduces Multi-Layer Ensemble Deep Reinforcement Learning for DDoS detection in a Cloud-Software Defined Networking (Cloud-SDN) environment, showcasing a proactive and adaptive defense mechanism. [12] investigates the effectiveness of a combined CNN and LSTM approach for DDoS detection in Software-Defined Networking (SDN), harnessing spatial and temporal dependencies. [13] combines CNNs and Deep Neural Networks (DNN) for DDoS detection in SDN, incorporating Information Entropy and Deep Learning techniques for enhanced detection. [14] explores One-Hot Encoding in the context of Transfer Learning for Breast Cancer Classification, enhancing adaptability for improved accuracy. [15] addresses imbalanced datasets using SMOTE and analyzes data complexity, aiming to mitigate class imbalance issues and provide insights into dataset characteristics for robust model training.

II. Methodology

The research methodology workflow illustrated in (Fig. 1) systematically develops an efficient detection system, starting with a comprehensive literature review on bot-net threats, deep learning algorithms, and hybrid-cloud infrastructures. The proposed detection system architecture integrates deep learning algorithms in a hybrid-cloud environment. Practical

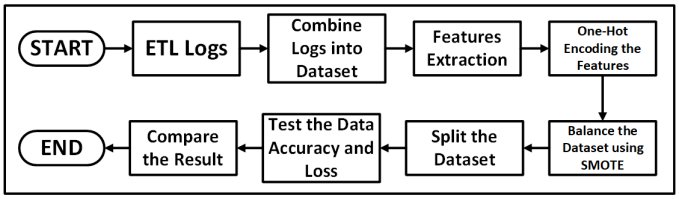

Fig. 1. Research Workflow

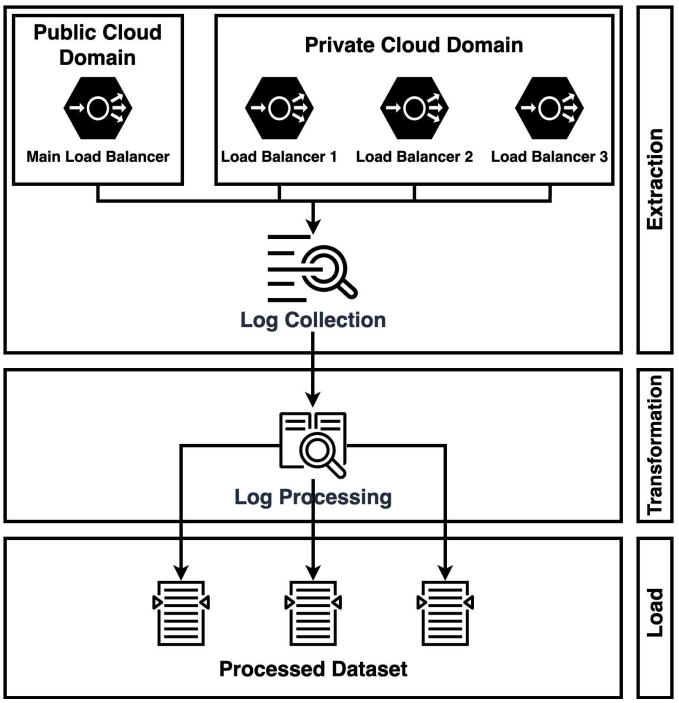

Fig. 2. ETL Architecture

steps include ETL processes for log extraction, transformation, and loading, leading to a comprehensive dataset. Feature extraction, one-hot encoding, and SMOTE balancing are performed before dataset division for training. Rigorous testing and validation assess the deep learning model's accuracy and loss metrics, concluding with result comparison and overall conclusions.

This section outlines the planned methodology using the Extract, Transform, and Load (ETL) architecture illustrated in (Fig. 2). ETL involves extracting, transforming, and loading data from diverse sources, such as access logs from different load balancer instances in public and private clouds. The objective is to consolidate this information into a unified data system or warehouse. The importance of data processing in business and data analysis is highlighted throughout each stage of the ETL method.

A. Data Extraction

In the initial phase of the ETL method, the extraction process involves retrieving data from a range of sources, including databases, text files, spreadsheets, APIs, and more. In this particular scenario, the data originates from multiple

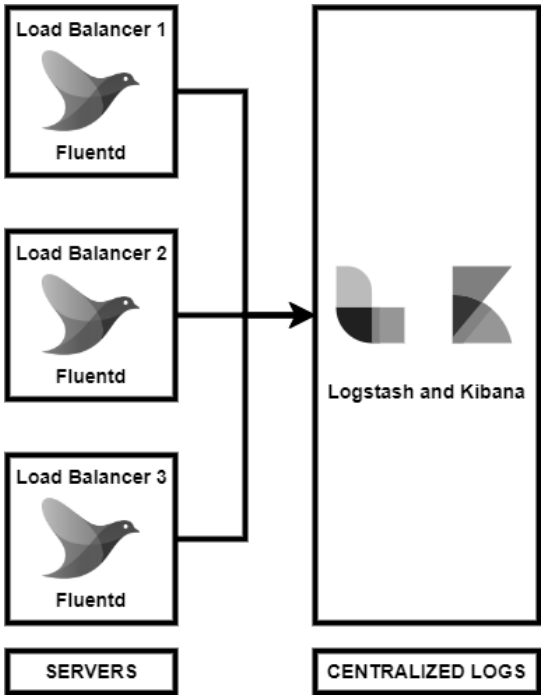

Fig. 3. Fluentd Log Architecture

load balancer instances within the public-cloud and private-cloud infrastructure, primarily sourced from general and access logs in their raw format.

B. Data Transformation

After data extraction, the next step is transformation, involving modifications like cleansing, reformatting, and integration of the load balancer instances raw logs into processed logs. This stage includes filtering, classification, merging, calculations, and normalization to prepare the data for analysis. Elements such as Protocol, Action, Country, Source IP, Destination IP, User, and Timestamp are extracted from the raw logs data in this case.

C. Loading Data Into a Centralized Data Storage

In the final phase, processed data is periodically transferred to a centralized warehouse illustrated in (Fig. 3) to meet business needs. The ETL method remains crucial for organizations managing diverse data sources, ensuring high-quality and relevant data. Fluentd, a versatile and cost-effective tool, handles log aggregation and processing in multi-tenant environments. The Fluentd system, with its components—agent, Logstash, and Kibana facilitates log collection, storage, and querying. A custom tool categorizes information from diverse log entries, aiding decision-making or serving as a data training model for various applications.

This research systematically analyzes log file entries, categorizing data by distinct characteristics and classifying it by protocol and command using (Algorithm 1). Subsequently, (Algorithm 2) categorizes processed log entries into Normal Traffic, Network Anomaly Traffic, and Malicious Traffic,

TABLE I. PROTOCOL AND COMMAND

Protocol	Command
SSH	ssh, scp, sftp
FTP	ftp, put, get, ls, mget, mput
Kerberos	kinit, klist, kdestroy
MySQL	mysql, mysqladmin, mysqldump
HTTP	curl, wget, httpie, lynx, links
Nmap	nmap, nping, ncat

providing a comprehensive understanding of network activities and a valuable tool for prioritizing and addressing issues. The filtered log specifically includes entries related to specific protocols and commands. The protocols and commands are mentioned in the TABLE I.

The log selectively captures activities associated with specific communication methods, focusing on the execution of particular commands within the mentioned protocols. In essence, the filtered log provides a more targeted and detailed view of relevant actions within the specified network or system.

1) Deep Learning as Classification Algorithms: We employ five deep learning algorithms to construct our approach, namely Simple Neural Network, Deep Neural Network, Artificial Neural Network, Convolutional Neural Network, and LSTM (Long Short-Term Memory). We selected these algorithms based on their distinctive characteristics.

Simple Neural Network (SNN) excels in simplicity and efficiency, making it suitable for resource-constrained tasks. Deep Neural Network (DNN) is adept at capturing complex patterns and hierarchical representations, ideal for intricate datasets. Artificial Neural Network (ANN) is versatile across data types, applied in tasks from image recognition to natural language processing. Convolutional Neural Network (CNN) specializes in spatial hierarchies and feature extraction, particularly effective in image and video analysis. Long Short-Term Memory (LSTM) excels at capturing long-term dependencies in sequential data, making it ideal for tasks involving time-series analysis and natural language processing.

2) Performance analysis: To assess our deep learning model, we used two primary metrics: loss, measuring prediction accuracy, and accuracy, indicating classification correctness. The importance of loss in guiding training, and counting prediction errors, is emphasized in [10]. Categorical cross-entropy, a common loss function for classification, measures the average difference between predicted and actual label distributions. The loss value is calculated and illustrated on loss equation (1).

Also, the research [10] mentions that accuracy is a straightforward yet essential metric that evaluates the model's ability to correctly classify data. It is calculated as the proportion of predictions that match the true labels illustrated on accuracy equation (2).

To assess generalization, we monitor validation metrics val-loss and val-accuracy on a separate dataset. Low val-loss signals proficiency with unseen data, while high val-accuracy indicates effective real-world application and good

Algorithm 1 Log File Classification

specify $protocol$ labels such as SSH, FTP, $Kerberos$, $MySQL$, $HTTP$, $Nmap$
specify $command$ labels such as ssh, scp, $sftp$, ftp, put, get, ls, $mget$, $mput$, $kinit$, $klist$, $kdestroy$, $mysql$, $mysqladmin$, $mysqldump$, $curl$, $wget$, $httpie$, $lynx$, $links$, $nmap$, $nping$, $ncat$

while not at the end of a log file **do**
  split each feature such as $protocol$, $command$
  if $protocol$ = SSH **then**
    if $command$ = ssh or $command$ = $sftp$ **then**
      set $class$ as $ssh-severity$
    else
      set $class$ as $ssh-non-severity$
    end if
  else if $protocol$ = FTP **then**
    if $command$ = put or $command$ = $mput$ **then**
      set $class$ as $ftp-severity$
    else
      set $class$ as $ftp-non-severity$
    end if
  else if $protocol$ = $Kerberos$ **then**
    if $command$ = $kdestroy$ **then**
      set $class$ as $kerberos-severity$
    else
      set $class$ as $kerberos-non-severity$
    end if
  else if $protocol$ = $MySQL$ **then**
    set $class$ as $mysql-non-severity$
  else if $protocol$ = $HTTP$ **then**
    set $class$ as $http-non-severity$
  else
    set $class$ as $nmap-non-severity$
  end if
end while

Algorithm 2 Log File Severity

specify $class$ labels such as $ssh-severity$, $ssh-non-severity$, $ftp-severity$, $ftp-non-severity$, $kerberos-severity$, $kerberos-non-severity$, $mysql-non-severity$, $http-non-severity$, $nmap-non-severity$
specify $level$ labels such as $malicious$, $network-anomaly$, $normal$

while not at the end of a log file **do**
  split each feature such as $class$
  if $class$ = $ssh-severity$ or $class$ = $ftp-severity$ or $class$ = $kerberos-severity$ **then**
    set $label$ as 1
  else if $class$ = $ssh-non-severity$ or $class$ = $http-non-severity$ or $class$ = $nmap-non-severity$ **then**
    set $label$ as 2
  else
    set $label$ as 3
  end if
end while

generalization to new data. Tracking these metrics alongside loss and accuracy comprehensively assesses our deep learning model's performance and generalization ability.

III. Experiments and Analysis

A. Dataset

Our study utilizes a dataset extracted from log files, encompassing diverse network activities across multiple servers and services (refer to Table II). This dataset is sourced from confidential channels within our company's network infrastructure. This research is conducted in the first author's confidential company. It comprises four main subsets (A, B, C, and D), each obtained from distinct load balancer instances, providing a comprehensive overview of network activity for specific services. The largest subset, Dataset A, consists of 10,000 lines and serves as the primary source for training and testing our model, with an 80% allocation for training and 20% for testing purposes. The smaller subsets B, C, and D, each containing 2,000 lines, are obtained from various service load balancers. We incorporate 80% of their training data with Dataset A to enrich the model's ability to generalize by leveraging data diversity. The dataset (see Table II) includes essential attributes for analyzing network interactions, such as 'Protocol' indicating the network protocol, 'Action' describing operations, and 'Source IP'/'Destination IP' identifying IP addresses. The 'User' attribute identifies associated entities, 'Class' categorizes activities, and 'Label' classifies traffic (Label 1 for malicious, Label 2 for anomalies, and Label 3 for normal activity). The primary model development relies on Dataset A, with a segment reserved for testing, while Datasets B, C, and D from various algorithmic sources are utilized for direct model testing. The dataset label distribution is depicted in TABLE III.

B. Simple Neural Network

In TABLE IV, the Simple Neural Network demonstrates variable performance across datasets. It achieves the highest accuracy of 89.01% on Dataset A but exhibits the lowest accuracy of 82.550% on Dataset C. Conversely, it attains the lowest loss percentage of 18.850% on Dataset D and the highest loss of 30.340% on Dataset C. This highlights the network's sensitivity to different datasets, emphasizing the importance of dataset-specific considerations in optimizing its performance.

C. Deep Neural Network

The Deep Neural Network, as outlined in TABLE IV, exhibits diverse performance metrics across various datasets. It achieves its highest accuracy of 89.300% on Dataset B, contrasting with its lowest accuracy of 81.600% on Dataset C. Notably, the network attains its lowest loss percentage,

TABLE II. EXAMPLE OF PROCESSED DATASET

No	Protocol	Action	Country	Source IP	Destination IP	User	Timestamp	Class	Label
0	Kerberos	klist	Hungary	91.188.227.187	172.29.41.33	apache	2022-09-10 11:23:08	kerberos-non-severity	3
1	Nmap	nmap	India	167.233.48.11	192.168.9.78	nginx	2022-09-10 11:23:08	nmap-non-severity	2
2	MySQL	mysqladmin	Japan	49.36.41.214	192.168.136.5	nginx	2022-09-10 11:23:08	mysql-non-severity	3
3	FTP	ls	Germany	185.87.149.153	10.122.121.189	apache	2022-09-10 11:23:08	ftp-non-severity	3
4	Nmap	nping	Canada	207.96.212.240	10.11.121.231	root	2022-09-10 11:23:08	nmap-non-severity	2
5	SSH	ssh	Canada	207.96.237.238	192.168.130.68	root	2022-09-10 11:23:08	ssh-severity	1
6	FTP	put	India	167.233.237.24	172.23.82.43	nginx	2022-09-10 11:23:08	ftp-severity	1
7	Kerberos	kinit	Hungary	91.188.237.125	192.168.136.33	guest	2022-09-10 11:23:08	kerberos-non-severity	2
8	MySQL	mysqldump	Japan	49.36.247.131	192.168.144.6	mysql	2022-09-10 11:23:08	mysql-non-severity	3
9	SSH	ssh	Hungary	91.188.133.116	192.168.232.32	mysql	2022-09-10 11:23:08	ssh-severity	1

TABLE III. LABEL COUNT DISTRIBUTION OF DATASET

Label	Count
Label 1	2689
Label 2	3928
Label 3	3383

18.710%, on Dataset B, while experiencing its highest loss of 32.430% on Dataset C. This underscores the network's sensitivity to dataset nuances and underscores the need for tailored strategies to optimize its performance on different datasets.

D. Artificial Neural Network

The Artificial Neural Network, as elucidated in TABLE IV, displays varied performance metrics across distinct datasets. Its pinnacle accuracy of 89.100% is achieved on Dataset B, while its lowest accuracy of 82.400% occurs on Dataset C. Notably, the network records its lowest loss percentage, 18.630%, on Dataset B, juxtaposed with its peak loss of 37.890% on Dataset C. This underscores the network's adaptability to different datasets and emphasizes the necessity of dataset-specific optimization for maximizing its performance.

E. Convolutional Neural Network

The Convolutional Neural Network (CNN) in TABLE IV displays varying performance across datasets, with its highest accuracy at 89.100% on Dataset B and a minimum of 83.800% on Dataset C. Notably, it records the lowest loss of 18.620% on Dataset B and the maximum loss of 24.270% on Dataset A. This underscores the network's adaptability to different datasets, emphasizing the importance of tailored optimization for improved performance in diverse contexts.

F. Long Short-Term Memory

The LSTM model in TABLE IV displays varied performance metrics across datasets. It attains its highest accuracy of 89.150% on Dataset B, with the lowest accuracy of 83.300% on Dataset C. The model records its minimum loss percentage, 18.570%, on Dataset B, while its maximum loss is 19.580% on Dataset A. This underscores the adaptability of the LSTM model to different datasets and emphasizes the necessity for tailored optimization approaches to enhance its performance in diverse scenarios.

IV. Result and Discussion

In this study, we conducted a performance comparison test evaluating various deep learning algorithms: Simple Neural Network, Deep Neural Network, Artificial Neural Network, Convolutional Neural Network, and Long Short-Term Memory, applied to Datasets A, B, C, and D displayed in TABLE IV.

When analyzing the performance metrics, distinct trends emerged. The Simple Neural Network showed an average accuracy of 85.678% with an average loss of 22.698%. The Deep Neural Network slightly improved accuracy to 86.111% but had a marginally increased average loss of 23.018%. The Artificial Neural Network further improved accuracy to 86.306% but with a higher average loss of 24.408%. In comparison, the Convolutional Neural Network stood out with the highest average accuracy of 86.681% and a lower average loss of 20.248%. The Long Short-Term Memory algorithm also demonstrated competitive accuracy at 86.644% and the lowest average loss of 19.050% among the assessed algorithms.

Throughout our analysis, Dataset B consistently showed superior performance in accuracy across all deep learning algorithms. Particularly, the Long Short-Term Memory algorithm performed exceptionally well on Dataset B, indicating its effectiveness in capturing complex patterns within the data. This consistent trend suggests Dataset B is well-suited for deep learning applications, with LSTM being the most effective algorithm.

Conversely, Dataset C consistently exhibited the lowest accuracy across various deep learning algorithms. The Deep Neural Network algorithm had the lowest accuracy within Dataset C, indicating challenges in capturing underlying patterns. This highlights the importance of considering dataset-specific characteristics when selecting appropriate algorithms and suggests a need for further investigation or preprocessing techniques for Dataset C. Additionally, exploring alternative architectures or incorporating domain knowledge may offer insights into enhancing model performance on this particular dataset.

Our investigation found that Dataset D consistently showed the lowest loss values with the Simple Neural Network algorithm. This suggests the effectiveness of the SNN model in minimizing discrepancies between predicted and actual values within Dataset D, demonstrating its robustness for this dataset.

TABLE IV. DEEP LEARNING ALGORITHM TEST COMPARISON RESULT

Dataset	SNN		DNN		ANN		CNN		LSTM	
	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss
Dataset A	0.89011	0.22480	0.88145	0.21880	0.88272	0.22000	0.88425	0.24270	0.88425	0.19050
Dataset B	0.85500	0.19120	0.89300	0.18710	0.89100	0.18630	0.89100	0.18620	**0.89150**	0.18570
Dataset C	0.82550	0.30340	0.81600	0.32430	0.82400	0.37890	0.83800	0.19190	0.83300	0.19580
Dataset D	0.85650	**0.18850**	0.85400	0.19050	0.85450	0.19110	0.85400	0.18910	0.85700	0.19000
Average	0.85678	0.22698	0.86111	0.23018	0.86306	0.24408	**0.86681**	0.20248	0.86644	**0.19050**

Moreover, the SNN algorithm's superior performance underscores its potential as a reliable tool for handling complex data structures and optimizing predictive accuracy in real-world applications.

However, it's crucial to note that this favorable pattern wasn't uniform across all datasets. Dataset C displayed the worst loss values with the Artificial Neural Network algorithm, indicating the performance of the ANN algorithm may depend on specific dataset characteristics. This underscores the nuanced nature of algorithmic performance and emphasizes the need for a tailored approach in selecting algorithms based on dataset-specific attributes to optimize overall model outcomes.

V. Conclusion

In conclusion, our research underscores the significance of adapting algorithm choice to suit the unique attributes of datasets, as evidenced by our comparison of deep learning algorithms across four distinct datasets (A, B, C, and D). While the Convolutional Neural Network (CNN) emerged as the top performer with the highest average accuracy (86.681%), the Long Short-Term Memory (LSTM) algorithm demonstrated superior performance in Dataset B for accuracy and average loss but encountered challenges in Dataset C, highlighting the need for tailored algorithm selection and potential preprocessing techniques. Conversely, the Simple Neural Network (SNN) consistently minimized loss in Dataset D, suggesting its effectiveness for tasks focusing on loss reduction. Our findings emphasize the complex relationship between algorithmic effectiveness and dataset characteristics, advocating for a careful approach to algorithm selection to achieve optimal results in real-world applications of deep learning models.

References

[1] S. Anshuman and Brij B. Gupta, "Distributed Denial-of-Service (DDoS) Attacks and Defense Mechanisms in Various Web-Enabled Computing Platforms: Issues, Challenges, and Future Research Directions," 2022 International Journal on Semantic Web and Information Systems (IJSWIS), vol.18, no.1, pp. 1-43, 2022.

[2] R. Mittal, V. Malik, V. Singh, J. Singh and A. Kaur, "Integrating Genetic Algorithm with Random Forest for Improving the Classification Performance of Web Log Data," 2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC), pp. 177-181, 2020.

[3] B. Abibullaev, I. Dolzhikova and A. Zollanvari, "A Brute-Force CNN Model Selection for Accurate Classification of Sensorimotor Rhythms in BCIs," in IEEE Access, vol. 8, pp. 101014-101023, 2020.

[4] M. D. Hossain, H. Ochiai, F. Doudou and Y. Kadobayashi, "SSH and FTP brute-force Attacks Detection in Computer Networks: LSTM and Machine Learning Approaches," 2020 5th International Conference on Computer and Communication Systems (ICCCS), pp. 491-497, 2020.

[5] L. Tsung-Han, C. Lin-Huang, and S. Chao-Wei, "Deep Learning Enabled Intrusion Detection and Prevention System over SDN Networks," 2020 IEEE International Conference on Communications Workshops (ICC Workshops), pp. 1-6, 2020.

[6] R. B. Said and I. Askerzade, "Attention-Based CNN-BiLSTM Deep Learning Approach for Network Intrusion Detection System in Software Defined Networks," 2023 5th International Conference on Problems of Cybernetics and Informatics (PCI), pp. 1-5, 2023.

[7] J. Hancock, T. M. Khoshgoftaar and J. L. Leevy, "Detecting SSH and FTP Brute Force Attacks in Big Data," 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 760-765, 2021.

[8] V. Robson, A. Arthur, L. Renata, S. Muhammad, C. Dick, H. Pedro, and Z. Demostenes, "Intrusion Detection System Based on Fast Hierarchical Deep Convolutional Neural Network," in IEEE Access, vol. 9, pp. 61024-61034, 2021.

[9] S. Hojjatinia, M. Monshizadeh and V. Khatri, "A Deep Intrusion Detection Model for Network Traffic Payload Analysis," 2023 International Conference on Software, Telecommunications and Computer Networks (SoftCOM), pp. 1-7, 2023.

[10] B. B. Gupta, A. Gaurav and D. Peraković, "A Big Data and Deep Learning based Approach for DDoS Detection in Cloud Computing Environment," 2021 IEEE 10th Global Conference on Consumer Electronics (GCCE), pp. 287-290, 2021.

[11] S. A. Christila and R. Sivakumar, "Multi-Layer Ensemble Deep Reinforcement Learning based DDoS Attack Detection and Mitigation in Cloud-SDN Environment," 2022 4th International Conference on Circuits, Control, Communication and Computing (I4C), pp. 451-455, 2022.

[12] M. Li, B. Zhang, G. Wang, B. ZhuGe, X. Jiang and L. Dong, "A DDoS attack detection method based on deep learning two-level model CNN-LSTM in SDN network," 2022 International Conference on Cloud Computing, Big Data Applications and Software Engineering (CBASE), pp. 282-287, 2022.

[13] L. Wang and Y. Liu, "A DDoS Attack Detection Method Based on Information Entropy and Deep Learning in SDN," 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), pp. 1084-1088, 2020.

[14] R. Karthiga, G. Usha, N. Raju and K. Narasimhan, "Transfer Learning Based Breast cancer Classification using One-Hot Encoding Technique," 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), pp. 115-120, 2021.

[15] N. A. Azhar, M. S. M. Pozi, A. M. Din and A. Jatowt, "An Investigation of SMOTE Based Methods for Imbalanced Datasets With Data Complexity Analysis," in IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 7, pp. 6651-6672, 1 July 2023.