Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 45 No. 1 (2024)

Deep Learning-Based Optimization of an IoT-Enabled Big Data Analytics Architecture for Edge-Cloud Computing

Himani Jain[1*], Monika Saxena[2]

[1*]Research ScholarDepartment of Computer Science,Banasthali Vidyapith
[2]Associate Professor, Department of Computer Science, Banasthali Vidyapith

Abstract-

A proliferation of data generated by the Internet of Things (IoT) and Big Data Analytics (BDA) has revolutionized the decision-making process. In distributed IoT networks, real-time processing encounters obstacles. This research investigates the application of deep learning to edge-cloud computing to optimize BDA. The IoT Botnet Dataset, which comprises attributes of internet transactions, is employed. The procedure entails a number of stages, including as cleaning the data, use Term Frequency-Inverse Document Frequency (TF-IDF) Vectorization to extract features, and using Principal Component Analysis (PCA) to identify the most relevant features. A hybrid model is proposed in this research that combines Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Deep Autoencoders for data analysis. The hybrid model trained well, with a tendency toward reducing training and validation losses, which settled at 0.0063 and 1.1182e-04, respectively, after 100 iterations. At the Initial Threshold, the model demonstrated elevated precision (0.97) and recall (0.954), accompanied by a minimal false negative rate (FN) of 0.13% and a false positive rate (FP) of 9%. At the Final Threshold, recall improved to 0.996, reducing the FP rate to 6% and increasing the FN to 2.59%, while precision decreased marginally to 0.943. The hybrid model exhibited enhanced precision (0.943) and recall (0.996) when compared to the "Auto Encoder" model. However, it did demonstrate a marginally higher false positive (FP) rate of 6% and false negative (2.59%).

Keywords: Internet of Things, Big Data Analytics, Deep Learning, Cloud Computing, Edge Computing

1. Introduction

Kevin Ashton coined the term Internet of Things (IoT) [1] in 1999 for supply chain management [2]. Smarter items can communicate with one another and the Internet[3]. With their unique digital identifiers, all these entities (things) on the Internet of Things may be remotely organized, managed, and controlled, expanding their reach far beyond their physical locations[4]. As the number of connected devices and smart items proliferates, the IoT continues to expand the spectrum of creative, inventive, and intelligent applications available to us[5]. These applications include smart healthcare[6], smart cities[7], smart agriculture[8], crowd sensing[9,10], crowdsourcing[11], etc., as shown in Figure 1[12].The idea behind edge computing is to bring processing closer to the network edge and data sources, which could solve the problems[13]. IoT connects objects to the Internet and other devices, allowing them to collect environmental data. Data analysis is crucial for implementing smart systems including smart cities, healthcare, transportation, and energy, which IoT enables[14]. With edge computing, computation is moved from the cloud or a centralized system to the periphery of the network, close to the data sources, where it can significantly reduce both network traffic and latency. While edge computing has been widely used for mobile task offloading[15] and content distribution[16], its implementation for data analytics has been more restricted[17,18]. Figure 1 illustrates the difference between Cloud and Edge Computing.

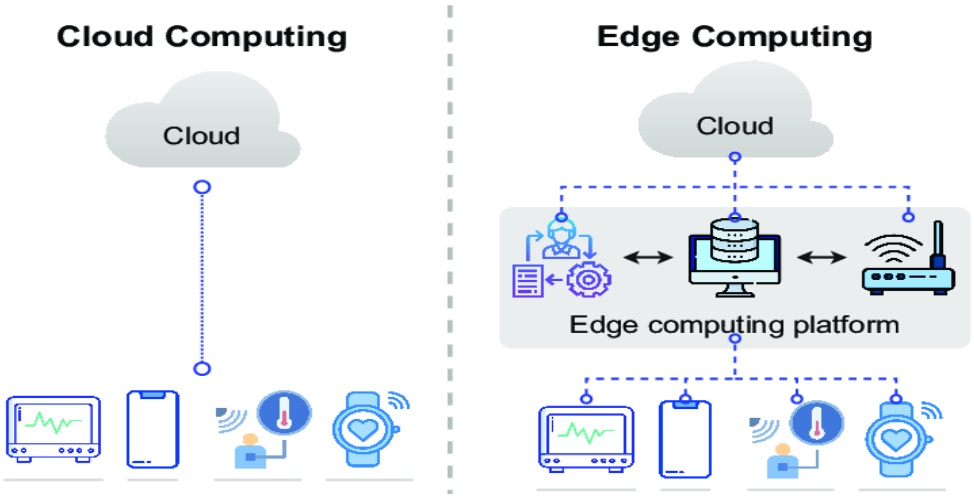

Figure 1. Cloud and Edge Computing[19]

Edge computing reduces latency and high-speed connections by processing data locally. It's great for real-time systems like IoT and autonomous cars but lacks scalability and redundancy. Cloud computing transmits data from remote centers, increasing latency. Its scalability, availability, and storage make it ideal for data analytics and AI training. Edge computing is cheaper to process, but cloud computing requires constant upkeep. Specific use cases and requirements determine the choice[20].Edge computing overcomes problems including poor reaction times, high bandwidth prices, excessive energy use, and inadequate security, spawning many new and fascinating fields of study and work[21].

Computing technology at the network's edge is called edge computing. Edge resources include data sources, cloud data centers, computing power, and communication connectivity. Edge processing protects users' privacy more than cloud uploading[22]. Edge computing is motivated by the demand for better computing methods as technology advances, yet present methods fail[23]. The IoT-endpoint edge computing architecture is shown in Figure 2.

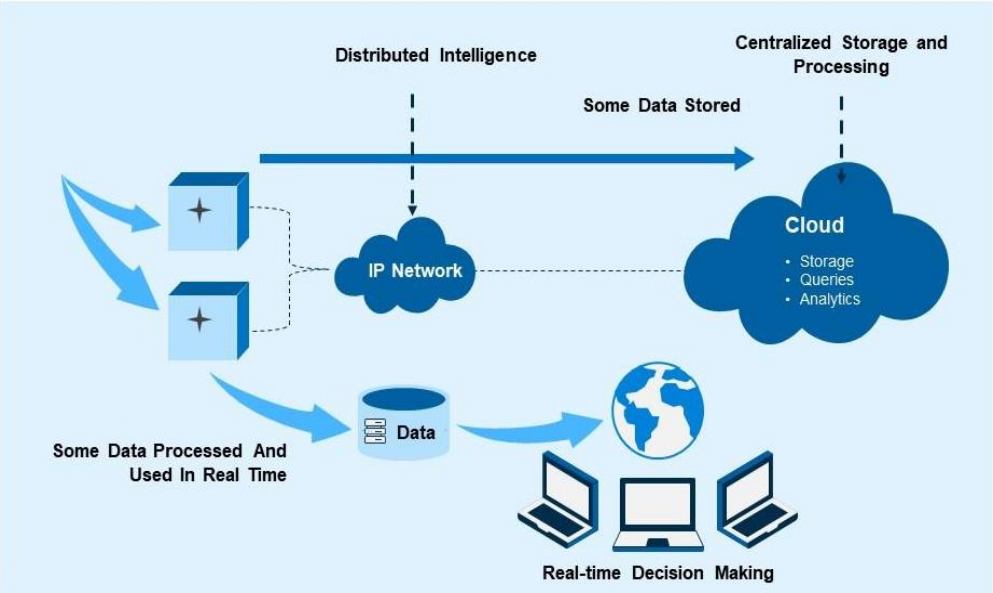

Figure 2. Edge Computing Architecture with IoT endpoints[24]

Deep Learning's capacity to analyzelabeled and unlabeled data, which are abundant in Big Data, can help handle a range of data. In recent years, Deep Learning has become increasingly relevant for Big Data analytics[25]. Deep Learning's capacity to build hierarchical representations for both labeled and unlabeled, structured, and unstructured input data is very important [26]. Higher-order representations can be learned from lower-level inputs using a hierarchical representation. Transcription of spoken words into text, object recognition from photographs, and motion analysis from videos are all examples [27]. Many real-world sources provide massive

amounts of labeled and unlabeled data in real-time[28]. The paper tackles IoT technological issues, particularly Big Data Analytics for Edge-Cloud Computing architecture. It finds data processing and feature selection inefficiencies that can hinder network attack and anomaly detection. The study uses deep learning to uncover patterns and representations from a cleaned IoT Botnet Dataset to improve this framework. The model is going to be thoroughly tested for network attack and anomaly detection and categorization. The research purposes encompass the development of a framework for optimizing IoT-enabled big data analytics using deep learning, the investigation of deep learning-driven security mechanisms for secure and real-time data transmission and processing in a distributed edge-cloud architect, and the reduction of latency and energy consumption in hybrid cloud and edge computing configurations.

2. Literature of Review

A review of the literature reveals that many authors have attempted this method and published their results.

Almutairi et al., (2022)[29]illustrate that themulti-user, multi-cloud edge computing with delay-optimal offloading of tasks for several users. Minimizing Unmanned Aerial Vehicles (UAVs)[30] overall service time is the goal of this optimization model, which makes use of Integer Linear Programming (ILP)[31] methods. The simulation results demonstrate that the proposed approach is effective over a large fleet of UAVs, decreasing service time by 33.5% for edge execution policies and 55.0% for cloud execution policies.

Hussein et al., (2022)[32] offered an offloading method for IoT-based processes in a powerful MEC setting. An optimization problem that considers task dependency, communication costs, workflow restrictions, device energy consumption, and the heterogeneous properties of the edge environment makes up the suggested task-based offloading technique. The experimental findings demonstrate that the suggested algorithm and offloading technique significantly reduce response times for IoT-based processes while preserving the least amount of energy consumption for mobile devices.

Ahmed et al., (2022)[33] provide a segmentation-based method for multiple-item detection using IoTs-enabled smart surveillance. The focus is on enhancing surveillance applications in smart cities through the integration of collaborative drones, deep learning, and IoTs. Based on experimental evidence, researchers know that data augmentation improves the effectiveness of the method by yielding accurate results in multiple object segmentation. The created method achieves an accuracy of 93% with ResNet-50 (Residual Neural Network) and 92% with VGG-16, while MobileNet achieved 95% accuracy.

Babar et al., (2022)[34]designed a framework for edge-cloud computing that uses machine learning to analyze large amounts of data gathered from the IoT. The proposed framework involves the development of an edge intelligence module that effectively processes and stores large-scale data at the periphery of the network, incorporating cloud technologies.The efficacy of the proposed data design is evaluated by the implementation of a simulation utilizing an authentic dataset in Apache Spark.

Apostol et al., (2021)[35]suggest a method for detecting anomalies in IoT botnet activity by employing unsupervised deep learning methods. The study conducted an empirical evaluation of the suggested approach, testing its ability to detect threats on both balanced and imbalanced datasets. The proposed strategy has been shown to obtain high accuracy (99.7%), high precision (0.99), and high recall (99%) in experiments.

Lahoura et al., (2021)[36] suggested a system for leveraging the cloud-based machine learning classifier Extreme Learning Machine (ELM) in the diagnosis of breast cancer. The healthcare sector can benefit from cloud computing since it enables constant access to the system whenever it is needed. The trial outcomes show an accuracy of 0.9868, precision of 0.9054, recall of 0.9130, and F1-score of 0.8129.

Gong et al., (2021)[37] illustrate thata qualitative case study was conducted at the Dutch Tax and Customs Administration to investigate enterprise architecture roles and capabilities for the implementation of big data analytics. The results show that large businesses that can effectively capture the competencies are more likely to employ EA to overcome the unique problems of BDA adoption and implementation.

Khayyat et al., (2020)[38] provide an innovative deep learning-based computational offloading algorithm that is adapted to distributed, edge-cloud computing infrastructures based on vehicles. A concept of computational offloading as well as resource allocation is incorporated to make sure that shared resources are utilized effectively by several vehicles, hence decreasing the total time and cost of energy of running the system. Results from computer simulations demonstrate that the suggested algorithm is capable of fast convergence and much lower system-wide consumption than benchmark solutions.

Tuli et al., (2020)[39]introduce a real-time scheduler based on A3C to facilitate distributed, concurrent learning in stochastic Edge-Cloud settings. To make effective scheduling decisions, authors employ the R2N2

architecture, which allows be recordingof a wide variety of host and task information in addition to temporal patterns. The suggested model is flexible in that it may adjust its hyper-parameters to meet the needs of a given application. Through sensitivity analysis, the authors explain the rationale behind the hyper-parameter selections.Extensive iFogSim and CloudSim simulation trials on the real-world Bitbrain dataset showed that the method reduced energy consumption by 14.4%, response time by 7.74%, SLA violations by 31.9%, and cost by 4.64%.

Castellanos et al., (2020) [40]Using a design process method built on the Attribute-Driven Design (ADD) technique and the Architecture trade analysis method (ATAM), demonstrate how to specify, implement, and measure performance metrics in BDA systems supported by domain-specific modeling and DevOps. These findings show that ACCORDANT is better suited to applications that require several iterations or to follow-on apps where reusing architectural components helps speed up development.

Jameel et al., (2019)[41]offer an investigation of the confidentiality of SWIPT-based cooperative NOMA systems and optimization based on deep learning. When a nearby user acts as a cooperative node when an eavesdropper is present, researchers obtain an analytical equation for the intercept probability. The findings prove the efficiency and effectiveness of deep learning optimization compared to more traditional approach algorithms that search iteratively.

Femminella et al., (2016)[42]examine the efficiency of a Hadoop benchmark suite in a testbed simulating an edge computing deployment, using both real and virtual infrastructure. The results of the experiments show that it is still practical to run Hadoop in a small cloud, despite the expected performance reduction in virtualized settings compared to the native one.

3. Research Methodology

The concept of designed architecture is examined regarding different techniques of research.

3.1 Dataset Description

The "IoT Botnet Dataset" is a trove of transactional data from the internet, with numerous attributes describing each one. The provided data encompasses various data elements, including unique row identifiers (pkSeqID), the start and end times of each record (Stime and Ltime), flow state flags (Flgs and flgs_number), textual and numerical representations of transaction protocols (Proto and proto_number), source and destination IP addresses and port numbers (Saddr, Sport, Daddr, Dport), as well as details regarding packet and byte counts (Pkts, Bytes, Spkts, Dpkts, Sbytes, Dbytes), transaction states and their numerical representations (State and state_number), record duration metrics (Dur, Mean, Stddev, Sum, Min, Max), and various other statistics pertaining to packets and bytes per source and destination IP, per protocol, and per port.

Features of the dataset include counts of incoming connections by source and destination IP address, as well as average rates per protocol and IP address. Finally, it offers categorical information about the traffic's category and subcategory as well as a class label (Attack) indicating whether the traffic is regarded normal (0) or an attack (1). It is likely that Deep Learning models for identifying intrusions into IoT botnets are trained and evaluated using this dataset.

3.2 Technique Used

The following are the important techniques used in the proposed methodology:

Term Frequency-Inverse Document Frequency (TF-IDF) Vectorization

TF-IDF is designed to assign weight to terms (words) within pre-processed documents [43]. In order to assign relative weight to individual words, this method takes into account not only how often a word appears in the current document, but also how often it appears in other publications. The TF and IDF values for each document are to be determined using the TF-IDF algorithm. Equation (1) [44] is utilized so that the weight of each phrase t in the document d may be computed.

$$W_{dt} = TF_{dt} * IDF_t \quad (1)$$

In the context of document retrieval, the weight of a document $\boldsymbol{d}$ with respect to a specific word $\boldsymbol{t}$ can be denoted as $\boldsymbol{W}$. Additionally, $\boldsymbol{tf}$ represents the count of words being searched for within a document [45]. The TF-IDF vectorized features retrieved from the IoT Botnet Dataset are shown in Table 1.

Table 1. Feature Extraction

Feature	Description
Sport	Source port number
Dport	Destination port number
Saddr	Source IP address
Daddr	Destination IP address
Bytes	Total number of bytes in transaction
Spkts	Source-to-destination packet count
Pkts	Total count of packets in transaction
Dpkts	Destination-to-source packet count
Sbytes	Source-to-destination byte count
Dbytes	Destination-to-source byte count
Rate	Total packets per second in transaction
Srate	Source-to-destination packets per second
Drate	Destination-to-source packets per second
TnP_PSrcIP	Total Number of packets per source IP
TnP_PDstIP	Total Number of packets per Destination IP
TnBPSrcIP	Total Number of bytes per source IP
TnBPDstIP	Total Number of bytes per Destination IP
Category	Traffic category
Subcategory	Traffic subcategory
Attack	Class label: 0 for Normal traffic, 1 for Attack Traffic

3.2.1 Principal Component Analysis (PCA)

PCA is a multivariate technique for analyzing a data table in which observations are characterized by many quantitative dependent variables that are connected among themselves.The data is extracted from the table and represented as a set of new orthogonal variables called principal components; these are then used to detect patterns of similarity between observations and variables, which are then displayed as dots on maps [46]. Feature sets can be refined further through the use of (PCA as a), a method that decreases the number of dimensions in the data without altering the essential details [47]. The following equation provides a useful representation of PCA:

$$X_{new} = X.W \tag{2}$$

Where:

- X_{new} is the transformed feature set.
- X *is the original feature set.*
- W *represents the transformation matrix obtained from PCA.*

Table 2 illustrates the selected features using Principal Component Analysis.

Table 2. Feature Selection using PCA.

Feature	Description
Sport	Source port number
Dport	Destination port number
Pkts	Total count of packets in transaction
Spkts	Source-to-destination packet count
Dpkts	Destination-to-source packet count
Rate	Total packets per second in transaction
Attack	Class label: 0 for Normal traffic, 1 for Attack Traffic
Category	Traffic category
Subcategory	Traffic subcategory

3.2.2 Deep learning approaches

Data representations at different levels of abstraction can be learned by computer models with deep learning [48]. The algorithms that make up Deep Learning systems use a multi-level, hierarchical learning technique to figure out how to extrapolate the most insightful possible abstract representations of the input. A deep learning model can be taught to distinguish between botnet and authentic data after features have been extracted and

selected. CNNs, RNNs, and Deep Autoencoder are only few of the deep learning models that can be utilized for this purpose. The deep learning model is trained by feeding it the features that have been hand-picked from the training data and their associated labels (botnet/legitimate). The model learns, through alterations to its internal parameters, how to map the features to the labels. After the model has been trained, it can be used to make predictions about the labels of unseen data.

- **Convolutional Neural Networks (CNNs)**

Extraction of spatial information from structured data is a strong advantage for Convolutional Neural Networks (CNNs), especially when dealing with images or data organised in a grid-like style [49]. CNNs excel at processing grid-like data, such as images or, in this case, the derived network-related metrics. Convolutional layers, which search the data for local patterns and gradually develop more sophisticated representations, allow these models to learn hierarchical features. CNNs likely help in this process by capturing detailed correlations within the network data, allowing the model to effectively detect and respond to anomalies or attacks. Their usefulness in this context is demonstrated by their ability to accurately classify or predict network-related outcomes through the extraction of significant characteristics from complicated datasets.

- **Recurrent Neural Networks (RNNs)**

In this research, the training data is put through a series of deep learning techniques, including CNNs, RNNs, and a Deep Autoencoder. RNNs are used to analyse temporal elements of IoT device network operations because they are trained to handle sequential data. RNNs are built to handle time-dependent data, making it possible for them to detect trends and associations across longer periods of time [50]. Network attacks and abnormalities can be detected and classified with the help of RNNs because of their ability to accurately simulate the time-series nature of network traffic. Using RNNs and other deep learning architectures, the system may learn complex patterns and representations from the data, improving its ability to recognize and respond to security risks in IoT networks.

- **Deep Autoencoder**

An autoencoder is a bottleneck layer in a fully linked artificial neural network. The encoder and the decoder are the two main components of an autoencoder. Encoders are responsible for transforming high-dimensional feature vectors into lower-dimensional representations at the bottleneck layer. Learning in an autoencoder is driven by the features provided as input and as output, therefore the input characteristics are first presented in a lower dimensional form by the bottleneck layer before being translated into a higher dimensional form by the decoder [51].

The Deep Autoencoder is utilized in this context to acquire a condensed representation of the pre-processed data acquired from the IoT Botnet Dataset. The Deep Autoencoder is trained on a set of characteristics with the intention of discovering hidden structures and patterns in the data. After compression, this representation is analyzed and classified to identify network threats and anomalies. A Deep Autoencoder helps the model extract relevant features from the data, improving the performance and accuracy of the deep learning approach.

3.3 Proposed Methodology

The defined procedure encompasses six primary stages for the analysis of data derived from the IoT Botnet Dataset. Initially, data is gathered, typically encompassing pertinent information pertaining to IoT devices and their respective networks. Subsequently, the gathered data undergoes preprocessing procedures, encompassing error correction, data standardization, and missing value management. Subsequently, the TF-IDF vectorization method is employed to extract valuable features, including but not limited to IP addresses, port numbers, packet counts, and network metrics. Using PCA for feature selection, a subset of these characteristics is subsequently chosen for analysis. The chosen attributes consist of data such as the counts of packets, the types, and categories of attacks, as well as the source and destination ports. The dataset is subsequently partitioned into training and testing sets, and the performance of a deep learning model is assessed for representation learning and pattern recognition. The efficacy of the model in predicting or classifying network attacks or anomalies is ultimately evaluated through performance analysis of its outputs. The Proposed layout in Figure 3 shows the operation depicted in diagrammatic form.

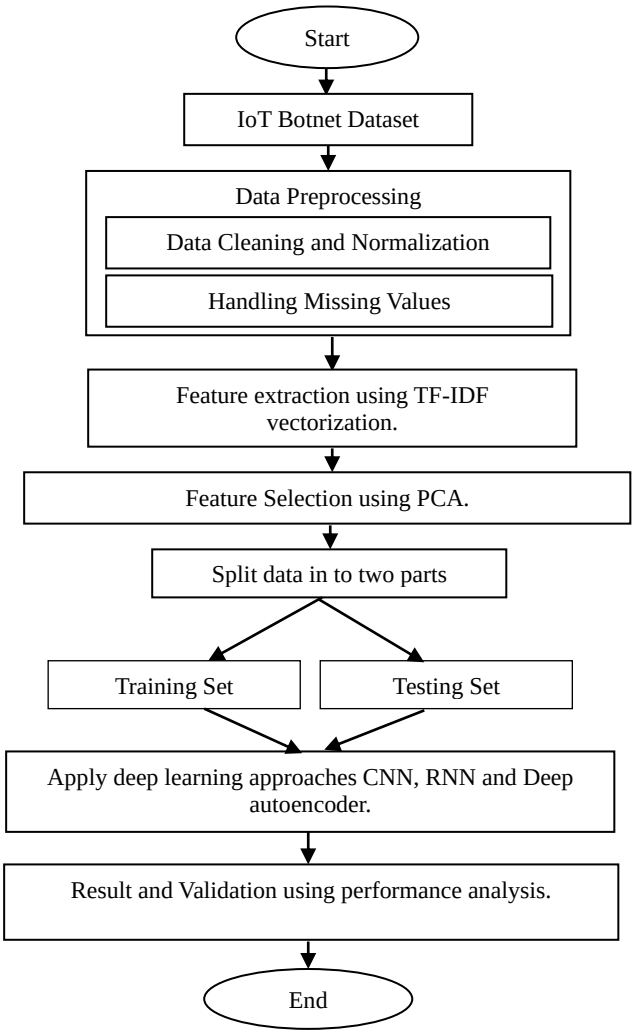

Figure 3. Proposed Methodology

3.4 Proposed Algorithm

Here's a fully mathematical algorithm for the steps outlined:

Step 1: Data Collection

- Let **D** be the IoT Botnet Dataset containing information related to IoT devices and their network activities.

Step 2: Data Preprocessing

- Data Cleaning
  - $\mathbf{D_{clean}} = \mathbf{clean}(\mathbf{D})$ removes errors and inconsistencies from the dataset.
- Normalization
  - $\mathbf{D_{normalize}} = \mathbf{normalize}(\mathbf{D_{clean}})$ scales the data.
- Handling Missing Values:
  - $\mathbf{D_{final}} = \mathbf{handle\_missing}(\mathbf{D_{normalized}})$ fills in or remove missing values.

Step 3: Feature Extraction

- TF-IDF Vectorization:

- $\mathbf{X = TF - IDF(D_{final})}$ extracts features using the Term Frequency-Inverse Document Frequency vectorization approach.

Step 4: Feature Selection

- Principle Component Analysis (PCA):
- $\mathbf{X_{selected} = PCA(X)}$ selects a subset of features **(Sport, Dport, Pkts, Spkts, Dpkts, Rate, Attack, Category, Subcategory)** using PCA.

Step 5: Data Split

- $\mathbf{X_{train}, X_{test} = split(X_{select})}$ splits the data into a training set (X_{train}) and a testing set $(\mathbf{X_{test}})$.

Step 6: Deep Learning

- Let $\mathbf{f_{DL}}$ be the deep learning model composed of Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and a Deep Autoencoder.
- Training:
  - $\mathbf{f_{DL} = train\ (X_{train})}$ trains the model on the training data.

Step 7: Performance Analysis

- **Testing:**
  - $\mathbf{y_{pred} = f_{DL}(X_{test})}$ predicts outcomes on the testing set.
- **Evaluation:**
  - Performance metrics$(\mathbf{P_{metrics}})$ are calculated to assess the model's performance.

4. Result and Discussion

The results obtained by a hybrid model comprising CNNs, RNNs, and Deep Autoencoder in detecting attacks specific to IoT botnets are detailed and discussed in this section. The study begins with an explanation of how the optimal network configuration and hyper-parameters were determined, and then continues with an in-depth analysis of the results. The research then delves into a more in-depth analysis of the model's predictions on the dataset that was used for validation. The result of the study includes a comparison of the study's findings to those of previous studies that have utilized the same or a comparable Bot-IoT dataset.

4.1 Training the hybrid model

The paper explains how to do Big Data Analytics in the cloud at the edge using the Internet of Things. The architecture relies on information gathered from the "IoT Botnet Dataset," which details internet transactions with a variety of properties. Internet Protocol (IP) addresses, port numbers, packet counts, and other attributes are all included in the collection. Data is pre-processed, features are extracted using TF-IDF vectorization, and final features are selected using PCA. After that, it analyzes the data using deep learning methods including CNNs, RNNs, and Deep Autoencoders. Using the provided loss and validation loss data, the hybrid model produces amazing results.

The training and validation losses of the model consistently go down over the course of training. The model quickly improves its predictions after an initial loss of 0.5334 in the first epoch, reaching a remarkable low loss of 0.0063 by the 100th epoch. The validation loss exhibits a similar pattern, going from 0.3181 to a value as small as 1.1182e-04. These outcomes indicate that the hybrid model learns to efficiently capture temporal dependencies with the RNN, enhance the representations with the Deep Autoencoder, and extract relevant features from the input data with the CNN. The efficacy of this model in producing accurate and precise predictions seems to have been enhanced by the integration of these methods. Figure 4 depicts the distinction between training loss and validation loss.

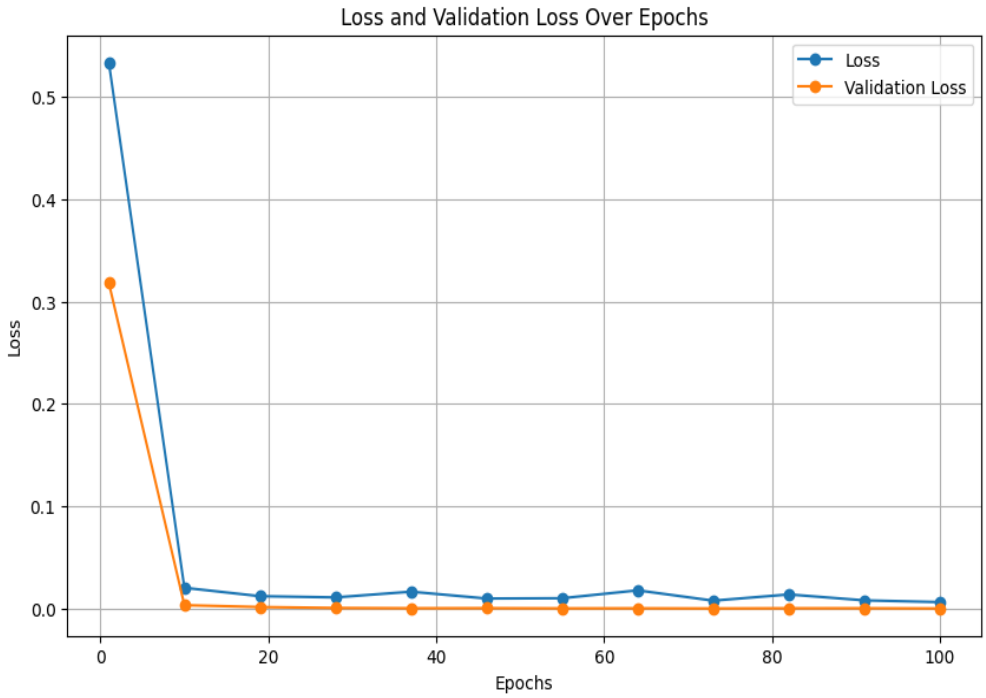

Figure 4. Training Progress Over 100 Epochs: Loss and Validation Loss.

4.2 Predictions

Two threshold levels, Initial Threshold and Final Threshold, are presented in the table, along with associated performance data for the corresponding classification model. The model has an accuracy of 0.97 at the Initial Threshold, meaning that 97% of the positive predictions were accurate, and a recall of 0.954, suggesting that 95.4% of the true positives were caught by the model. The 9% false positive rate (FP) signifies that 9% of the negative instances were erroneously categorized as positive. Conversely, the FN stands at 0.13%, signifying that just 0.13% of the true positives were erroneously classified as negatives. Conversely, the precision exhibits a marginal decline to 0.943 at the Final Threshold, while the recall demonstrates an improvement to 0.996. This indicates that the model is approximately increasing the percentage of accurate positive identifications. Reduced to a false positive rate of 6%, this indicates fewer incidents were wrongly labelled as positive. However, the false negative rate increases to 2.59%, indicating that a greater proportion of true positives are now being incorrectly labelled as false negatives. Experiment findings for both the initial and final thresholds are outlined in Table 3.

Table 3. Experimental Results.

	Precision	Recall	FP (%)	FN (%)
Initial Threshold	0.97	0.954	9	0.13
Final Threshold	0.943	0.996	6	2.59

Precision, recall, false positives (FP), and false negatives (FN) characteristics are graphically represented in Figure 5 for both the initial and final threshold values. The influence on the accuracy and efficacy of the hybrid model in identifying IoT botnet-specific attacks is highlighted by this visualization, which allows for a clear comparison of the performance metrics at different threshold levels.

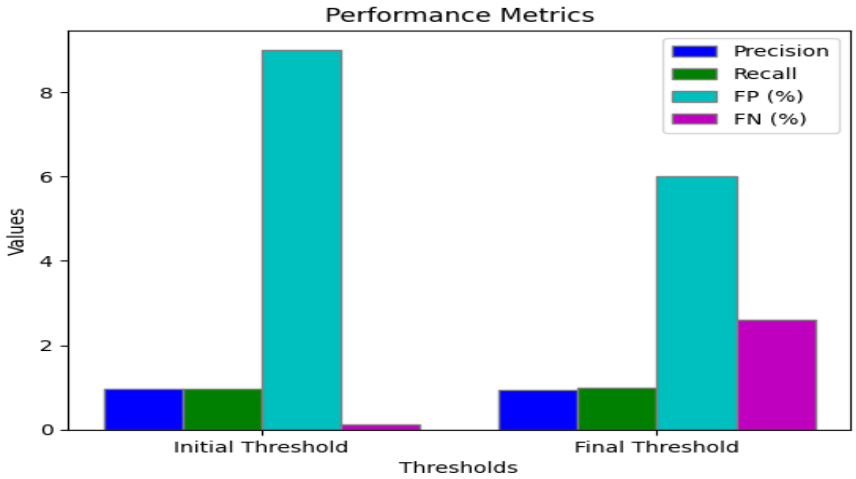

Figure 5. Performance Metrics at Different Thresholds

4.3 Comparative study

The performance metrics of two distinct models are utilized to assess their standing in the comparison table. The "Hybrid Model," an architecture that integrates a Deep Autoencoder, CNNs, and RNNs, exhibits a remarkable precision rate of 94.3%, signifying its ability to accurately discern pertinent instances. Additionally, it obtains a noteworthy recall rate of 99.6%, which signifies its capacity to accurately detect a substantial portion of pertinent instances. In contrast, the aforementioned model exhibits a false negative rate (2.59%) and a 6% false positive rate (FP), respectively, suggesting that it occasionally erroneously categorizes non-relevant instances. Conversely, the precision of the "Auto Encoder" [35] model is marginally lower at 91.6%, indicating a narrow space for enhancement in accurately classifying pertinent instances. Its 100% recall rate indicates that it successfully captures all target instances, but its 10.3% FP rate is somewhat high. The surprising lack of a false-negative rate suggests that it never overlooks important data. Metrics for two models' performances are listed in Table 4.

Table 4. Comparison of Hybrid Model and Auto Encoder model

	Precision	Recall	FP (%)	FN (%)
Hybrid Model	0.943	0.996	6	2.59
Auto Encoder [35]	0.916	1	10.3	0

As shown in Figure 6, the proposed hybrid model compares favorably to an existing model that solely employs Autoencoder [35] when pitted against a model that includes Deep Autoencoder, CNNs, and RNNs. Differences in how these two models work are clearly depicted in this visual comparison.

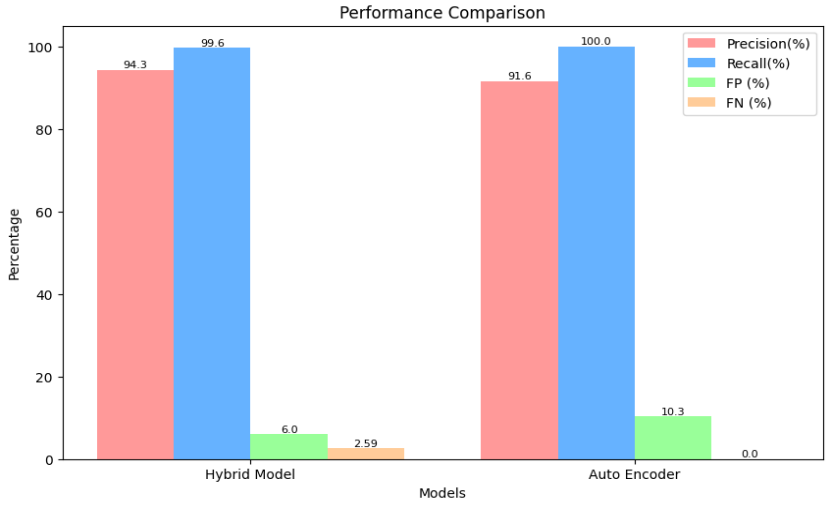

Figure 6. Performance Metrics of Hybrid Model and Auto Encoder

5. Conclusion and Future Scope

IoT and Big Data Analytics provide data-driven decision-making, but encounter hurdles in real-time processing. In order to process IoT data quickly and effectively, this research focuses on integrating deep learning with edge-cloud computing. This study introduces an IoT-enabled Big Data Analytics architecture for Edge-Cloud Computing, which solves the problems of low processing efficiency in real time over dispersed IoT networks. For real-time analytics, it promotes deep learning at the edge to circumvent latency, bandwidth, and privacy concerns. The study intends to construct tailored deep-learning frameworks, explore security measures, and design improved algorithms for efficient data processing. Edge-cloud computing is suggested as a method to deal with the rising data generation and consumption prompted by the IoT. The significance of Deep Learning in Big Data Analytics for deriving actionable insights from massive datasets is also emphasized. Recent initiatives to improve IoT-based system processes are highlighted in a review of relevant literature. The research presents an architecture for IoT-enabled big data analytics in edge-cloud computing.

The architecture uses deep learning models like CNNs, RNNs, and Deep Autoencoder for intrusion detection in IoT botnets, with techniques like TF-IDF vectorization for feature extraction and PCA for feature selection. Data is gathered, pre-processed, features are extracted and selected, data is divided, deep learning is trained, and results are analysed. Over the course of 100 epochs, both the training and validation losses have decreased, demonstrating considerable progress in the model's training process. That the model is successful at feature extraction, temporal dependency capture, and representation refinement is shown by this result. The classification performance at various threshold values reveals that the model reaches 97% precision and 95.4% recall at the Initial Threshold. The precision is slightly lower at 94.3%, but the recall is much greater at 99.6% when using the Final Threshold. In a head-to-head comparison with an Auto Encoder model, the hybrid model is found to be more accurate (94.3% vs. 91.6%) and has a higher rate of correct predictions (99.6% vs. 100%). The false positive rate is slightly higher at 6% compared to 10.3%, and the false negative rate at 2.59% compared to 0%. Based on these findings, it appears that the hybrid model is an effective method for detecting assaults conducted via an IoT botnet. In-depth comparisons with other methods and models could be conducted in future research to confirm the superiority of the suggested hybrid strategy and pinpoint places for improvement.

References

1. Li, Wei, Yuanbo Chai, Fazlullah Khan, SyedRoohUllah Jan, Sahil Verma, Varun G. Menon, Kavita, and X. Li. 2021. “A Comprehensive Survey on Machine Learning-Based Big Data Analytics for IoT-Enabled Smart Healthcare System.” *Mobile Networks & Applications* 26, no. 1: 234–52. https://doi.org/10.1007/s11036-020-01700-6.
2. Evtodieva, T. E., D. V. Chernova, N. V. Ivanova, and J. Wirth. 2019. *‘The internet of things: possibilities of application in intelligent supply chain management.’ Digital Transformation of the Economy: Challenges, Trends, and New Opportunities*: 395–403.
3. Abdollahzadeh, Sanay, and Nima Jafari Navimipour. 2016. “Deployment Strategies in the Wireless Sensor Network: A Comprehensive Review.” *Computer Communications* 91: 1–16.
4. Piccialli, Francesco, and Jai E. Jung. 2017. “Understanding Customer Experience Diffusion on Social Networking Services by Big Data Analytics.” *Mobile Networks & Applications* 22, no. 4: 605–12. https://doi.org/10.1007/s11036-016-0803-8.
5. Qin, S. Joe. 2014. “Process Data Analytics in the Era of Big Data.” *AIChE Journal* 60, no. 9: 3092–100. https://doi.org/10.1002/aic.14523.
6. Baker, Stephanie B., Wei Xiang, and Ian Atkinson. 2017. “Internet of Things for Smart Healthcare: Technologies, Challenges, and Opportunities.” *IEEE Access* 5: 26521–44. https://doi.org/10.1109/ACCESS.2017.2775180.
7. Latif, S., H. Afzaal, and N. A. Zafar. 2018. “Intelligent Traffic Monitoring and Guidance System for a Smart City.” In International Conference on Computing, Mathematics and Engineering Technologies (iCoMET): 1–6. IEEE Publications. https://doi.org/10.1109/ICOMET.2018.8346327.
8. Babar, Muhammad, Fazlullah Khan, Waseem Iqbal, Abid Yahya, Fahim Arif, Zhiyuan Tan, and Joseph M. Chuma. 2018. “A Secured Data Management Scheme for Smart Societies in the Industrial Internet of Things Environment.” *IEEE Access* 6: 43088–99. https://doi.org/10.1109/ACCESS.2018.2861421.
9. Pouryazdan, Maryam, Claudio Fiandrino, Burak Kantarci, TolgaSoyata, DzmitryKliazovich, and Pascal Bouvry. 2017. “Intelligent Gaming for Mobile Crowd-Sensing Participants to Acquire Trustworthy Big Data on the Internet of Things.” *IEEE Access* 5: 22209–23. https://doi.org/10.1109/ACCESS.2017.2762238.
10. Liu, Jinwei, Haiying Shen, Husnu S. Narman, Wingyan Chung, and Zongfang Lin. 2018. “A Survey of Mobile Crowdsensing Techniques: A Critical Component for the Internet of Things.” *ACM Transactions on Cyber-Physical Systems* 2, no. 3: 1–26. https://doi.org/10.1145/3185504.

11. Lashkari, Rezazadeh, Javad, Reza Farahbakhsh, and KumbesanSandrasegaran. "Crowdsourcing and sensing for indoor localization in IoT: A review." 2018. “Bahareh.” *IEEE Sensors Journal* 19, no. 7: 2408–34.
12. Hassan, Rosilah, Faizan Qamar, M. K. Hasan, AzanaHafizahMohd Aman, and Amjed Sid Ahmed. 2020. “Internet of Things and Its Applications: A Comprehensive Survey.” *Symmetry* 12, no. 10: 1674. https://doi.org/10.3390/sym12101674.
13. Ghosh, AnandaMohon, and Katarina Grolinger. 2020. “Edge-Cloud Computing for Internet of Things Data Analytics: Embedding Intelligence in the Edge with Deep Learning.” *IEEE Transactions on Industrial Informatics* 17, no. 3: 2191–200.
14. L’heureux, Alexandra, Katarina Grolinger, Hany F. Elyamany, and Miriam A. M. Capretz. 2017. “Machine Learning with Big Data: Challenges and Approaches.” *IEEE Access* 5: 7776–97. https://doi.org/10.1109/ACCESS.2017.2696365.
15. Chen, Min, and Yixue Hao. 2018. “Task Offloading for Mobile Edge Computing in the Software-Defined Ultradense Network.” *IEEE Journal on Selected Areas in Communications* 36, no. 3: 587–97. https://doi.org/10.1109/JSAC.2018.2815360.
16. Roman, Rodrigo, Javier Lopez, and Masahiro Mambo. 2018. “Mobile Edge Computing, Fog et al.: A Survey and Analysis of Security Threats and Challenges.” *Future Generation Computer Systems* 78: 680–98. https://doi.org/10.1016/j.future.2016.11.009.
17. El-Sayed, Hesham, Sharmi Sankar, Mukesh Prasad, Deepak Puthal, Akshansh Gupta, Manoranjan Mohanty, and Chin-Teng Lin. 2017. “Edge of Things: The Big Picture on the Integration of Edge, IoT and the Cloud in a Distributed Computing Environment.” *IEEE Access* 6: 1706–17. https://doi.org/10.1109/ACCESS.2017.2780087.
18. Kumari, Aparna, Sudeep Tanwar, Sudhanshu Tyagi, Neeraj Kumar, Reza M. Parizi, and K. R. Choo. 2019. “Fog Data Analytics: A Taxonomy and Process Model.” *Journal of Network & Computer Applications* 128: 90–104. https://doi.org/10.1016/j.jnca.2018.12.013.
19. Chengoden, R., N. Victor, T. Huynh-The, G.Yenduri, R. H. Jhaveri, M.Alazab, S. Bhattacharya, P. Hegde, P. K. R. Maddikunta, and T. R. Gadekallu. 2023. “Metaverse for Healthcare: A Survey on Potential Applications, Challenges, and Future Directions.” *IEEE Access* 11: 12765–95. https://doi.org/10.1109/ACCESS.2023.3241628.
20. Hu, Xiaoyan, Lifeng Wang, Kai-Kit Wong, Meixia Tao, Yangyang Zhang, and Zhongbin Zheng. 2019. “The Synergy of Edge and Central Cloud Computing with Wireless Mimo Backhaul.” In IEEE Global Communications Conference (GLOBECOM): 1–6. IEEE Publications. https://doi.org/10.1109/GLOBECOM38437.2019.9014044.
21. Mitra, Anuran, Soumita Biswas, Tinku Adhikari, Arindam Ghosh, Soumalya De, and Raja Karmakar. 2020. “Emergence of Edge Computing: An Advancement over Cloud and Fog.” In 11th International Conference on Computing, Communication and Networking Technologies (ICT): 1–7. IEEE Publications. https://doi.org/10.1109/ICCCNT49239.2020.9225270.
22. Shi, Weisong, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. 2016. “Edge Computing: Vision and Challenges.” *IEEE Internet of Things Journal* 3, no. 5: 637–46. https://doi.org/10.1109/JIOT.2016.2579198.
23. Satyanarayanan, Mahadev, Guenter Klas, Marco Silva, and Simone Mangiante. 2019. “The Seminal Role of Edge-Native Applications.” In IEEE International Conference on Edge Computing (EDGE): 33–40. IEEE Publications. https://doi.org/10.1109/EDGE.2019.00022.
24. https://www.slideteam.net/edge-computing-architecture-with-iot-endpoints.html.
25. Ravi, Daniele, Charence Wong, Benny Lo, and Guang-Zhong Yang. 2017. “A Deep Learning Approach to On-Node Sensor Data Analytics for Mobile or Wearable Devices.” *IEEE Journal of Biomedical & Health Informatics* 21, no. 1: 56–64. https://doi.org/10.1109/JBHI.2016.2633287.
26. Deng, Li. 2011. “An Overview of Deep-Structured Learning for Information Processing.” In *Proc. Asian-Pacific Signal and Information Proc. Annual Summit and Conference (APSIPA-ASC)*: 1–14.
27. Tannahill, Barnabas K., Chris E. Maute, Yunus Yetis, Maryam N. Ezell, Aldo Jaimes, Roberto Rosas, AzimaMotaghi, Halid Kaplan, and Mo Jamshidi. 2013. “Modeling of System of Systems via Data Analytics—Case for ‘Big Data’ in SoS.” In 8th International Conference on System of Systems Engineering: 177–83. IEEE Publications.
28. Sagiroglu, Seref, and Duygu Sinanc. 2013. “Big Data: A Review.” In international conference on collaboration technologies and systems (CTS): 42–7. IEEE Publications. https://doi.org/10.1109/CTS.2013.6567202.
29. Almutairi, J., M. Aldossary, H. A. Alharbi, B. A. Yosuf, and J. M. H. Elmirghani. 2022. “Delay-Optimal Task Offloading for UAV-Enabled Edge-Cloud Computing Systems.” *IEEE Access* 10: 51575–86. https://doi.org/10.1109/ACCESS.2022.3174127.

30. Wakode, Madhuri S. 2021. "Role of AI and Big Data Analytics in UAV-Enabled IoT Applications for Smart Cities." *Unmanned Aerial Vehicles for Internet of Things (IoT) Concepts, Techniques, and Applications*: 193–205.
31. Hussain, M. M., M. M. S. Beg, and M. S. Alam. 2020. "Fog Computing for Big Data Analytics in IoT Aided Smart Grid Networks." *Wireless Personal Communications* 114, no. 4: 3395–418. https://doi.org/10.1007/s11277-020-07538-1.
32. Hussein, Mohamed K., and Mohamed H. Mousa. 2022. "Efficient Computation Offloading of IoT-Based Workflows Using Discrete Teaching Learning-Based Optimization." *Computers, Materials & Continua* 73, no. 2: 3685–703. https://doi.org/10.32604/cmc.2022.026370.
33. Ahmed, Imran, Misbah Ahmad, AbdellahChehri, Mohammad Mehedi Hassan, and Gwanggil Jeon. 2022. "IoT Enabled Deep Learning Based Framework for Multiple Object Detection in Remote Sensing Images." *Remote Sensing* 14, no. 16: 4107. https://doi.org/10.3390/rs14164107.
34. Babar, Muhammad, Mian Ahmad Jan, Xiangjian He, Muhammad Usman Tariq, Spyridon Mastorakis, and Ryan Alturki. 2022. "An Optimized IoT-Enabled Big Data Analytics Architecture for Edge–Cloud Computing." *IEEE Internet of Things Journal* 10, no. 5: 3995–4005.
35. Apostol, Ioana, Marius Preda, Constantin Nila, and Ion Bica. 2021. "IoT Botnet Anomaly Detection Using Unsupervised Deep Learning." *Electronics* 10, no. 16: 1876. https://doi.org/10.3390/electronics10161876.
36. Lahoura, Vivek, Harpreet Singh, Ashutosh Aggarwal, Bhisham Sharma, Mazin Abed Mohammed, RobertasDamaševičius, Seifedine Kadry, and Korhan Cengiz. 2021. "Cloud Computing-Based Framework for Breast Cancer Diagnosis Using Extreme Learning Machine." *Diagnostics* 11, no. 2: 241. https://doi.org/10.3390/diagnostics11020241.
37. Gong, Yiwei, and Marijn Janssen. 2021. "Roles and Capabilities of Enterprise Architecture in Big Data Analytics Technology Adoption and Implementation." *Journal of Theoretical & Applied Electronic Commerce Research* 16, no. 1: 37–51. https://doi.org/10.4067/S0718-18762021000100104.
38. Khayyat, Mashael, Ibrahim A. Elgendy, Ammar Muthanna, Abdullah S. Alshahrani, Soltan Alharbi, and Andrey Koucheryavy. 2020. "Advanced Deep Learning-Based Computational Offloading for Multilevel Vehicular Edge-Cloud Computing Networks." *IEEE Access* 8: 137052–62. https://doi.org/10.1109/ACCESS.2020.3011705.
39. Tuli, Shreshth, ShashikantIlager, KotagiriRamamohanarao, and RajkumarBuyya. 2020. "Dynamic Scheduling for Stochastic Edge-Cloud Computing Environments Using a3c Learning and Residual Recurrent Neural Networks." *IEEE Transactions on Mobile Computing* 21, no. 3: 940–54. https://doi.org/10.1109/TMC.2020.3017079.
40. Castellanos, C., B. Perez, D. Correal, and C. A. Varela. 2020. "A Model-Driven Architectural Design Method for Big Data Analytics Applications." In IEEE International Conference on Software Architecture Companion (ICSA-C): 89–94. IEEE Publications. https://doi.org/10.1109/ICSA-C50368.2020.00026.
41. Jameel, F., W. U. Khan, Z. Chang, T.Ristaniemi, and J. Liu. 2019. "Secrecy Analysis and Learning-Based Optimization of Cooperative NOMA SWIPT Systems." In IEEE International Conference on Communications Workshops (ICC Workshops): 1–6. IEEE Publications. https://doi.org/10.1109/ICCW.2019.8756894.
42. Femminella, Mauro, Matteo Pergolesi, and Gianluca Reali. 2016. "Performance Evaluation of Edge Cloud Computing System for Big Data Applications." In 5th IEEE International Conference on Cloud Networking (Cloudnet): 170–5. IEEE Publications. https://doi.org/10.1109/CloudNet.2016.56.
43. Nurjannah, Musfiroh, HamdaniHamdani, and IndahFitri Astuti. 2016. "PenerapanAlgoritma Term Frequency-Inverse Document Frequency (TF-IDF) Untuk Text Mining." *InformatikaMulawarman* 8, no. 3: 110–3.
44. Maarif, Abdul Aziz. 2015. "'PenerapanAlgoritma TF-IDF untukPencarian Karya Ilmiah.' Dok. Karya Ilm.| Tugas Akhir| Progr. Stud. Tek." *Inform* S1| Fak. IlmuKomput.| Univ. Dian Nuswantoro Semarang 5: 4.
45. Alfarizi, Muhammad Ibnu, LailisSyafaah, and Merinda Lestandy. 2022. "'Emotional Text Classification Using TF-IDF (Term Frequency-Inverse Document Frequency) And LSTM (Long Short-Term Memory).' JUITA." *JurnalInformatika* 10, no. 2: 225–32.
46. Labrín, Caterina, and Francisco Urdinez. 2020. "Principal Component Analysis." In *R for Political Data Science*: 375–93. Chapman & Hall/CRC.
47. Jo, Han-Shin, Chanshin Park, Eunhyoung Lee, Haing Kun Choi, and Jaedon Park. 2020: 1927. "Path Loss Prediction Based on Machine Learning Techniques: Principal Component Analysis, Artificial Neural Network, and Gaussian Process." *Sensors* 20, no. 7. https://doi.org/10.3390/s20071927.
48. LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. "Deep Learning.". *Nature* 521, no. 7553: 436–44. https://doi.org/10.1038/nature14539.

49. Shen, Chaopeng. 2018. “A Transdisciplinary Review of Deep Learning Research and Its Relevance for Water Resources Scientists.” *Water Resources Research* 54, no. 11: 8558–93. https://doi.org/10.1029/2018WR022643.
50. Khaki, Saeed, Lizhi Wang, and Sotirios V. Archontoulis. 2019. “A Cnn-Rnn Framework for Crop Yield Prediction.” *Frontiers in Plant Science* 10: 1750. https://doi.org/10.3389/fpls.2019.01750.
51. Vachhani, Bhavik, Chitralekha Bhat, Biswajit Das, and Sunil Kumar Kopparapu. “Deep Autoencoder Based Speech Features for Improved Dysarthric Speech Recognition.” In *Interspeech*: 1854–8. 2017. https://doi.org/10.21437/Interspeech.2017-1318.