Researchers Journal of Science and Technology (REJOST)
https://rejost.com.ng

A Deep Learning Model for Classifying Candidate-Related Tweets in the 2023 Nigerian Presidential Election

S. A. Bawa[1]*, A. A Abubakar [2], S. Abdulkadir[1]., M. A Ahmad[2], A. A. Babajo[1]

Abstract

The increasing use of social media for political communication has transformed electoral discourse in Nigeria, particularly during the 2023 presidential election. While existing studies have largely focused on sentiment analysis or electoral prediction, limited attention has been given to candidate-level classification of political discourse as a distinct computational task. This study develops a deep learning framework for classifying tweets associated with the three leading presidential candidates Atiku Abubakar (PDP), Bola Ahmed Tinubu (APC), and Peter Obi (LP) based on their online presence. Twitter data were collected from six campaign-related hashtags between August 1, 2022 and February 28, 2023 to capture real-time political engagement. After preprocessing procedures including text cleaning, normalization, and duplicate removal, a total of 121,958 unique tweets were retained for model development. Two models were implemented: a baseline Artificial Neural Network (ANN) using TF-IDF vectorized features, and a Convolutional Neural Network (CNN) employing distributed word embeddings trained from scratch to capture localized campaign expressions and Nigerian political slang. Both models were evaluated using accuracy, precision, recall, F1-score, and AUC. The ANN achieved an accuracy of 0.5142, indicating limited effectiveness in distinguishing candidate-related tweet patterns. In contrast, the CNN achieved an accuracy of 0.9673 with balanced precision and recall, demonstrating superior performance in learning contextual and linguistic structures within short, informal political text. These findings confirm that CNN-based architectures are well suited for structuring large-scale candidate-related political discourse on social media. While this study does not infer voter preference or predict election outcomes, it provides a robust computational approach for

analyzing digital political communication in Nigeria. Future research may extend this framework using multilingual datasets, transformer-based language models, and cross-platform political discourse analysis.

✉ S.A. Bawa: saadatunas@gmail.com

[1] Department of Informatics, Kaduna State University, Nigeria

[2] Department of Secure Computing, Kaduna State University, Nigeria

Keywords: Deep Learning, Tweet Classification, Political Communication, Convolutional Neural Network, Social Media Analytics, Nigerian Election

Article history

Received: November 13, 2025
Received in revised form: January 16, 2026
Accepted for publication: January 22, 2026
Available online: January 28, 2026

1.0 Introduction

Social media has become a critical platform for political communication, civic engagement, and public opinion expression. In Nigeria, Twitter (now X) plays a prominent role in electoral discourse, enabling citizens to react to political events, circulate campaign messages, and engage in real-time discussions about candidates (Ikefuama, 2023). Increased mobile internet access and youth participation significantly intensified online political conversations during the 2023 Nigerian Presidential Election, as users actively produced content rather than merely consuming it. Candidate-related hashtags such as #Atiku, #Atikulate, #BAT, #Batified, #Obi, and #Obidatti became central to online political mobilization, allowing supporters to amplify messages and construct candidate identities. This reflects the growing influence of social media in political messaging and opinion formation (Alhaimer, 2023; Akande et al., 2023). However, the resulting volume of unstructured textual data poses substantial challenges for manual analysis and systematic interpretation, particularly when the objective extends beyond sentiment detection to understanding candidate-specific discourse structures. Advances in Artificial Intelligence (AI) and Deep Learning (DL) offer effective solutions for large-scale text analysis (Mbunge & Batani, 2023). In particular, Convolutional Neural Networks (CNNs) have demonstrated strong performance in Natural Language Processing tasks by automatically learning contextual and semantic features from text, without extensive manual feature engineering (Zhang et al., 2023). Compared to Artificial Neural Networks (ANNs), CNNs are especially well suited for short, informal texts such as tweets, where meaning is often conveyed through localized expressions, hashtags, and contextual word patterns (López et al., 2022). These characteristics are particularly relevant in Nigerian political Twitter discourse, which frequently includes campaign-specific slang, creative spellings, and informal linguistic constructions.

Existing Nigerian studies on the 2023 presidential election have largely emphasized sentiment analysis or electoral outcome inference. For example, Attai et al. (2024) applied machine learning models including SVM, Random Forest, and XGBoost for sentiment classification, while Olabanjo et al. (2023) employed deep learning and transformer-based models such as LSTM and BERT to analyze public sentiment on Twitter. While these studies demonstrate the analytical value of social media data, they offer limited

insight into candidate-level discourse classification as a standalone computational task, independent of sentiment polarity or vote prediction. As a result, there remains a methodological gap in structuring and distinguishing candidate-related political discourse based on linguistic and contextual patterns.

Against this background, the present study makes three key contributions. First, it addresses the identified gap by focusing on candidate-level tweet classification, rather than sentiment analysis or election prediction, during the 2023 Nigerian presidential election. Second, it applies a CNN-based contextual modeling approach to automatically learn linguistic patterns in Nigerian political Twitter discourse, and systematically compares its performance with a baseline ANN model. Third, the study employs distributed word embeddings trained from scratch to capture localized campaign expressions, Nigerian political slang, and informal linguistic variations that are not adequately represented in standard pretrained embeddings. By framing Twitter analysis as a tool for understanding digital political discourse rather than predicting electoral outcomes, this study contributes a robust computational framework for analyzing large-scale political communication in Nigeria.

2.0 Materials and Methods

2.1 Research Process

The study adopts an experimental research design, focusing on developing and evaluating deep learning models for tweet classification. A supervised learning framework was employed, where labeled datasets containing tweets associated with Atiku Abubakar, Bola Ahmed Tinubu, and Peter Obi were used to train and validate the models. The choice of supervised deep learning is informed by the need for automated extraction of contextual features from large volumes of unstructured social media data, which traditional machine learning models may struggle to capture effectively. The research process adopted in this study, detailing the stages of data collection, preprocessing, feature extraction, model development, evaluation, and comparative analysis, is presented in Figure 1.

2.2 Dataset Collection

The study used the Twitter data collected by Odegbile and Oyelami (2024). The dataset comprises tweets that were collected from six presidential candidate-related hashtags using the Twitter API between August 1, 2022 and February 28, 2023. Hashtags such as #Atiku**,** #Atikulate**,** #BAT**,** #Batified**,** #Obi**,** and #Obidatti were selected because they represent commonly-used identifiers for the three presidential candidates. These hashtags enabled the automated retrieval of tweets directly associated with each candidate. Metadata such as timestamps and tweet IDs were collected alongside the tweet text. No attempt was made to infer sentiment strength, political preference, or offline support, as the objective was strictly tweet categorization.

The hashtags listed in Table 1 were selected because they serve as unique identifiers associated with each candidate's online presence. These hashtags enabled the retrieval and labeling of tweets for supervised learning. The dataset was not collected to infer voter preference or electoral outcomes; rather, it provides structured input for training models to classify tweets into candidate-related categories based on linguistic and contextual patterns. The dataset has 364,827 tweets organised into six files.

2.3 Data Preprocessing

The raw tweets were subjected to a series of preprocessing steps to ensure data consistency and suitability for model training. These steps included:

1. Text Cleaning – Removal of URLs, emojis, user mentions, punctuation marks, and special characters.

2. Normalization – Conversion of all text to lowercase and removal of repeated characters and elongated expressions typical in informal communication.
3. Stop-word Removal – Elimination of frequent but semantically insignificant words to reduce noise.
4. Tokenization – Splitting of text into individual tokens for analysis.
5. Duplicate Removal – Deduplication of identical tweets to avoid model bias.

After preprocessing, 121,958 unique tweets remained, labeled based on their associated candidate hashtag. The CNN leveraged embedding-based representations to mitigate class imbalance effects, ensuring robustness across categories.

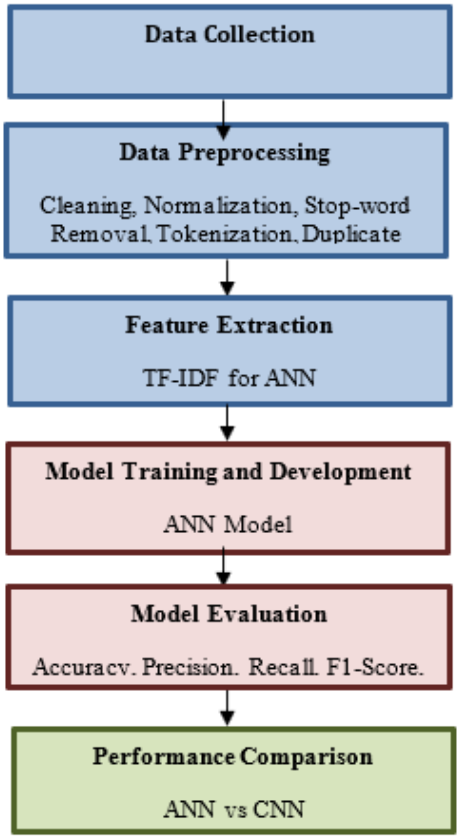

Fig 1: Research Process for the study

Table 1: Dataset Description for Candidate-Based Tweet Classification

Candidate	Party	Hashtags Used	Files Loaded	Tweets Collected
Atiku Abubakar	PDP	#Atiku, #Atikulate	atiku.csv, atikulate.csv	77,290
Bola Ahmed Tinubu	APC	#BAT, #Batified	bat.csv, batified.csv	112,810
Peter Obi	LP	#Obi, #Obidatti	obi.csv, obiDatti.csv	174,727
		Total		**364,827**

2.4 Feature Extraction

Two feature extraction strategies were adopted corresponding to the models used:

1. The Artificial Neural Network (ANN) utilized TF-IDF vectorized features, which numerically represent word importance within the documents.
2. The Convolutional Neural Network (CNN) used distributed word embeddings, enabling the model to learn contextual and semantic relationships from the text without manual feature engineering.

The embedding layer used a 100-dimensional space, trained from scratch to capture localized campaign expressions and Nigerian political slang.

2.5 Model Development

Two models were implemented to achieve the study's objectives**:** a baseline ANN and an advanced CNN.

2.5.1 Baseline Model: Artificial Neural Network (ANN)

The ANN model replicates the approach used in the benchmark study (Ajayi & Akinrolabu, 2023) and serves as a comparative baseline illustrated in Figure 2. This ANN is a shallow neural network relying on handcrafted TF-IDF features rather than contextual embeddings.

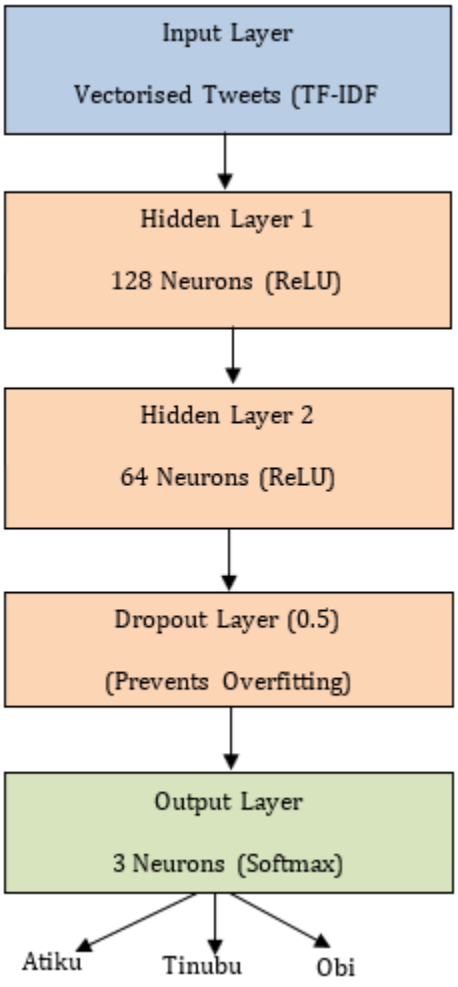

Fig 2: Baseline ANN Model Architecture

2.5.2 Deep Learning Model: Convolutional Neural Network (CNN)

A multi-layer architecture comprising embedding, convolutional, pooling, and fully connected layers. The CNN was designed to automatically extract spatial patterns such as campaign slogans, nicknames, and recurring lexical identifiers used to reference each candidate. Unlike the ANN, the CNN requires no manual feature engineering, making it more suitable for informal and context-rich text. Figure 3 shows the Convolutional Neural Network (CNN) architecture used in this study for the classification of candidate-related tweets.

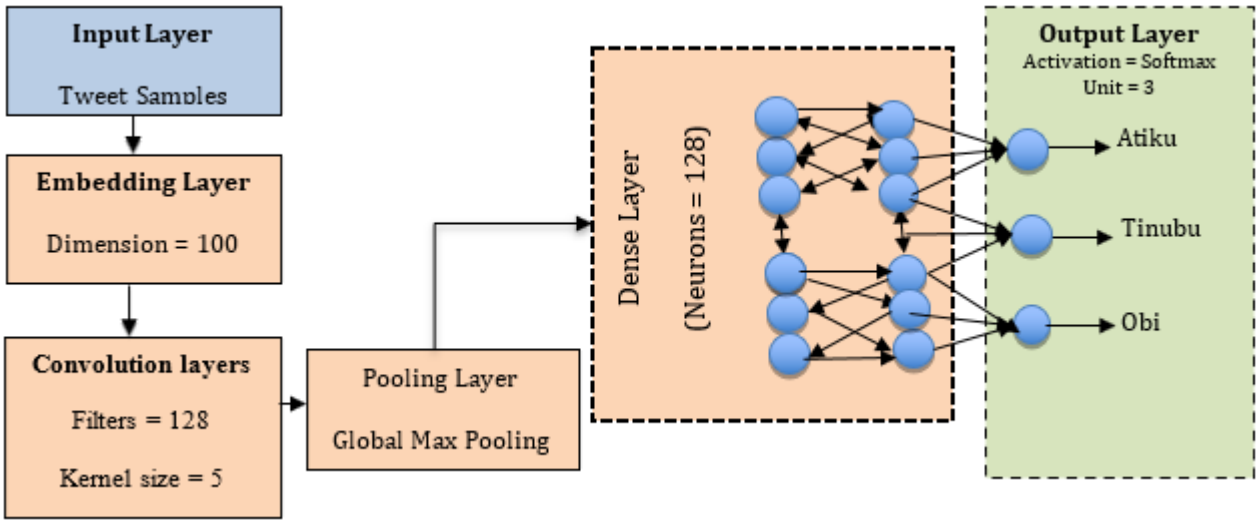

Fig 3: CNN Architectural Design

This CNN architecture captures word patterns that signify sentiment or preference (e.g., "vote for Obi", "support Atiku"), providing higher contextual understanding compared to ANN.

2.6 Model Training and Evaluation

The ANN and CNN models using the pre-processed dataset of 121,958 labeled tweets. The data were split into training, validation, and testing sets using an 80:10:10 ratio to ensure fair and consistent evaluation. This resulted into 97,566 tweets for training and 24,392 for validation and testing. The ANN received TF-IDF vector inputs, while the CNN operated on distributed embedding vectors. Identical hyperparameters, which include batch size, learning rate, and number of epochs, were applied where suitable, ensuring the comparison reflects architectural differences rather than tuning variations. The class distribution after preprocessing is shown in Table 2. No resampling techniques were applied; the CNN mitigates class imbalance through embedding-based feature learning.

Hyperparameters: Key hyperparameters for the ANN and CNN models, such as learning rate, batch size, epochs, optimizer, and activation functions, were selected to ensure effective model training and robust performance. Table 3 summarizes these configurations.

Table **2: Class Distribution after Preprocessing**

Candidate	Tweets	Percentage
Atiku Abubakar	25,678	21%
Bola Ahmed Tinubu	36,145	30%
Peter Obi	60,135	49%

Table 3: Hyperparameters

Hyperparameter	ANN	CNN
Learning rate	0.001	0.001
Batch size	32	64
Epochs	20	25
Optimizer	Adam	Adam
Activation	ReLU (hidden), Softmax (output)	ReLU (conv/dense), Softmax (output)
Loss function	Categorical Cross-Entropy	Categorical Cross-Entropy

Hardware Environment

The implementation of this study was conducted using Python 3 on the Google Collaboratory platform, which provides an efficient and accessible cloud-based environment. The experiments were executed on a CPU-based Google Compute Engine instance with 13 GB of RAM and a disk size of 108 GB. This hardware configuration ensured that the computational requirements of the model development and evaluation processes were adequately met, enabling efficient processing of the dataset and timely execution of experiments.

The following performance metrics were used to evaluate the performance of the two model:

1. Confusion matrix: This is a table used to measure how well a classification model is performing by comparing predicted values against actual values for a dataset as shown in Table 4.

Table 4: Confusion Matrix

		Predicted
Actual	True Positive (TP)	False Negative (FN)
	False Positive (FP)	True Negative (TN)

True Positive (TP) is when a model correctly classifies candidate tweets as positive. False Positive (FP) is when a model incorrectly classifies candidate tweets as positive case when they are actually negative. False Negative (FN) is when a model incorrectly classifies candidate tweets as a negative case when they are actually positive. True Negative (TN) is when a model correctly classifies candidate tweets as negative case.

2. Accuracy: The proportion of correctly classified tweets.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

3. Precision: The fraction of correctly classified candidate tweets among all predicted for that candidate.

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

4. Recall (Sensitivity): The fraction of correctly classified candidate tweets among all actual tweets for that candidate.

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

5. F1-Score: The harmonic mean of Precision and Recall.

$$F1 = 2 \times \frac{Precision\ .Recal}{TPrecision + Recal} \quad (4)$$

6. AUC (Area Under Curve): Measures model discriminative power.

These metrics were selected to assess the ability of the models to correctly classify tweets into their respective candidate categories.

2.7 Tools and Frameworks

All preprocessing, modeling, and evaluation were implemented using Python programming language and the following libraries as shown in Table 5. Python was selected for its versatility and rich ecosystem of machine learning and deep learning libraries. All experiments were implemented on Google Colab.

2.8 Ethical Considerations

All tweets were collected from publicly available data, with no attempt to identify or track individual users. No personal information was used, and all analyses were conducted solely on the text content of tweets. Ethical guidelines regarding social media research and privacy were adhered to.

Table 5: Tasks, Tools and Frameworks used in the Study

Task	Tools/Frameworks
Data Manipulation	Pandas, NumPy
Text Cleaning & NLP	NLTK, re, WordCloud
Feature Extraction	TF-IDF (Scikit-learn), Keras Tokenizer
Modeling	TensorFlow/Keras
Evaluation	Scikit-learn (metrics), Matplotlib, Seaborn
Development Environment	Google Colab / Jupyter Notebook

3.0 Results

3.1 Classification Models

The ANN and CNN models were implemented using Python programming languages. The ANN implementation is based on the architecture adapted from Ajayi and Akinrolabu (2023) as discussed in Section 2.5.1. The source code of the ANN implementation is presented in Appendix A. Furthermore, the CNN implementation is based on the designed presented in Section 2.5.2. The source code of the CNN implementation is presented in Appendix B. The development of the two models for tweets classification achieves objective 1 and objective 2 of this research, which sought to implement an ANN Model for tweet classification and develop a CNN model for improved tweet classification, respectively.

3.2 Performance of the Baseline ANN Model

Figure 4 illustrates the results of confusion matrix for the ANN model. The confusion matrix shows that the ANN model frequently misclassified tweets into the Tinubu class, indicating sensitivity to class imbalance. This demonstrates the model's limited feature extraction capability and its inability to capture nuanced contextual dependencies within short-form political text. Furthermore, Table 4.1 displays the classification performance of the ANN model based on Accuracy, Precision, Recall, F1-Score and AUC. The ANN model achieved an overall accuracy of 0.5142, with precision of 0.1714, recall of 0.3333, F1-score of 0.2264, and AUC value of 0.5221. While the model successfully identified certain frequently occurring words and hashtags, its performance was limited by its shallow architecture and handcrafted TF-IDF features. The ANN struggled to capture the contextual dependencies and semantic nuances embedded in short-form political tweets especially those containing creative expressions, slang, campaign slogans, or implicit references. These limitations

resulted in lower precision and recall scores across all candidate classes, indicating that the ANN was unable to robustly discriminate between tweet categories in a highly dynamic linguistic environment.

The accuracy result of ANN is consistent with Ajayi and Akinrolabu (2023), who reported 67% accuracy in manifesto analysis, which is likely constrained by shallow feature extraction and limited contextual understanding. This outcome supports the argument that more sophisticated deep learning architectures, such as CNNs, are better suited for robust classification of informal political text like tweets.

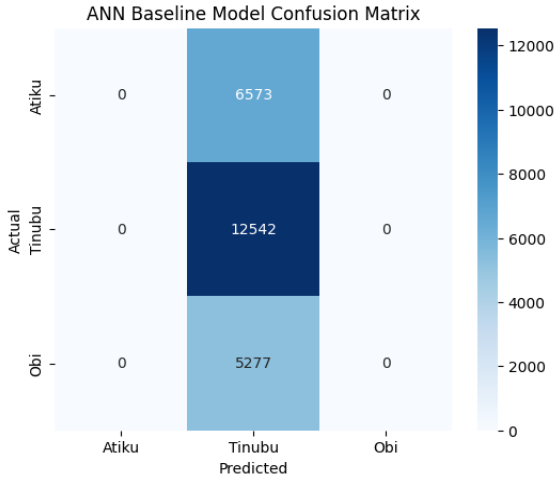

Fig 4: ANN confusion matrix showing candidate tweet classification results

Table 4: ANN Baseline Model Performance

Metric	Value
Accuracy	0.5142
Precision (macro)	0.1714
Recall (macro)	0.3333
F1-score (macro)	0.2264
AUC	0.5221

3.3 ROC Curve Analysis

The ANN ROC curve illustrated in Figure 5 achieved an AUC of 0.5221, indicating limited discriminatory ability across the three candidate classes.

3.3 Performance of the CNN Model

Figure 6 illustrates the results of confusion matrix for the CNN model. The confusion matrix indicates that the CNN classified tweets for each candidate with minimal misclassification. Tweets associated with Tinubu achieved the highest correct classification of 12305 tweets (≈98%), followed closely by Atiku with 6298 correct classification, then Obi with 4991 correct classifications. The CNN model misclassified more than 200 misclassified tweets for each candidate with Obi having the highest number (183+103 = 286) of misclassification, followed by Atiku (216+59 = 275) and Tinubu (81+156 = 237). Similar to the ANN model, Table 5 displays the classification performance of the CNN model based on Accuracy, Precision, Recall, F1-Score and AUC.

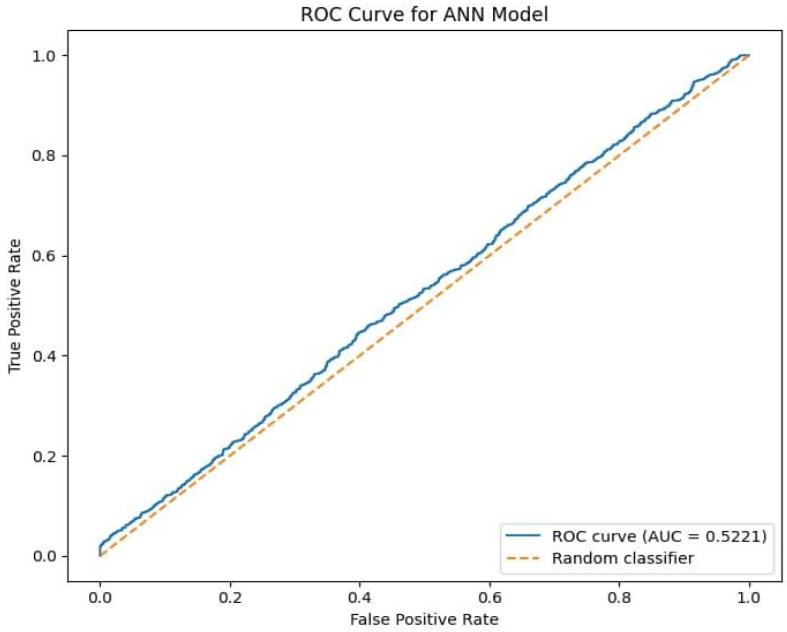

Figure 5: ANN ROC Curve

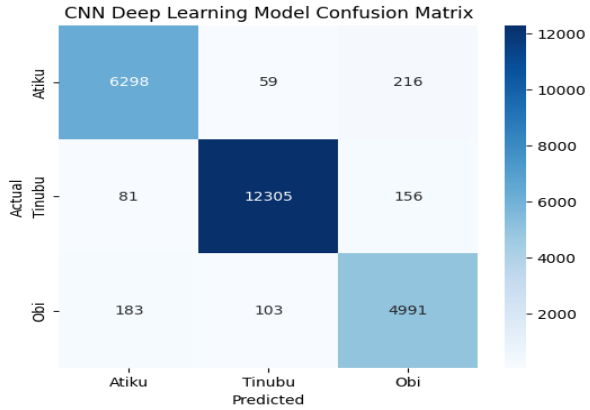

Figure 6: CNN confusion matrix showing candidate tweet classification results

Table 5: CNN Model Performance

Metric	Value
Accuracy	0.9673
Precision (macro)	0.9591
Recall (macro)	0.9617
F1-score (macro)	0.9604
AUC	0.9970

The CNN achieved an accuracy of 0.9673, precision of 0.9591, recall of 0.9617, F1-score of 0.9604, and an AUC of 0.9970, demonstrating its superior capacity to classify tweets according to candidate association. Unlike the ANN, the CNN leverages convolutional filters to extract n-gram features and contextual patterns, enabling it to recognize variations in naming conventions, hashtags, and rhetorical structures. This capability resulted in balanced precision and recall values,

indicating that the model did not merely fit dominant patterns but generalized well across candidate-related discourse. The CNN's higher AUC score further confirms its effectiveness in distinguishing between the three tweet classes.

These scores demonstrate the model's superior classification performance, indicating that it effectively learned the linguistic and contextual patterns present in tweets associated with each candidate. The model's near-perfect AUC further confirms its strong discriminative ability across all classes. These results highlight the CNN's efficiency in handling text-based classification tasks, where spatial relationships such as word positioning, recurring lexical patterns, and contextual cues assist the model in distinguishing between candidate-related discourse.

3.4 Model Performance Comparison

Table 6 presents a side-by-side comparison of both models, which Figure 6 illustrates the performance difference between the ANN and CNN models. The CNN model outperformed the ANN model by 45.3% in accuracy, 73.4% in F1-score and 47.5% in AUC (0.997 vs. 0.522). These results demonstrate the CNN's enhanced ability to recognize contextual and structural patterns present in candidate-related tweets, enabling more reliable classification performance than the ANN.

3.5 CNN ROC Curve Analysis

The CNN model illustrated in Figure 7 demonstrated a substantially higher AUC of 0.99701, reflecting superior classification performance. These results suggest that the CNN more effectively captures contextual and semantic patterns in the tweets compared to the ANN.

Table 6: Comparative Model Performance

Metric	ANN (Baseline)	CNN Model	Difference
Accuracy	0.5142	0.9673	0.4531
Precision (macro)	0.1714	0.9591	0.7877
Recall (macro)	0.3333	0.9617	0.6284
F1-score (macro)	0.2264	0.9604	0.7340
AUC	0.5221	0.9970	0.4749

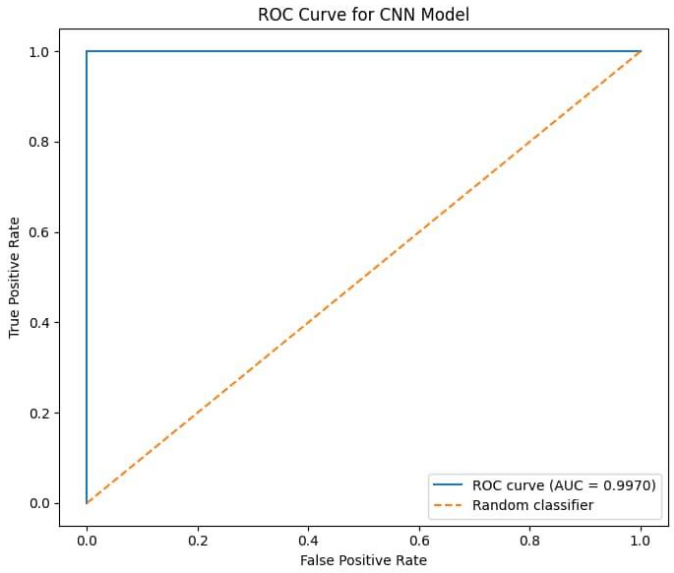

Figure 7: CNN ROC-Curve

From the model comparison chart presented in Figure 8, the results clearly show that the CNN model offers substantially stronger classification performance than the ANN. The performance gap reflects the CNN's ability to automatically learn hierarchical and semantic patterns in text such as recurring campaign hashtags, stylistic expressions, and candidate-specific lexical cues without the need for manual feature engineering. This strengthened feature representation enables the CNN to reliably distinguish between tweet categories associated with different candidates. It is important to note that this superior performance relates solely to the classification of online discourse and does not imply political preference, sentiment direction, or real-world electoral outcomes.

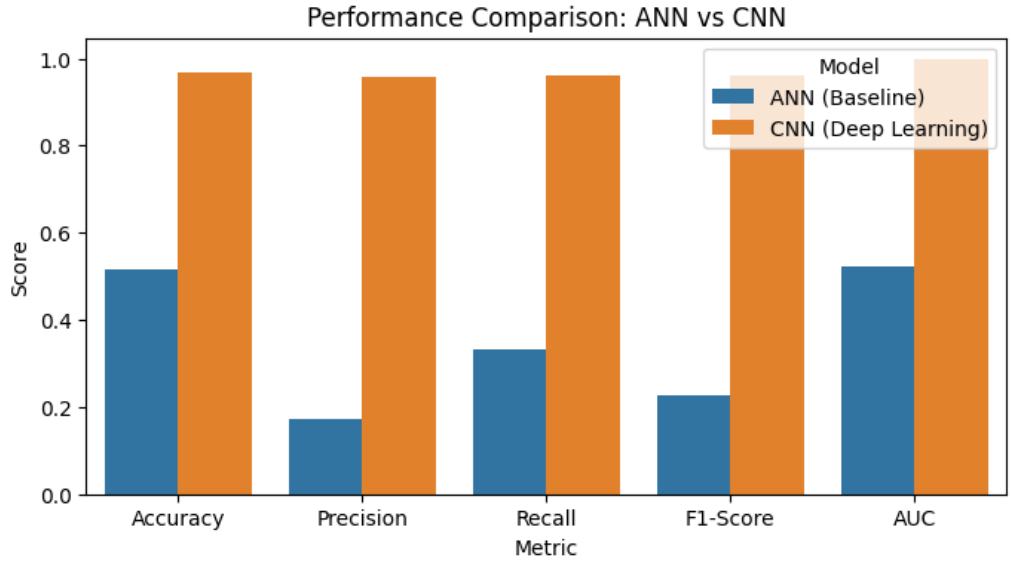

Figure 8: Model Performance Comparison Chart

Thus, these results provide answers to the study's research questions which sought to know the performance of ANN model in classifying candidate-related tweets, and how the CNN model can improve classification accuracy compared to the ANN model.

4.0 Discussion

The experimental results demonstrate that the CNN model substantially outperforms the ANN baseline in classifying tweets associated with the three presidential candidates. The ANN model, which relies on TF-IDF representations, treats text as a collection of independent terms and therefore fails to preserve word order and contextual dependencies. This limitation reduces its ability to distinguish candidate-specific linguistic patterns in informal political discourse, resulting in modest classification performance and an AUC value of 0.5221, which is only marginally better than random classification.

In contrast, the superior performance of the CNN model is attributable to its hierarchical feature extraction and contextual encoding capabilities. By leveraging distributed word embeddings trained from scratch and convolutional filters, the CNN effectively learns localized semantic patterns, campaign-specific slogans, hashtags, and recurring lexical markers embedded in Nigerian political tweets. This architectural advantage is reflected not only in the high accuracy, precision, recall, and F1-score values, but also in the ROC–AUC score of 0.9970, indicating an excellent discriminative ability across all candidate classes. The ROC analysis confirms that the CNN reliably

separates candidate-related discourse regardless of classification threshold, whereas the ANN exhibits limited separability.

Importantly, the CNN's strong performance should not be interpreted as evidence of candidate popularity, voter preference, or electoral viability. Rather, the model captures the degree to which linguistic markers associated with each candidate are consistent and learnable within online discourse. Variations in spelling, slogan coherence, hashtag usage, and rhetorical style may make certain candidate-related tweets easier to classify than others without implying any offline political significance. The task addressed in this study is therefore one of linguistic separability, not political endorsement or sentiment direction.

Although the dataset exhibited an uneven distribution of tweets across candidates, explicit resampling techniques such as oversampling or SMOTE were not applied. Instead, robustness to class imbalance was addressed through embedding-based representations and the use of macro-averaged evaluation metrics, which assign equal importance to all classes regardless of their frequency. The balanced precision, recall, and F1-score values across candidate classes suggest that the CNN generalized effectively despite the imbalance, supporting the methodological choices adopted in this study. These findings are consistent with prior research indicating that deep learning models using dense embeddings are less sensitive to class dominance compared to frequency-based classifiers (Zhang et al., 2023; Lopez et al., 2022).

Overall, the findings reinforce the suitability of CNN-based architectures for analyzing political text in social media environments, particularly where discourse is informal, expressive, and context-dependent, as observed in Nigerian political Twitter discussions. By structuring large-scale candidate-related discourse without inferring sentiment or electoral outcomes, the study contributes to computational political communication research by demonstrating how deep learning models can be used to organize and analyze digital political narratives.

4.1 Limitations and Error Analysis

Despite the strong performance of the CNN model, several limitations are acknowledged. First, the study relies exclusively on Twitter data, which does not represent the broader Nigerian population and may over-represent politically active or urban users. Second, the classification task is limited to English and undetermined (Pidgin-tagged) tweets, excluding indigenous languages such as Hausa, Yoruba, and Igbo that are widely used in Nigerian political discourse. Third, while embeddings were trained from scratch to capture localized language patterns, this approach may limit transferability to other electoral contexts without retraining. Finally, misclassifications observed in the confusion matrices suggest that tweets containing sarcasm, mixed candidate references, or ambiguous hashtags remain challenging, highlighting the need for future work incorporating discourse-level or transformer-based models.

5.0 Conclusion

This study demonstrates that deep learning models, particularly CNN-based architectures, are highly effective for classifying candidate-related tweets in the context of Nigeria's 2023 presidential election. By leveraging distributed word embeddings and convolutional filters, the CNN model captures contextual and linguistic features embedded within informal political discourse, significantly outperforming a TF-IDF-based ANN baseline. While the study does not infer voter preference or electoral outcomes, it provides a robust computational framework for structuring large-scale political text data and

contributes to the growing body of research in digital political communication.

Declarations

Ethical Approval and Consent to Participate
Not Applicable

Consent for Publication
Not Applicable

Availability of Data and Materials
The Nigeria's 2023 Presidential Election Dataset", Mendeley Data, V2 analyzed during the current study is available at https://10.17632/whb-6rychpx.2

Competing of interest
The authors declare that they have no competing interests, regarding the research, authorship or publication of this manuscript.

Funding
No funding was received for this study

Author's Contribution
S.A., served as the student, conducted the data collection, preprocessing, model development, analysis, and interpretation of the results. A.A. served as the major supervisor, providing overall guidance, M.A. provided conceptual input, and critical revisions of the study. S., acted as the second supervisor, offering additional supervision, reviewing the methodology, critical revisions and making editorial corrections. All authors read and approved the final manuscript.

Acknowledgements
The authors wish to thank the Departments of Informatics and Secure Computing, Faculty of Computing, Kaduna State University, for providing computing resources needed for this research.

References

Ajayi, O. O., & Akinrolabu, O. D. (2023). Machine perception of political manifestos in predicting performance of public office holders. *JPPUMA JurnalIlmu Pemerintahan Dan Sosial Politik Universitas Medan Area*, *11*(2), 138–148. https://doi.org/10.31289-/jppuma.v11i2.10716.

Akande, P. A., Adeleke, O. F., Ajao, R. A., & Amusa, A. K. (2023). A Semantic Analysis of Selected Peter Obi's Political Campaign Tweets and Readers' Comments. *International Journal of Social Sciences and Management Review*, *06*(06), 197–216. https://doi.org/10.3-7602/ijssmr.2023.6615

Alhaimer, R. S. (2023). Unveiling the digital persona image: the influence of social media on political candidates' brand personality and voter behaviour in Kuwait. *Humanities and Social Sciences Communications*, *10*(1). https://doi.org/-10.1057/s41599-023-02420-4

Attai, K., Asuquo, D., Okonny, K. E., Johnson, E. A., Bassey, A., John, A., Bardi, I., Iroanwusi, C., & Michael, O. (2024). Sentiment analysis of Twitter discourse on the 2023 Nigerian general elections. *European Journal of Computer Science and Information Technology*, *12*(4), 18–35. https://doi.org/10.37745/ejcsit.2013/-vol12n41835

Ikefuama, A. E. (2023). Social media and political participation in Nigeria: examining the use of Twitter in election campaigning. *dspace.ub.uni-siegen.de*. https://doi.org/1-0.25819/ubsi/10434

López, O. a. M., López, A. M., & Crossa, J. (2022). Fundamentals of artificial neural networks and deep learning. In *Springer eBooks* (pp. 379–425). https://doi.org/10.-1007/978-3-030-89010-0_10

Mbunge, E., & Batani, J. (2023). Application of deep learning and machine learning models to improve healthcare in sub-Saharan Africa: Emerging opportunities, trends and implications. *Telematics and Informatics Reports*, *11*, 100097. https:-//doi.org/10.1016/j.teler.2023.100097.

Odegbile, A., & Oyelami, O. (2024). "Nigeria's 2023 Presidential Election Dataset", Mendeley Data, V2. https://10.17632/-whb6rychpx.2

Olabanjo, O., Wusu, A., Afisi, O., Asokere, M., Padonu, R., Olabanjo, O., Ojo, O., Folorunso, O., Aribisala, B., & Mazzara, M. (2023). From Twitter to Aso-Rock: A sentiment analysis framework for understanding Nigeria 2023 presidential election. *Heliyon*, *9*(5), e16085. https://-doi.org/10.1016/j.heliyon.2023.e16085

Zhang, Z., Chen, K., Wang, R., Utiyama, M., Sumita, E., Li, Z., & Zhao, H. (2023). Universal multimodal representation for language understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(7), 1–18. https-://doi.org/10.1109/tpami.2023.3234170

Appendix A: Code Implementation for Dataset Loading and Preprocessing

```
# Import Required Libraries
import pandas as pd
import numpy as np
import re
import nltk
import string
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from sklearn.model_selection import
train_test_split

# Download NLTK resources
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('omw-1.4')

# Define Dataset Paths
atiku_path =
'/content/drive/MyDrive/ElectionPrediction/Raw
Data/atiku.csv'
atikulate_path =
'/content/drive/MyDrive/ElectionPrediction/Raw
Data/atikulate.csv'
bat_path =
'/content/drive/MyDrive/ElectionPrediction/Raw
Data/bat.csv'
batified_path =
'/content/drive/MyDrive/ElectionPrediction/Raw
Data/batified.csv'
obi_path =
'/content/drive/MyDrive/ElectionPrediction/Raw
Data/obi.csv'
obidatti_path =
'/content/drive/MyDrive/ElectionPrediction/Raw
Data/obiDatti.csv'
# Load the Datasets
atiku = pd.read_csv(atiku_path)
atikulate = pd.read_csv(atikulate_path)
bat = pd.read_csv(bat_path)
batified = pd.read_csv(batified_path)
obi = pd.read_csv(obi_path)
obidatti = pd.read_csv(obidatti_path)
# Merge Hashtag Files by Candidate
atiku_df = pd.concat([atiku, atikulate],
ignore_index=True)
```

```
bat_df = pd.concat([bat, batified],
ignore_index=True)
obi_df = pd.concat([obi, obidatti],
ignore_index=True)
# Add Candidate Labels
atiku_df['label'] = 'Atiku'
bat_df['label'] = 'Tinubu'
obi_df['label'] = 'Obi'
# Combine All into One Dataset
df = pd.concat([atiku_df, bat_df, obi_df],
ignore_index=True)
print("Total Tweets Before Cleaning:", len(df))
# Keep Only Needed Columns
df = df[['text', 'lang', 'retweets', 'num_replies',
'quotes', 'label']]
# Remove Missing or Empty Texts
df.dropna(subset=['text'], inplace=True)
df = df[df['text'].str.strip() != ""]
# Remove Duplicates
df.drop_duplicates(subset=['text'], inplace=True)
print("Total Tweets After Removing
Duplicates:", len(df))
# Filter English or Undetermined (Pidgin) Tweets
df = df[df['lang'].isin(['en', 'und'])]
# Clean Text Function
stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

def clean_text(text):
    text = text.lower()
    text = re.sub(r'http\S+|www\S+|https\S+', ",
text)  # Remove URLs
    text = re.sub(r'@\w+|#', ", text) # Remove
mentions and hashtags
    text = re.sub(r'[^\w\s]', ", text)              #
Remove punctuation
    text = re.sub(r'\d+', ", text)                  #
Remove digits
    tokens = text.split()
    tokens = [lemmatizer.lemmatize(word) for
word in tokens if word not in stop_words]
    return ' '.join(tokens)
# Apply Cleaning
df['clean_text'] = df['text'].apply(clean_text)
print("Total Tweets After Text Cleaning:",
len(df))
# Encode Labels Numerically
label_mapping = {'Atiku': 0, 'Tinubu': 1, 'Obi': 2}
df['label_encoded'] =
df['label'].map(label_mapping)
# Split into Train and Test Sets (80/20)
X_train, X_test, y_train, y_test = train_test_split(
df['clean_text'], df['label_encoded'],
test_size=0.2, random_state=42,
stratify=df['label_encoded']
)
# Save Clean Data (Optional)
df.to_csv('/content/drive/MyDrive/ElectionPredic
tion/Processed_Data/Cleaned_Twitter_Election_
Data.csv', index=False)
print("\n✅ Data Preprocessing Complete!")
print(f"Training Samples: {len(X_train)}")
print(f"Testing Samples: {len(X_test)}")
print("\nSample Cleaned Text:\n",
df['clean_text'].head(5))
```

Appendix B: Code Implementation for Model Training

```
import tensorflow as tf
from tensorflow.keras.preprocessing.text import
Tokenizer
from tensorflow.keras.preprocessing.sequence
import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,
Dropout, Embedding, Conv1D,
GlobalMaxPooling1D, Flatten
from tensorflow.keras.utils import to_categorical
from sklearn.metrics import accuracy_score,
precision_score, recall_score, f1_score,
confusion_matrix, roc_auc_score,
classification_report
import matplotlib.pyplot as plt
import seaborn as sns
```

```
## Tokenization and Sequence Preparation
vocab_size = 10000
max_length = 50
tokenizer = Tokenizer(num_words=vocab_size,
oov_token="<OOV>")
tokenizer.fit_on_texts(X_train)

X_train_seq =
tokenizer.texts_to_sequences(X_train)
X_test_seq =
tokenizer.texts_to_sequences(X_test)
X_train_pad = pad_sequences(X_train_seq,
maxlen=max_length, padding='post',
truncating='post')
X_test_pad = pad_sequences(X_test_seq,
maxlen=max_length, padding='post',
truncating='post')

## One-Hot Encode Labels
y_train_cat = to_categorical(y_train,
num_classes=3)
y_test_cat = to_categorical(y_test,
num_classes=3)
print("✅ Text Tokenization and Padding
Complete")
print("Vocabulary Size:",
len(tokenizer.word_index))
print("Training Shape:", X_train_pad.shape)
## BBASELINE MODEL: ANN
ann_model = Sequential([
Dense(128, activation='relu',
input_shape=(X_train_pad.shape[1],)),
Dropout(0.5),
Dense(64, activation='relu'),
Dropout(0.3),
Dense(3, activation='softmax')
])
ann_model.compile(optimizer='adam',
loss='categorical_crossentropy',
metrics=['accuracy'])
# ANN expects flattened features (we use TF-
IDF-like input)
X_train_flat =
X_train_pad.reshape(X_train_pad.shape[0], -1)
X_test_flat =
X_test_pad.reshape(X_test_pad.shape[0], -1)
print("\n🚀 Training ANN Baseline Model...")
history_ann = ann_model.fit(X_train_flat,
y_train_cat, epochs=10, batch_size=32,
validation_split=0.2, verbose=1)
ann_model.summary()
## DEEP MODEL: CNN
cnn_model = Sequential([
Embedding(vocab_size, 100,
input_length=max_length),
   Conv1D(128, 5, activation='relu'),
   GlobalMaxPooling1D(),
Dense(128, activation='relu'),
Dropout(0.5),
Dense(3, activation='softmax')
])
cnn_model.compile(optimizer='adam',
loss='categorical_crossentropy',
metrics=['accuracy'])
print("\n🚀 Training CNN Deep Learning
Model...")
history_cnn = cnn_model.fit(X_train_pad,
y_train_cat, epochs=10, batch_size=64,
validation_split=0.2, verbose=1)
cnn_model.summary()
```