INTERNATIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT (IJSREM)
VOLUME: 08 ISSUE: 04 | APRIL - 2024 SJIF RATING: 8.448 ISSN: 2582-3930

Deepfake on Social Media: Harnessing Deep Learning for Identifying Falsified Tweet

Jatinder Kaur[1]
School of Computer Science and Engineering
Lovely Professional University
Punjab, India
Jatinder.29755@lpu.co.in

Palakurthy Sai Vinay[1]
School of Computer Science and Engineering
Lovely Professional University
Punjab, India
saivinay1209@gmail.com

Gandlapati Sunil Kumar Reddy[1]
School of Computer Science and Engineering
Lovely Professional University
Punjab, India
gandlapatisunilreddy@gmail.com

Paripelli Venkata Shiva Sai[1]
School of Computer Science and Engineering
Lovely Professional University
Punjab, India
shivasaii788@gmail.com

Vasipalli Madhan Mohan Reddy[2]
School of Computer Science Engineering
Lovely Professional University
Punjab, India
madhanmohanreddy@gmail.com

***Abstract*: Recent advancements in natural language production have enabled the creation of sophisticated deepfake social media messages, posing a significant threat to public discourse. In response, this study focuses on the development of reliable detection methods for identifying automated text on websites such as Twitter. Leveraging Tweepfake, a publicly accessible dataset, a simple deep learning model employing tf-tf-idf, word2vec and tokenizer from keras library word embeddings and a typical architecture for a convolutional neural network (CNN) is proposed. Comparative analysis against baseline methods, including various feature-based approaches and various forms of deep learning, such Long Short-Term Memory (LSTM), demonstrates the suggested method's greater performance. Experimental results showcase an impressive accuracy of 91% in accurately classifying tweet data either bot-generated or human-generated. This research contributes to the ongoing efforts to combat the proliferation of deepfake content on social media platforms.**

Keywords: Deep Learning, Machine Learning, Deepfake Detection, Text Classification

I. INTRODUCTION

Social media networks have Unnaturally converted the way people communicate, allowing individualizes to partake their studies, opinions, and multimedia content with unknown ease [1]. Still, this openness has also paved the way for the proliferation of automated accounts, generally known as bots, which can circulate both authentic and manipulated content [2]. Of particular concern is the rise of deepfake technology, a confluence of artificial intelligence and multimedia manipulation, which has the implicit to deceive druggies by generating realistic yet entirely fabricated content [3].

The manipulation of social media content, including the spread of deepfake material, poses significant challenges to public converse and popular processes. By exploiting the immediate and far-reaching nature of social media platforms, vicious actors can propagate false information to manipulate public opinion and sow distrust [4]. This manipulation is eased by a diapason of automated accounts, ranging from subtly human-like cyborg accounts to completely automated social bots [5].

Recent advancements in natural language processing, exemplified by models like GPT and Grover, have further empowered adversaries to create convincing deepfake content [6], [7]. These models, distinguished by their capacity to produce language that is both logical and pertinent to the situation, have already been used to produce deceptive social media posts and comments [8]. Despite the absence of widespread harm thus far, the potential misuse of such technology raises pressing concerns regarding the integrity of online discourse and the spread of misinformation [9]. The emergence of powerful generative models like GPT-2 has exacerbated these concerns, as evidenced by their ability to produce text that is indistinguishable from human-authored content [10]. Detecting machine-generated text, particularly that produced by sophisticated models like GPT-2, presents a formidable challenge for existing detection methods [11]. While various approaches have been proposed for identifying deepfake content in multimedia formats, such as videos and images [12], [13], the detection of deepfake text remains relatively underexplored.

Existing detection strategies often rely on statistical features and machine learning algorithms, which struggle to effectively discriminate between writing produced by a machine and text written by a human [14]. Moreover, the prevalence of short-form content on social media platforms like Twitter further complicates the detection task, as traditional methods are better suited to longer-form text [15]. Additionally, the lack of adequately labeled datasets containing actual deepfake social media messages poses a significant obstacle to the development and evaluation of detection techniques [16].

In response to these challenges, this study aims to investigate novel approaches for detecting deepfake text on social networking websites. Leveraging a dataset of tweets produced by machines and by humans, we evaluate a range of tweet classification using machine learning and deep learning algorithms. Furthermore, we explore several methods for feature extraction tailored to the unique characteristics of short-form

social media content. By addressing these challenges, we seek to advance the development of robust detection methods capable of safeguarding the integrity of online discourse.

The rest of this paper is structured as follows: A survey of the literature on deepfake text identification is given in Section II, and in Section III discusses proposed frame. Section IV discussion of the results, then a summary of the findings in Section V. In the end, Section VI brings the paper to a close and provides insight for further dissection trials.

II. LITERATURE SURVEY

Deepfake technologies have evolved across various domains, initially emerging in computer vision [18–20], and subsequently extending to audio manipulation [21, 22] and text synthesis [24]. Deepfakes in computer vision typically include facial manipulation, which includes identity swapping, emotion switching, whole-facial synthesis, and attribute manipulation, alongside body reenactment [22, 23]. Recent advancements in audio deepfakes have enabled the cohort of spoken audio derived from text corpora, leveraging the voices of multiple speakers [21].

The advent of the transformer and self-attention mechanism architectures in 2017 revolutionized language modelling, enabling the development of transformative language models like GPT [25], BERT [26], and GPT-2 [23]. These models not only excel in language generation tasks but also in natural language understanding. For instance, GPT-2, introduced by Radford et al. [23] in 2019, demonstrated the capacity to autonomously produce logical, anthropomorphic text paragraphs from minimal input. Concurrently, GROVER [9] and CTRL [17] provided novel methodologies for learning and generating multi-field documents with specific styles and content. OPTIMUS [27] further enhanced text generation capabilities by incorporating variational autoencoders. Several methods for detecting deepfake text automatically have emerged, broadly categorized as follows:

- Simple Classifier: These include Binary classifiers created from scratch using deep learning or machine learning techniques.
- Detection using zero-shot: uses pre-trained language models' output as features for later classifiers.
- Detection based on fine-tuning: Involves using a basic neural network to fine-tune language models that have already been trained.

The detection of deepfake texts often employs various techniques and tools. For instance, the GLTR tool [28] aids in spotting deepfake texts by displaying statistical linguistic distinctions writing produced by a machine and text written by a human samples. GROVER's approach [9] focused on fine-tuning-based detection, utilizing trained language models beforehand, like BERT, GPT2, and GROVER itself.

However, recent studies have highlighted limitations in existing detection methods. For instance, [18] conducted an internal study on GPT-2 generated text samples, revealing discrepancies in accuracy between detectors based on different architectures. Moreover, deepfake text detection methods have primarily focused on news articles, which are lengthier than social media messages, raising concerns about generalizability. Additionally, reliance on single generative models like GPT-2 or GROVER limits the understanding of actual situations.

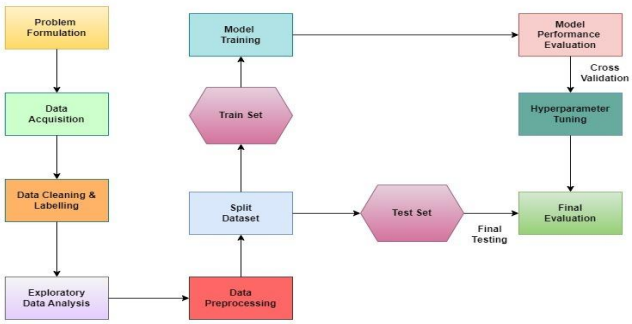
Our Tweepfake dataset seeks to address these gaps by providing a diverse collection of tweets generated by multiple generative models. This dataset facilitates research in identifying shallower deepfake texts using various generative methods, thereby contributing to advancements in deepfake detection methodologies.

I. TABLE : Comparing existing methods for analysis.

Year	Methods	Dataset	Findings
2018	LSTM	Cresci and collaborator dataset	They used data to build a smart bot detector with 96% accuracy [33].
2020	BERT-based detector	English tweets from the PAN competition dataset	The authors devised a bot detection model, achieving 83.36% weighted F1-score [34].
2021	RoBERTa based detector	Tweepfake dataset	Authors separated human and bot text, shared deepfake tweet dataset, and tested 13 detection models [35].
2022	GANBOT framework	Twitter social bot	Their model beat old LSTM methods in bot detection [36].
2023	XGBoost	Human-written essays and ChatGPT generated essays	Authors explored TF-IDF and manual features for ChatGPT detection [37].
2023	Transformer-based ML Model	ChatGPT query dataset, ChatGPT rephrase Dataset	Authors explored the difference between AI and human writing [38].

III. PROPOSED FRAMEWORK

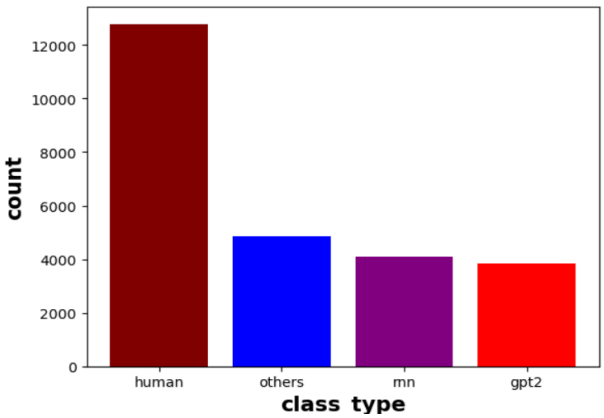
This section gives a summary of the dataset employed both in the research and the ways employed for feature engineering. also, it discusses the selection and perpetration of Deep learning as well as conventional machine learning models infrastructures.



1. FIGURE: Methodology framework.

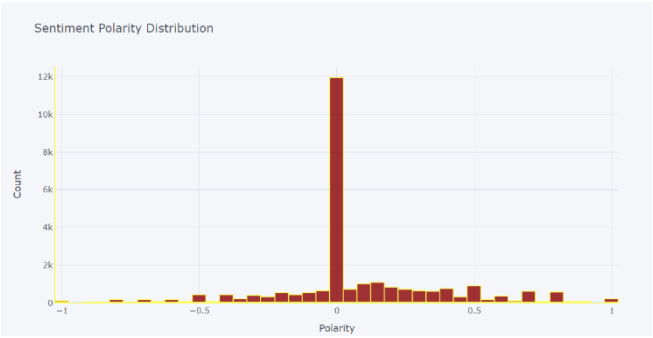
A. *DATASET AND DATA PRE-PROCESSING*

This study utilizes the Tweepfake Dataset comprising user profiles and tweet data from Twitter. The dataset contains 25,572, entries in total, categorized into human and bot accounts. Among them, there are 17 human accounts contributing 12,786 tweets and 23 bot accounts contributing to the remaining tweets. Each entry is properly labeled indicating whether the user is a bot or not.

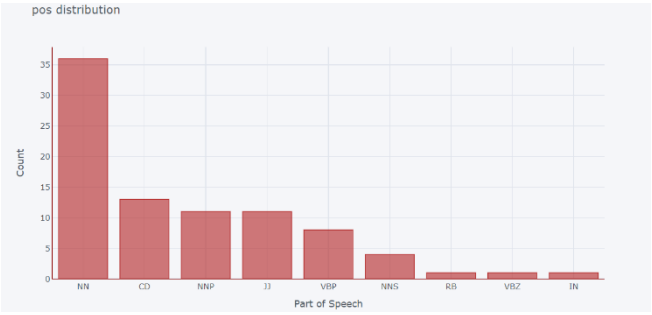


2. FIGURE: A count plot displaying the distribution of data by class.

Datasets often contain unstructured or semi-structured data, which may include irrelevant information. This unnecessary data not only prolongs the model's training time but can also adversely affect its performance. Preprocessing is essential to enhancing the efficiency of machine learning models as well as preservation computational resources. By Getting the text ready, the model's capacity for accurately expect results is improved. Preprocessing typically involves the subsequent actions: tokenization, case converting, stop word removal, and elimination of numerical values.



3. FIGURE :Text Sentiment Polarity distribution.



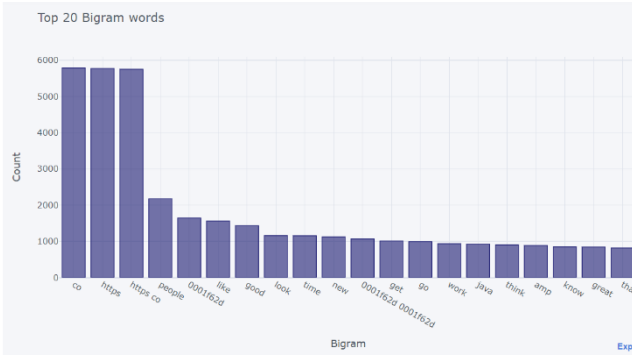
4. FIGURE :Text words are part of speech distribution

B. *FEATURE EXTRACTION*

In order to effectively develop models for machine learning, it's essential to employ feature extraction ways that transfigure raw data into an organized style that is appropriate for model training. The following feature extraction techniques are used in this study:

1 BAG-OF-WORDS

The Bag-of-Words (BoW) method for feature extraction, a foundational method in natural language processing. BoW represents text data by quantifying the occurrence frequency of each word within a document, disregarding word order and grammar. This approach transforms textual information into a structured format suitable for machine learning models. By creating a sparse matrix where rows represent documents and columns represent unique words in the corpus, BoW captures the essence of document content. Despite its simplicity, BoW provides valuable insights into the textual data, enabling models to learn patterns and make informed predictions. Its intuitive nature and effectiveness make it popular choice for various text-based applications.



5. FIGURE : Bigram Bag of words

2 TF-IDF

The feature extraction method known as TF-IDF (Term Frequency-Inverse Document Frequency), a powerful method widely used in natural language processing tasks. TF-IDF assesses a word's significance inside a document in relation to a broader corpus. The computation involves assigning a value to every word considering both its occurrence frequency within the document (TF) and its scarcity across the corpus (IDF) [29]. This approach allows us to capture the significance of words while mitigating the impact of common terms. By giving terms that appear frequently in a document but infrequently in the corpus more weights, TF-IDF emphasizes words that are discriminative for classification tasks. This method offers valuable insights into textual data, facilitating accurate model training and prediction.

3 WORD2VEC

Word2Vec is a natural language processing (NLP) technology that uses vectors in a space with several dimensions to represent words and help us grasp their relationships. Now, with gensim, we can easily implement Word2Vec and play around with it. The idea is that Word2Vec learns from a bunch of text data, like a big collection of articles or tweets, and it learns to represent each word as a vector. These vectors obtain the semantic significance of words based on how they're used in context.

Word2Vec has two main models: Words in a Continuous Bag (CBOW) and Skip-gram. While Skip-gram predicts surrounding context relevant terms given a target word, which CBOW forecasts depending on its surrounding context words. With gensim, we can train our own Word2Vec models by feeding it our text data. We can then use these trained models to find similar words, calculate word similarities, or even visualize word embeddings in a 2D space.Word2Vec is that it can capture subtle relationships between words. As an illustration, it can understand that "king" is to "queen" as "man" is to "woman", or that "Paris" is similar to "France" in the same way "Berlin" is to "Germany".

In summary, Word2Vec with gensim is a powerful tool for understanding the semantic relationships between words in text data. It's easy to use and can provide valuable insights into how language works.

C. MODELS OF MACHINE LEARNING AND DEEP LEARNING

Machine learning (ML) and deep learning (DL) models have enabled computers to draw conclusions and forecasts from data or judgments without explicit programming, revolutionizing a number of fields. In this section, we will discuss some popular ML and DL models commonly used in various applications.

1 LOGISTIC REGRESSION

The fundamental of logistic regression are ML algorithm utilized in tasks involving binary classification. It basically guesses the chance that something you put in fits into one category or another, using this thing called the logistic function. As the name suggests, logistic regression is commonly utilized for categorization as opposed to regression tasks due to its simplicity and effectiveness, especially when the features and the goal variable have a linear or perhaps transformed to be linear. Logistic regression is like a decent tool we use to figure out if something is one thing or another, especially when we only have two choices. Its simulates the likelihood that given input x utilizes the logistic function to determine whether a given class, y, which is defined as:

II. Equation [30]

$$P(Y = 1|X = x) = \frac{1}{1+e^{(\beta_0 + \beta_1 x)}}$$

2 RANDOM FOREST CLASSIFIER

Random forest comparable to a bunch of friends working together to make decisions. They each make their own guesses (like trees growing in a forest), and then we just go with what most of them think (for picking choices) or their average guess (for estimating numbers). It is known for its robustness, scalability, and capability to handle high-dimensional data with categorical and numerical features. Random forest models are particularly effective for classification tasks and can handle complex interactions between features.

3 SUPPORT VECTOR CLASSIFIER

Strong supervised learning models called support vector machines (SVMs) are employed for regression and classification problems. SVMs identify the ideal hyperplane that divides the classes in the feature space. Support Vector Classifier (SVC) is a variant of SVM specifically designed for classification tasks. It is useful for both linear and non-linear classification since it seeks to minimize classification errors while maximizing the margin between classes problems. The simple version of the SVM thingy tries to draw this line that splits the different groups in our data space the best it can. When the groups are easy to split in a straight line, we get this formula to decide which side something belongs to:

III. Equation [30]

$$f(x) = sign(\boldsymbol{w}.\boldsymbol{x} + b)$$

4 K NEAREST NEIGHBORS CLASSIFIER

A straightforward but powerful non-parametric technique for classification and regression problems is the K Nearest Neighbor (KNN) algorithm. It groups things according to the most common class among their k closest neighbors in the area of features. Since KNN makes no assumptions regarding the distribution of the underlying data, it versatile and suitable for both linear and non-linear decision boundaries. However, its performance may degrade with high-dimensional or noisy data. Given the KNN method determines the Euclidean distance between each new data point and every other point in the training dataset. On the basis of these distances, it then chooses the k closest neighbors. A majority vote among the new data point's closest neighbors determines the class label.

5 CONVOLUTION NEURAL NETWORKS (CNN)

Deep learning models are CNNs widely employed in image identification, classification, segmentation, and other computer vision tasks. Their purpose is to automatically and adaptably learn the spatial characteristics' hierarchies from the input data through the application of convolutional filters. CNNs excel at capturing local patterns and spatial dependencies in images, making them highly effective for tasks involving visual data. The output of CNN is obtained through a series of mathematical operations, including convolution, activation, pooling, and fully connected layers.

6 LONG SHORT TERM MEMORY (LSTM)

One kind of recurrent neural network (RNN) is the LSTM architecture designed to model sequential data with long-range dependencies, like NLP, or natural language processing tasks, examination of time series, and speech recognition. Unlike traditional RNNs, LSTM networks can selectively recall or lose track of knowledge arbitrary time intervals, making them ideal for capturing temporal dynamics and handling vanishing or exploding gradient problems. The equations governing the behavior of a LSTM cell are as follows:

IV. Equation [31]

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i)$$
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f)$$
$$c_t = f_t \odot c_{t-1} i_t + \tanh \odot (W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$
$$o_t = \sigma(W_{x0}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_c)$$
$$h_t = o_t \odot \tanh(c_t)$$

D. DJANGO

Django stands out as a versatile and robust web framework, empowering developers to build complex web applications swiftly and efficiently. Conceived in 2003 by Adrian Holovaty and Simon Willison, Django embodies Python's philosophy of readability and simplicity, providing developers with a powerful toolkit to streamline web development processes.

At the heart of Django lies its Model-View-Template (MVT) architecture, an alternative to the traditional Model-View-Controller (MVC) pattern. This architectural design separates the data model, user interface, and business logic, fostering a clean and modular codebase that promotes scalability and maintainability.

Django's Object-Relational Mapping (ORM) system abstracts database interactions into Python objects, reducing the complexity of database operations and facilitating cross-database compatibility. Developers can now concentrate on application logic instead of database design thanks to this abstraction management, accelerating the development process.

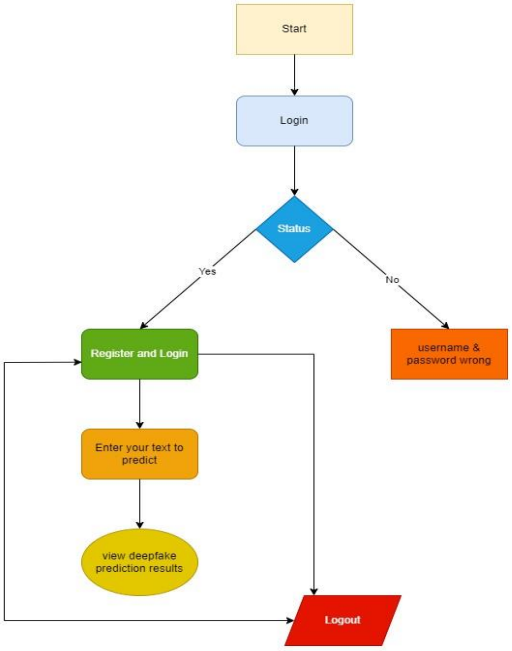
The framework's built-in administrative interface further enhances productivity by providing a user-friendly platform for managing site content and executing database record CRUD (Create, Read, Update, Delete) operations. With minimal configuration, developers can leverage the admin interface to interact with their application's data effortlessly.

Form handling in Django is seamless, thanks to its robust form library, which simplifies form creation, validation, and processing. Developers can define forms using Python classes and seamlessly integrate them into their applications, streamlining user input validation and ensuring data integrity.

Django's routing of URLs mechanism enables developers to map URLs to observe features, enabling tidy and intuitive URL designs. This routing system simplifies navigation within web applications and enhances user experience by providing logical URL structures.

The framework's template engine empowers developers to create dynamic web pages using HTML templates with embedded Python code. This separation of presentation and logic promotes code reusability and facilitates the development of responsive and maintainable web applications.

Security is a top priority in Django, with built-in features to mitigate prevalent online vulnerabilities such cross-site request forgery (CSRF), SQL injection, and cross-site scripting (XSS). Django's robust security measures, including its authentication system and middleware, provide developers with peace of mind when building secure web applications.



6. FIGURE: Flow chart Web app.

Django finds widespread adoption across diverse domains, such as social networking sites, e-commerce platforms, and content management systems (CMS), data analytics tools, and real-time applications. Its versatility, scalability, and extensive feature set make it a preferred choice for developers seeking to build sophisticated web applications.

In conclusion, Django's elegant design, coupled with its comprehensive features and emphasis on simplicity, cements its position as a leading web framework for rapid and efficient web development in the Python ecosystem.

E. METHODOLOGY

This segment outlines the approach used to categorize tweets, including the application of both conventional machine learning models and deep learning. The aim is to develop an effective framework capable of distinguishing between human and bot accounts based on tweet content.

Models for Convolutional Neural Networks (CNNs) are examples of deep learning provide significant capabilities in automatically learning intricate features from text inputs. They excel in capturing hierarchical patterns, regional links, and enduring ties, thereby enabling the extraction of meaningful representations from textual data. By leveraging stacked layers of CNNs, the model can effectively capture dependencies within the text.

In this work, we provide a hybrid methodology integrating both deep learning and traditional machine learning techniques for tweet classification. Alongside the CNN model, we incorporate several classical machine learning algorithms, including Logistic Regression, Random Forest Classifier, Support Vector Classifier, and K-Nearest Neighbor Classifier.

The methodology begins with the acquisition of a dataset with labels sourced from a publicly available repository. This tweets are included in the dataset originating from both accounts from both humans and bots. To enhance the quality of the text and simplify the data, several preprocessing actions are applied, including text cleaning and normalization.

The dataset is then divided for training and testing into an 80:20 ratio, ensuring the model's performance can be accurately evaluated. The following action entails changing the textual data into numerical using word embedding in vectors, a technique known for capturing semantic information effectively.

For the deep learning component, a CNN architecture is employed. This CNN model is designed to process the vectorized tweet data, extracting relevant features and patterns. Simultaneously, the traditional machine learning models, including K-Nearest Neighbor Classifier, Support Vector Classifier, Random Forest Classifier, and Logistic Regression - are trained on the same dataset.

The efficacy of the proposed methodology is assessed using four important evaluation parameters: F1-score, recall, accuracy, and precision. These metrics offer in-depth understanding of the categorization models' performance, enabling an extensive comparison between the both conventional machine learning techniques and deep learning.

To sum up, the methodology presented in this research combines the strengths of combining conventional machine learning methods and deep learning approaches for tweet classification. By leveraging the complementary capabilities of these models, we aim to develop a robust framework capable of accurately distinguishing between human and bot-generated tweets.

IV. RESULT

In this section, we delve into the experimental procedures conducted for this research and delve into the ensuing findings. Our study aims to detect deepfake tweets using models from both deep learning and machine learning. To validate our proposed methodology, we have employed a diverse array of machine learning models, namely Logistic Regression, Random Forest, K-Nearest Neighbors, Support Vector Machine, and Naive Bayes. The characteristics and intricacies of each model are elucidated in Table II. These models have been implemented with hyperparameters meticulously tailored to our dataset's nuances. Through rigorous fine-tuning, we have optimized the value ranges of these hyperparameters to ensure optimal performance.

A. MACHINE LEARNING MODELS RESULT

In this section, we delve into the thorough experiments conducted as part of this study, along with a thorough analysis of the results obtained. Our research focuses on deepfake text detection, exploring a range of feature engineering strategies to enhance detection accuracy. We compare the performance of frequency-based methods like TF-IDF and a bag of words, to discern their efficacy in aiding supervised machine learning models. Specifically, we assess the performance of Logistic Regression, Random Forest, Naive Bayes, K-Nearest Neighbors, and Support Vector Machine models across several assessment criteria, such as F1 score, accuracy, recall, and precision.

Notably, the effectiveness of each model is contingent upon the feature extraction method employed, elucidating the nuanced interplay between feature engineering and model performance.

I. TABLE: Machine learning accuracy results

Models	**Bag-of-words**	**TF-IDF**	**Word2Vec**
Logistic Regression	0.784	0.778	0.744
Random Forest	0.768	0.773	0.799
Support Vector Machine	0.779	0.789	0.794
Naive Bayes	0.740	0.758	0.701
K-Nearest Neighbors	0.697	0.531	0.778

B. RESULT OF DEEP LEARNING MODELS

In this section, we delve into the utilization of cutting-edge deep learning architectures, such as Long Short-Term Memory networks (LSTM) and Convolutional Neural Networks (CNN). Renowned for their prowess in text classification tasks, these deep learning models have garnered widespread acclaim in literature. The embedding layer serves as the initial processing step, transforming each word in the input data into vector representations to facilitate model training. Meanwhile, the layer that drops out performs a pivotal role in mitigating enhancing and overfitting model generalization through haphazard deactivating neurons during training. To generate the final predictions, the dense layer, coupled with a SoftMax activation function, amalgamates the model's learned features and produces the requisite output. Each the categorical cross-entropy loss function is utilized to instruct the model, with parameter optimization facilitated by the 'Adam' optimizer. Through the adoption of these cutting-edge deep learning architectures and optimization techniques, we aim to achieve superior performance in deepfake text detection tasks.

II. TABLE: Deep learning accuracy results

models	**Tokenizer**	**Word2Vec**
CNN	0.91	0.90
LSTM	0.89	0.88

V. CONCLUSION

In conclusion, the detection of deepfake text emerges as a critical and formidable challenge amidst the proliferation of misinformation and manipulated content in the digital age. This study has endeavoured to confront this challenge head-on by proposing a novel method for identifying deepfake text and rigorously assessing its efficacy. Leveraging a diverse dataset comprising tweets from both humans and bots, we employed a myriad of deep learning and machine learning models, complemented by sophisticated approaches for feature engineering. Notably, our experimentation encompassed the utilization of established feature extraction techniques like BoW, Word2Vec, Tokenizer and TF-IDF.

Our proposed approach, amalgamating methods like CNN and Tokenizer, has produced encouraging outcomes, reaching an impressive accuracy of 0.91 in effectively identifying deepfake text. Moreover, we carried out an exhaustive comparison of our approach with other cutting-edge transfer learning models documented in prior literature. Notably, the simplicity and computational efficiency inherent in the CNN model architecture emerged as a standout feature, facilitating superior performance and adept handling of out-of-vocabulary terms.

The results of this study highlight the possibility of leveraging CNN-based approaches in deepfake text detection tasks, obviating the requirement for intricate and time-intensive models of transfer learning.

VI. FUTURESCOPE

As we navigate the ever-evolving landscape of social media, the proliferation of deepfake content presents important challenges to the integrity of online discourse. In particular, the rise of falsified tweets and fabricated textual content highlights the critical necessity for reliable detection techniques to combat misinformation and preserve the credibility of digital communication channels. Moving forward, upcoming studies endeavors in the realm of deep learning hold immense potential for advancing the field of text authentication and verification.

One promising avenue for future exploration lies in the development of sophisticated deep learning models tailored specifically for identifying falsified tweets and other textual content on social media platforms. By leveraging cutting-edge neural network structures and natural language processing techniques, researchers can improve the precision and efficiency of detection algorithms, enabling more efficient identification of deepfake text in real-time.

In conclusion, Moving forward, an essential area of focus involves refining the User Interface (UI) of the website to enhance user experience and accommodate additional functionalities. Diverse front-end solutions present opportunities for augmenting the platform's accessibility and usability.

VII. REFERENCE

[1] J. P. Verma and S. Agrawal, ''Big data analytics: Challenges and applications for text, audio, video, and social media data,'' Int. J. Soft Comput., Artif. Intell. Appl., vol. 5, no. 1, pp. 41–51, Feb. 2016.

[2] H. Siddiqui, E. Healy, and A. Olmsted, ''Bot or not,'' in Proc. 12th Int.Conf. Internet Technol. Secured Trans. (ICITST), Dec. 2017, pp. 462–463.

[3] M. Westerlund, ''The emergence of deepfake technology: A review,''Technol. Innov. Manage. Rev., vol. 9, no. 11, pp. 39–52, Jan. 2019.

[4] J. Ternovski, J. Kalla, and P. M. Aronow, ''Deepfake warnings for political videos increase disbelief but do not improve discernment: Evidence from two experiments,'' Ph.D. dissertation, Dept. Political Sci., Yale Univ., 2021.

[5] S. Vosoughi, D. Roy, and S. Aral, ''The spread of true and false news online,'' Science, vol. 359, no. 6380, pp. 1146–1151, Mar. 2018.

[6] S. Bradshaw, H. Bailey, and P. N. Howard, ''Industrialized disinformation: 2020 global inventory of organized social media manipulation,'' Comput. Propaganda Project Oxford Internet Inst., Univ. Oxford, Oxford, U.K., Tech. Rep., 2021.

[7] C. Grimme, M. Preuss, L. Adam, and H. Trautmann, ''Social bots: Humanlike by means of human control?'' Big Data, vol. 5, no. 4, pp. 279–293, Dec. 2017.

[8] X. Liu, Y. Zheng, Z. Du, M. Ding, Y. Qian, Z. Yang, and J. Tang, ''GPT understands, too,'' 2021, arXiv:2103.10385.

[9] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, and Y. Choi, ''Defending against neural fake news,'' in Proc. 33rd Int. Conf. Neural Inf. Process. Syst. (NIPS), Dec. 2019, pp. 9054–9065, Art. no. 812.

[10] L. Beckman, ''The inconsistent application of internet regulations and suggestions for the future,'' Nova Law Rev., vol. 46, no. 2, p. 277, 2021, Art. no. 2.

[11] J.-S. Lee and J. Hsiang, ''Patent claim generation by fine-tuning OpenAI GPT-2,'' World Pat. Inf., vol. 62, Sep. 2020, Art. no. 101983.

[12] R. Dale, ''GPT-3: What's it good for?'' Natural Lang. Eng., vol. 27, no. 1, pp. 113–118, 2021.

[13] W. D. Heaven, ''A GPT-3 bot posted comments on Reddit for a week and no one noticed,'' MIT Technol. Rev., Cambridge, MA, USA, Tech. Rep., Nov. 2020, p. 2020, vol. 24. [Online]. Available: www.technologyreview.com

[14] S. Gehrmann, H. Strobelt, and A. M. Rush, ''GLTR: Statistical detection and visualization of generated text,'' 2019, arXiv:1906.04043.

[15] D. I. Adelani, H. Mai, F. Fang, H. H. Nguyen, J. Yamagishi, and I. Echizen, ''Generating sentiment-preserving fake online reviews using neural language models and their human- and machine-based detection,'' in Proc. 34th Int. Conf. Adv. Inf. Netw. Appl. (AINA). Cham, Switzerland: Springer, 2020, pp. 1341–1354.

[16] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, and Y. Choi, ''Grover—A state-of-the-art defense against neural fake news,'' in Proc. Adv. Neural Inf. Process. Syst., vol. 32. Curran Associates, 2019. [Online]. Available: http://papers.nips.cc/paper/9106-defending-againstneural-fake-news.pdf

[17] N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong, and R. Socher, ''CTRL: A conditional transformer language model for controllable generation,'' 2019, arXiv:1909.05858.

[18] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, ''Synthesizing obama: Learning lip sync from audio,'' ACM Trans. Graph., vol. 36, no. 4, pp. 1–13, Aug. 2017.

[19] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner, ''Face2Face: Real-time face capture and reenactment of RGB videos,'' in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 2387–2395.

[20] C. Chan, S. Ginosar, T. Zhou, and A. Efros, ''Everybody dancenow,'' in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 5932–5941.

[21] Y. Jia, Y. Zhang, R. Weiss, Q. Wang, J. Shen, F. Ren, P. Nguyen, R. Pang, I. L. Moreno, and Y. Wu, ''Transfer learning from speaker verification to multispeaker text-to-speech synthesis,'' in Proc. Adv. Neural Inf. Process. Syst., vol. 31, 2018, pp. 4485–4495.

[22] Y. Wang, K. Wang, Y. Wang, D. Guo, H. Liu, and F. Sun, ''Audio-visual grounding referring expression for robotic manipulation,'' in Proc. Int. Conf. Robot. Autom. (ICRA), May 2022, pp. 9258–9264.

[23] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, ''Language models are unsupervised multitask learners,'' OpenAI Blog, vol. 1, no. 8, p. 9, 2019.

[24] Y. Zhou, J. Yang, D. Li, J. Saito, D. Aneja, and E. Kalogerakis, ''Audiodriven neural gesture reenactment with video motion graphs,'' in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 3408–3418.

[25] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, ''Improving language understanding by generative pre-training,'' OpenAI, Amer. AI Res. Lab., Tech. Rep., 2018.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, ''BERT: Pre-training of deep bidirectional transformers for language understanding,'' 2018, arXiv:1810.04805.

[27] C. Li, X. Gao, Y. Li, B. Peng, X. Li, Y. Zhang, and J. Gao, ''Optimus: Organizing sentences via pre-trained modeling of a latent space,'' 2020, arXiv:2004.04092

[28] P. von Platen, ''How to generate text: Using different decoding methods for language generation with transformers,'' Hugging Face, Manhattan, NY, USA, Tech. Rep., 2020.

[29] S. Robertson, ''Understanding inverse document frequency: On theoretical arguments for IDF,'' J. Document., vol. 60, no. 5, pp. 503–520, Oct. 2004.

[30] Christopher M. Bishop, ''Pattern Recognition and Machine Learning,'' Springer,738(7), 29-60, 2006.

[31] Sepp Hochreiter, and Jurgen Schmidhuber ''Long short-term memory,'' Neural computation, 9(8), 1735-1780, 1997.

[32] Saima Sadiq, Turki Aljrees, Saleem Ullah. "Deepfake Detection on Social Media: Leveraging Deep Learning and FastText Embeddings for Identifying MachineGenerated Tweets" , IEEE Access, 2023.

[33] S. Kudugunta and E. Ferrara, ''Deep neural networks for bot detection,'' Inf. Sci., vol. 467, pp. 312–322, Oct. 2018.

[34] D. Dukić, D. Keča, and D. Stipić, ''Are you human? Detecting bots on Twitter using BERT,'' in Proc. IEEE 7th Int. Conf. Data Sci. Adv. Anal.(DSAA), Oct. 2020, pp. 631–636.

[35] T. Fagni, F. Falchi, M. Gambini, A. Martella, and M. Tesconi, ''TweepFake: About detecting deepfake tweets,'' PLoS ONE, vol. 16, no. 5, May 2021, Art. no. e0251415.

[36] S. Najari, M. Salehi, and R. Farahbakhsh, ''GANBOT: A GAN-based framework for social bot detection,'' Social Netw. Anal. Mining, vol. 12, no. 1, pp. 1–11, Dec. 2022.

[37] R. Shijaku and E. Canhasi, ''ChatGPT generated text detection,'' Tech. Rep., 2023.

[38] S. Mitrović, D. Andreoletti, and O. Ayoub, ''ChatGPT or human? Detect and explain. Explaining decisions of machine learning model for detecting short ChatGPT-generated text,'' 2023, arXiv:2301.13852.