Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Distinguishing Human-Generated and Machine-Generated Text Using Deep Learning

Neeti Sangwan[1*], Anu Saini[2], Lakshay Gupta[3], Mayank Rawat[4] and Mukul Mahawar[5]
[1]*Assistant Professor, Computer Science & Engineering, Maharaja Surajmal Institute of Technology, New Delhi, India.*
[2]*Assistant Professor, Computer Science & Engineering, G.B. Pant DSEU Campus, New Delhi, India.*
[3,4,5]*Student, Computer Science & Engineering, Maharaja Surajmal Institute of Technology, New Delhi, India.*
** Corresponding Author: Neeti Sangwan*
neetisangwan@gmail.com

ARTICLE INFO

Received: 20 Oct 2024

Revised: 21 Nov 2024

Accepted: 18 Dec 2024

ABSTRACT

The advent of machine learning and natural language processing has led to significant advancements in text generation, which differentiate human-written content from machine-generated content. Differentiating between these two sources has become crucial in various domains, including journalism, academia, cybersecurity, and content moderation. This research delves into developing a deep learning model that accurately distinguishes between human and machine-generated texts and addresses the challenges posed by the proliferation of automated text generation systems. A dataset comprising 100,000 labelled instances has been used to train/test and predict. The model achieves a commendable accuracy to outperform the state-of-art.

Keywords: Human-generated, Machine-generated, Deep Learning, Convolution

1. INTRODUCTION

Machine-generated Text is becoming increasingly common in a variety of applications, such as news articles, social media posts, and product descriptions. However, malevolent use of machine-generated text can also be possible in disseminating misinformation or propaganda.

In the digital age, the proliferation of advanced Large Language Models (LLMs) has led to an unprecedented ability for machines to generate text that closely resembles human language. With advancement of powerful deep learning techniques, particularly in the field of natural language processing (NLP), has resulted in a paradigm shift in text generation. It becomes very important to distinguish between human-generated content and machine-generated text as it raises critical challenges for various applications, including content moderation, cybersecurity, and ensuring the authenticity of information. Traditional approaches for detecting machine-generated text, such as rule-based systems or keyword analysis, often struggle to keep pace with the rapid evolution of AI-generated content. As a result, researchers and practitioners are increasingly turning to deep learning methodologies, leveraging neural networks, and advanced architectures to tackle this complex problem.

Deep learning models offer the promising potential to learn intricate patterns from large volumes of data. Methods such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and transformer-based architectures have shown promising results in capturing the intricacies that distinguish between human and machine-generated text. Detection of machine-generated text employs techniques like anomaly detection, linguistic analysis, and neural network models trained on diverse datasets.

In this paper, different deep learning models are studied and implemented to recognize human and machine-generated text. Through this research, various stages in detecting machine-generated text are studied and comparison is done with various methodologies and approaches used in the distinct research papers. Section 2 provides a review of work done in the field of text detection and its key algorithms and techniques. Section 3 details the dataset considered for experimentation and the steps followed to preprocess the dataset before experimentation.

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Section 4 provides the detailed architecture of the base model and other implemented models including the proposed model. Section 5 explains the prediction results obtained from the models discussed in section 4. Models are evaluated based on accuracy, precision, recall, and F1-score. Section 6 concludes the work.

2. LITERATURE REVIEW

Earlier work in this field has entailed the usage of methods such as frequency count, n-grams, and deep learning models to produce a high accuracy score.

Recent studies have used various algorithms like XGBoost, Random Forest, and Multi-Layer Perceptron (Liao et al., 2023) to achieve machine text detection. Choosing any one model over another may depend upon the application. Various features for classifying human and AI-generated texts, particularly focusing on ChatGPT were investigated (Mindner et al., 2023). Their findings on the effectiveness of lexical and syntactic features were instrumental in designing our feature extraction methods. A comprehensive survey on the automatic detection of machine-generated texts, exploring various algorithms and their performance has been provided that gives existing methodologies and their limitations (Jawahar et al.,2020). Machine learning techniques were explored to differentiate between human and ChatGPT-generated texts (Katib et al., 2023). Their work on leveraging traditional machine learning algorithms and modern neural networks helps to use a hybrid approach for better accuracy. Text detection in the Persian language, demonstrating the universality of detection techniques across languages to highlight the importance of linguistic diversity in training models, influencing the dataset selection to include various languages (Khanmohammadi, 2023). Shijaku and Canhasi (2023) emphasized the challenges of detecting ChatGPT-generated texts, particularly addressing the evolving nature of AI models. Their research underscored the need for continuously updating detection models. Deep transformer models for detecting and classifying ChatGPT-generated content were examined (Oghaz et al., 2023). Their findings were crucial in selecting our model architecture for this study.

Fröhling and Zubiaga, 2021 investigated feature-based detection of automated language models, including GPT-2 and GPT-3. Their exploration of specific features, such as perplexity and burstiness, provided insights into feature engineering for proposed models. Nguyen and Echizen, 2018 focused on detecting computer-generated text using fluency and noise features. Their innovative approach to leveraging text coherence and grammaticality inspired aspects of the feature extraction process.

The foundational work on distributed representations of words and phrases introduced key concepts in word embeddings, which are integral to modern NLP models (Mikolov et al., 2013 and Mikolov et al., 2013). This work laid the groundwork for understanding the vector space representations used in implemented deep learning models. Goodfellow et al. (2014) introduced Generative Adversarial Networks (GANs), which are pivotal in generating realistic synthetic data. Understanding GANs helped authors to comprehend the underlying mechanics of text generation models like GPT, for detection strategies. Goodfellow et al. (2016) provided a comprehensive overview of deep learning techniques, offering foundational knowledge crucial for developing the models. Radford et al. (2018) proposed generative pretraining for language understanding, significantly advancing the capabilities of language models. This approach is baseline to the architecture of GPT models that are implemented by the authors in this paper. Vaswani et al. (2017) introduced the Transformer model, a groundbreaking architecture that forms the basis of many modern NLP models, including GPT. Their work on attention mechanisms was critical in developing proposed deep learning approach. Devlin et al. (2018) presented BERT, a model that excels in understanding contextual relationships in text. The techniques used in BERT influenced the authors' choice of model architecture for text detection tasks. Yang et al. (2019) developed XLNet, which enhanced autoregressive pretraining methods. Their improvements over previous models provided valuable insights into optimizing the model's performance. Kiros et al. (2015) introduced skip-thought vectors, a method for learning continuous representations of sentences for sentence-level feature extraction. Rajpurkar et al. (2016) created the SQuAD dataset, a benchmark for machine comprehension of text. This dataset ensures robust performance and generalization for training and evaluating the models. Abadi et al. (2016) developed TensorFlow, a system crucial for large-scale machine learning that facilitated implementing and training implemented deep learning models. Conneau et al. (2017) explored supervised learning of universal sentence representations, providing methods to enhance the generalization of the models across different text types.

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Bahdanau, Cho & Bengio (2014) introduced neural machine translation by jointly learning to align and translate, influencing the implemented approach to sequence-to-sequence learning in text detection. Kim (2014) demonstrated the use of convolutional neural networks for sentence classification, providing a methodological framework that inspired part of the model architecture.

3. DATASET AND PREPROCESSING

The main contribution of this work revolves around using a robust and diverse dataset capable of representing both human-written and machine-generated text. Such a dataset is vital for training and testing the machine learning model designed to distinguish between the two.

The OpenAI GPT-2 detector dataset is considered in this work. This dataset contains 100,000 labelled instances of human and machine-generated text along with their classification (False implies machine-generated text and True indicates human-generated text). Data from various sources like Kaggle, and GitHub, LLM models like GPT and BARD are used to produce the unique and comprehensive dataset. Small segment of the dataset is shown in fig. 1.

id	text	length	ended
1	is one of many that have been struggling	1024	FALSE
2	refine. 135 SHARES Share Tweet	1024	FALSE
3	The fate of a pig was not decided until	1024	FALSE
4	DoJ's scheduled attacks are started	795	TRUE
5	- Virginian Journal blog	1024	FALSE
6	4 Atheists, Not Scientists, Learned	1024	FALSE
7	In a second hearing on Tuesday, U.S.	1024	FALSE
8	Soviet microbattery (generally, 3 watts)	227	TRUE

Fig. 1: Dataset

Data is first encoded using One-Hot encoding. This encoding convert the nominal data into a format that allows the machine learning algorithms to understand and manipulate the data.

Tools developed specifically to derive dense vector representations for individual sentences like Tfidfvectorizer and Sentence Transformers are used to vectorize the data. MiniLM is a sentence transformer model that transforms input sentences into a vector of size 383. The size of this vector gives the number of features that are further used by the models as input. The main strength of MiniLM is that its speed of vectorizing a sentence is much greater than BERT but with slightly less accuracy. Computable vectors are generated from sentences by using all-MiniLM-v2. Fig. 2 shows the flow diagram of the steps followed to preprocess the dataset.

These embeddings then get summed to yield a single vector in a 768-dimensional embedding space. This space is the foundation for the comparison of human and machine-generated texts. A small instance of the final dataset after preprocessing is shown in fig. 3.

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

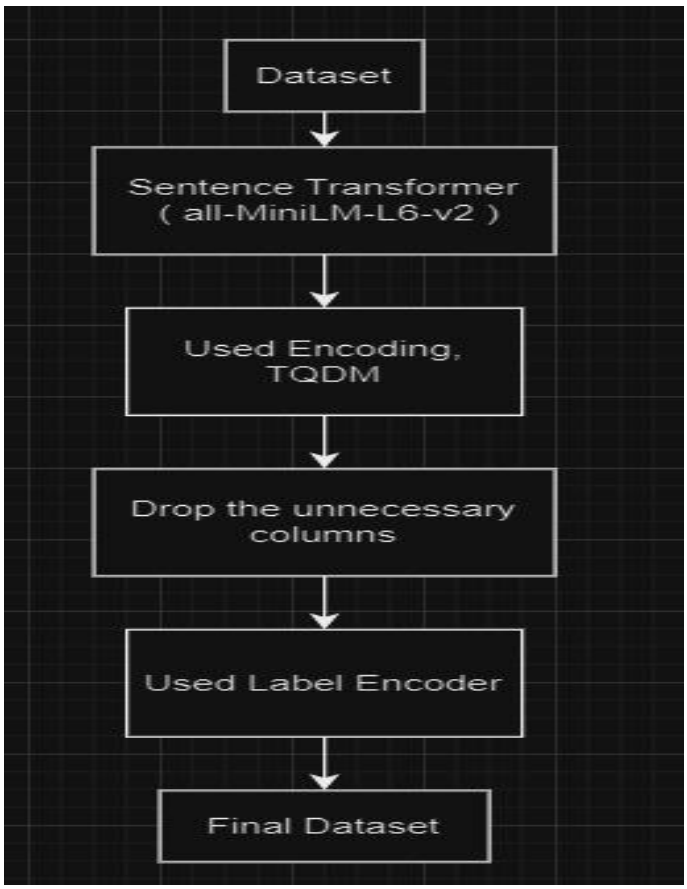

Fig. 2: Preprocessing of dataset

id	ended	0	1	2	3	4	5	6
0	0	0.112144	-0.02484	0.062954	0.059289	0.059902	0.013061	0.020567
1	0	-0.04016	-0.02977	0.039795	0.063586	0.051465	0.062381	-0.03509
2	0	0.129149	0.041479	0.028066	0.03196	0.010417	0.002107	-0.05105
3	1	-0.08362	0.058547	0.037818	0.029454	0.015685	0.002456	0.018189
4	0	-0.0407	0.003687	0.057338	-0.0376	0.089345	0.037959	0.002849
5	0	-0.03623	-0.03877	-0.0081	0.035167	0.087634	0.043605	-0.0106
6	0	-0.10229	0.018906	-0.11192	-0.00852	0.016294	0.023388	0.031596
7	1	-0.01811	0.03575	-0.07682	0.067177	-0.06527	-0.00581	-0.02147
8	0	-0.08027	0.003566	0.003111	-0.04405	0.001328	0.086353	0.023659

Fig. 3: Dataset after Sentence Transformer

4. MODELS

In this section, implementation details of CNN and LSTM-based model (Khanmohammadi, 2023) is explained. Later in this section, various endevours for generating the deep learning models for better results to differentiate between human-generated and machine-generated are explained followed by the details of the proposed model.

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

4.1 BASE MODEL

Outline of the base model architecture i.e. CNN + LSTM+ Dense is shown in fig. 4. In this architecture, first layer is input layer that accepts characters/ tokens present in the text data as an input. The input layer receives one 384/1 vector at a time. Output of this layer fed to 1-dimensional convolutional layer that is used to perform convolutions on input data to make connections of frequently occurring pairs of words. This convolutional layer extract combinations of word embedding frequently used by machine or human. It extracts features using 64 filters (size 30) across sequence and reduces the size to 355.

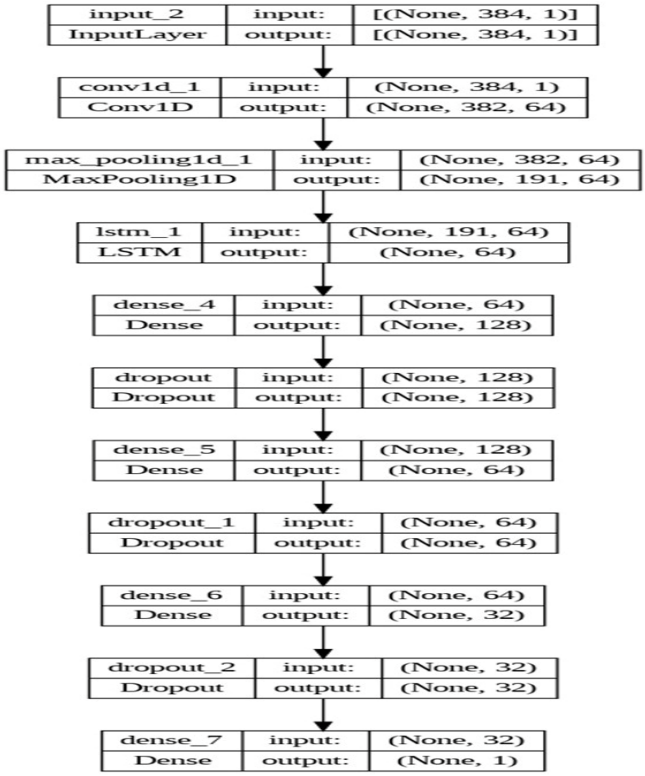

Fig. 4: Base Model Architecture

4.2 MODEL1: CNN+ 3 DENSE MODEL

The first layer of **CNN+3 Dense Model** is input layer that represents characters/ tokens present in the text data. The input layer receives one 384/1 vector at a time. Output of this layer fed to two layers: flatten layer and 1-dimensional convolutional layer. The flatten layer turns multidimensional data into one-dimensional data. The output of the Flatten layer is fed into a dense layer which is a simple fully connected layer. Three dense layers are used that applies a linear transformation to flattened the data from 384 to 64 using Relu activation function. After the Dense layer, batch normalisation and Dropout layers are used to prevent overfitting of the model. Batch normalization layer normalizes the activations fed from dense layer. The Dropout layer randomly assigns a fraction of the inputs a weight of 0 thus preventing overfitting. 1-dimensional convolutional layer is used to capture global patterns within the entire sequence. It extracts features using 128 filters (size 5) across sequence and reduces the size to 380. The output of the 1-dimensional convolutional layers is further fed to a batch normalisation layer which improves training speed by normalising the activations fed from the convolutional layer. The next layer is the concatenate layer that combines the outputs from both the streams (64,128) and merge their information into vector of size 192. The output of the concatenate layer is passed to a Dense layer that applies a final linear to predict the probability of machine-generated text. Fig. 5 represents the detailed architecture of CNN+3 Dense Model.

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

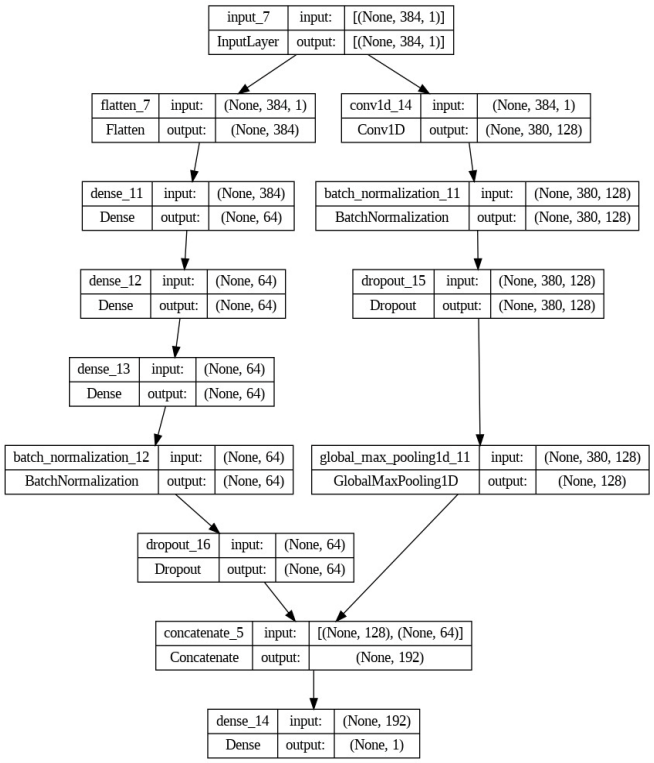

Fig. 5: Architecture of CNN+3 Dense Model

4.3 MODEL 2: 3 CNN+ 2 DENSE MODEL

In **3 CNN+2 Dense Model**, input layer accepts characters/ tokens present in the text data as an input. The input layer receives one 384/1 vector at a time as shown in fig. 6. Output of this layer fed to two layers: 1-dimensional convolutional layer and flatten layer. 1-dimensional convolutional layer is used to perform convolutions on input data. This convolutional layer extract combinations of word embedding frequently used by machine or human. It extracts features using 128 filters (size 5) across sequence and reduces the size to 380. The output of the 1-dimensional convolutional layers is further fed to a batch normalisation layer which improves the stability of the proposed model by normalising the activations before applying the activation functions for faster training. Three 1-dimensional convolutional layers are considered in a row. This is to increase the ability of the model to make connections of frequently occurring pairs of words.

The second layer which receives the output of the input layer is the flatten layer which is used to turn multidimensional data into one-dimensional data. The output of the Flatten layer is fed into a dense layer which is a simple fully connected layer. Two dense layers apply a linear transformation to flatten the data from 384 to 64 using Relu activation function. After the Dense layer, batch normalisation and Dropout layers are used to prevent overfitting of the model. The Dropout layer randomly assigns a fraction of the inputs a weight of 0 thus preventing overfitting. The next layer is the concatenate layer that combines the outputs from convolutional layer and dense layer into a vector of size 192. The output of the concatenate layer is passed to a Dense layer that applies a final transformation using sigmoid activation function as it predicts the class.

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

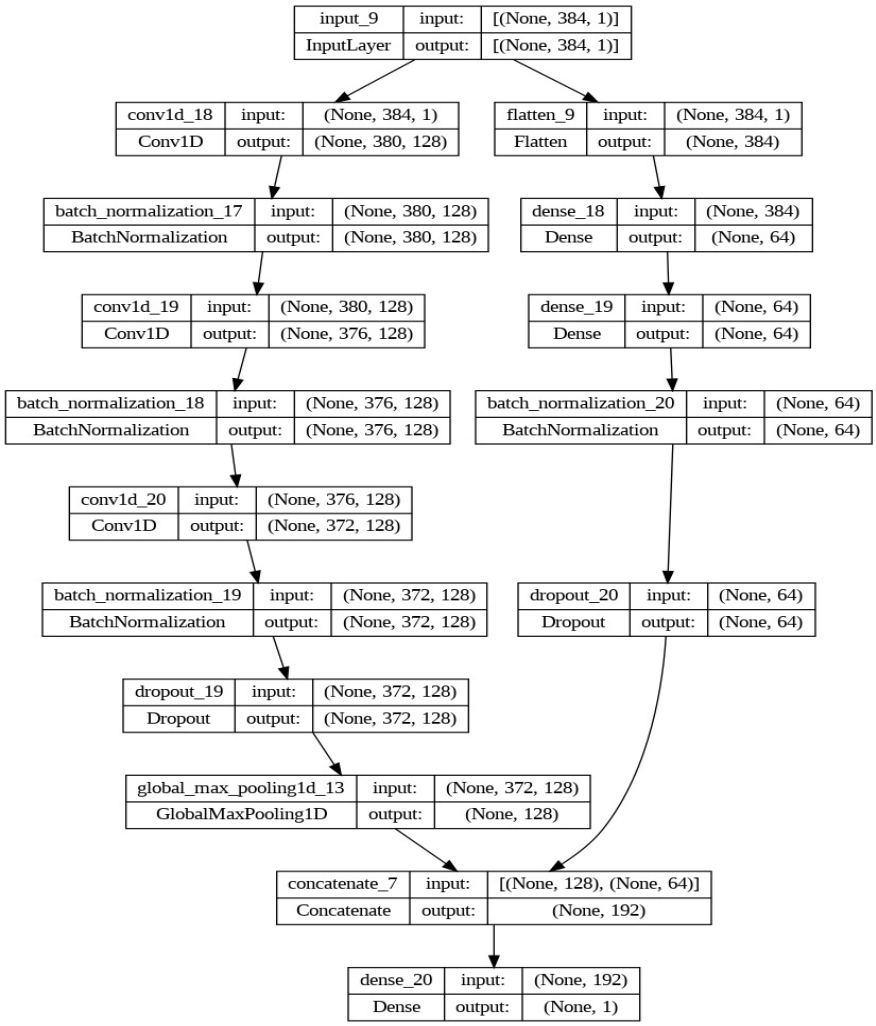

Fig. 6: Architecture of 3 CNN+2 Dense Model

4.4 MODEL 3: 4 CNN+ 3 DENSE MODEL

Fig. 7 shows the architecture of the proposed model i.e. 4 CNN +3 Dense model. The first layer is the input layer which takes text data as an input. The input layer receives one 384/1 vector at a time. Output of this layer fed to two layers: 1-dimensional convolutional layer and flatten layer. 1-dimensional convolutional layer is used to perform convolutions on input data. This convolutional layer extract combinations of word embedding frequently used by machine or human. It extracts features using 128 filters (size 5) across sequence and reduces the size to 380. The output of the 1-dimensional convolutional layers is further fed to a batch normalisation layer which improves the stability of the proposed model by normalising the activations before applying the activation functions for faster training. Four 1-dimensional convolutional layers are considered in a row. This is to increase the ability of the model to make connections of frequently occurring pairs of words. The second layer which receives the output of the input layer is the flatten layer which is used to turn multidimensional data into one-dimensional data. The output of the Flatten layer is fed into a dense layer which is a simple fully connected layer. Three dense layers are used that applies a linear transformation to flattened the data from 384 to 128, then flattened to 64 and finally to size 32 using Relu activation function. After the Dense layer, batch normalisation and Dropout layers are used to prevent overfitting of the model. The Dropout layer randomly assigns a fraction of the inputs a weight of 0 thus preventing overfitting. The next layer is the concatenate layer that combines the outputs from convolutional layer and dense layer into a vector of size 160. The output of the concatenate layer is passed to a Dense layer that applies a final transformation using sigmoid activation function as it predicts the class.

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

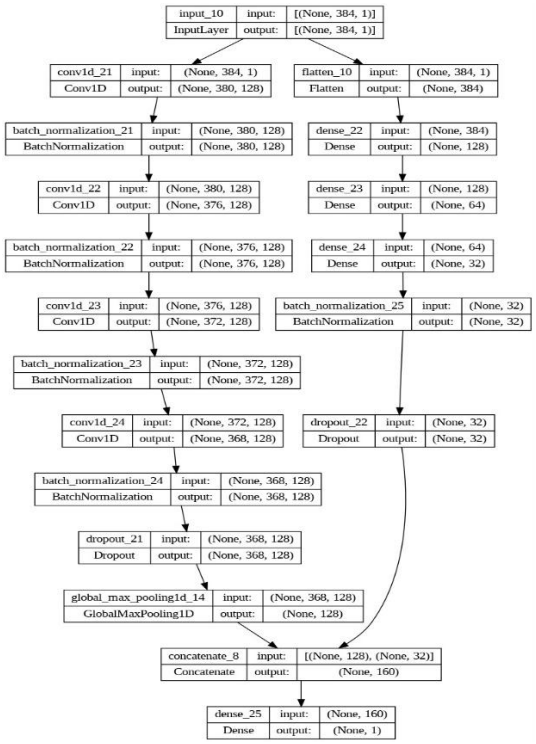

Fig. 7: Architecture of Proposed Model i.e. 4 CNN+3 Dense Model

5. RESULT AND DISCUSSION

In this section, performance of all the models are analysed and evaluated on the basis of their accuracy, precision, recall, and F1 score. These metrics provide a comprehensive view of the performance of the models across multiple dimensions. Accuracy can be deduced by summing the true positives and true negatives, then dividing by the total number of samples. Precision focuses on the proportion of true positive predictions (human-generated) out of all the instances that the model predicted as human-generated. Recall shows the model's ability to correctly identify all human-generated texts. F1-Score balance between precision and recall and provides overall performance.

Fig. 8, fig. 9, fig. 10 and fig. 11 show the confusion matrix of the architectures discussed in the previous section i.e. CNN+LSTM+Dense model, CNN+3 Dense model, 3 CNN+2 Dense model and 4 CNN+3 Dense model respectively.

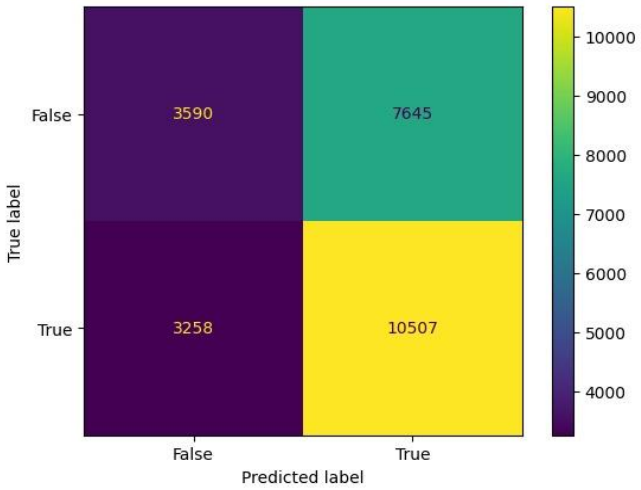

Fig. 8: Confusion Matrix of CNN+LSTM+Dense Model (Khanmohammadi, 2023)

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

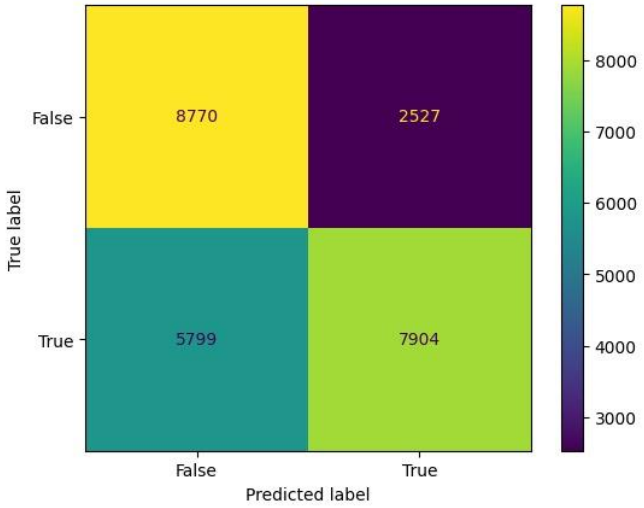

Fig. 9: Confusion Matrix of CNN+3 Dense Model

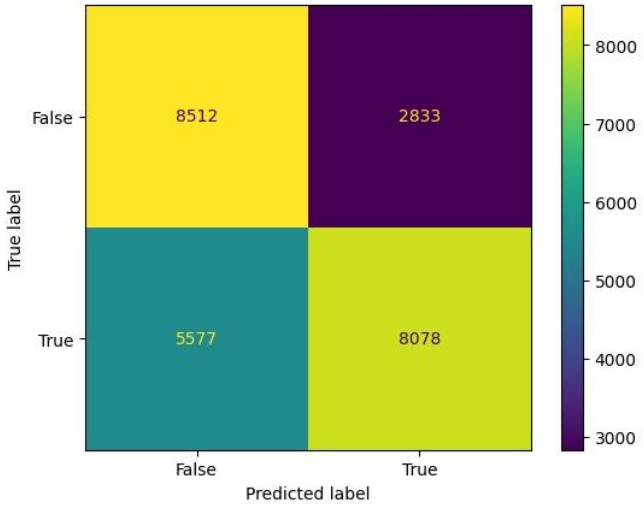

Fig. 10: Confusion Matrix of 3CNN+2 Dense Model

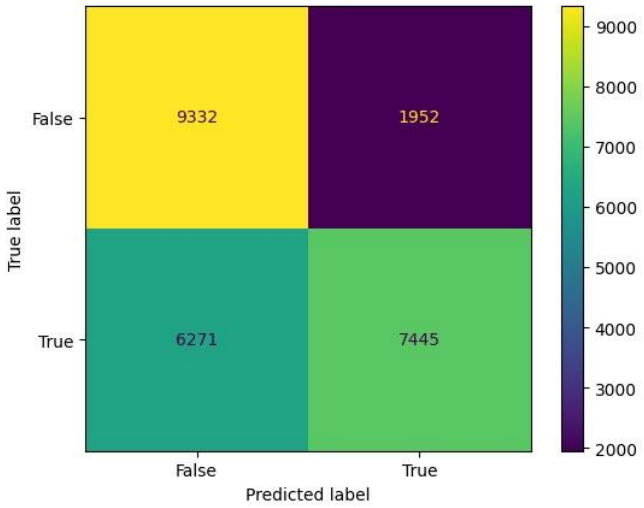

Fig. 11: Confusion Matrix of 4CNN+3 Dense Model

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The CNN+LSTM+Dense model (Khanmohammadi, 2023) correctly identified 3590 instances of machine-generated text (True label '0', Predicted label '0') and there are 7645 misclassifications i.e. True label '0' and Predicted label '1'. There are 3258 instances in which human-generated text is wrongly predicted as machine-generated. The model correctly identified 10507 instances of human-generated text. CNN+ 3 Dense Model and 3 CNN+2 Dense Model identify the 16,674 and 16590 instances respectively to the class they belong to. 8326 and 8410 instances are incorrectly predicted by CNN+ 3 Dense Model and 3 CNN+2 Dense Model respectively.

The proposed model i.e. 4 CNN+3 Dense model correctly identified 9332 instances of machine-generated text (True label '0', Predicted label '0') and there are 1952 misclassifications i.e. True label '0' and Predicted label '1'. There are 6271 instances in which human-generated text is wrongly predicted as machine-generated. The model correctly identified 7445 instances of human-generated text. The proposed model performs better than other models at identifying machine-generated text ('0') than human-generated text ('1'). Table 1 shows the performance of the distinct models employing Open AI GPT2 Detector dataset using the metrics: Accuracy, Precision, Recall and F1-score.

Table 1: Performance Evaluation

Model	Dataset	Accuracy	Precision	Recall	F1-score
Base model i.e. CNN+LSTM+Dense (Khanmohammadi, 2023)	Open AI GPT 2 Detector Dataset	56.38%	55.15%	54.14%	52.77%
CNN+ 3 Dense	Open AI GPT 2 Detector Dataset	66.69%	67.89%	68.10%	67.99%
3CNN+ 2 Dense	Open AI GPT 2 Detector Dataset	66.36%	67.45%	67.66%	67.55%
4 CNN + 3 Dense	Open AI GPT 2 Detector Dataset	**67.10%**	**69.52%**	**68.49%**	**69.00%**

The proposed model provides an accuracy of 67.10% which is higher in comparison to other considered models. All other metrics also shows that the proposed models predicts the result more correctly. Higher precision score indicates the reliability of the proposed model in correctly identifying each class. 68.49% of Recall value shows the ability of proposed model to capture a significant portion of each class. The F1 score balances precision and recall, providing a single metric to evaluate the overall performance. The proposed model outstands the state-of-art models in terms of F1-score. Fig. 12 shows the variation in the prediction accuracy of the models with increase in epochs in which base represents the base model i.e. CNN+LSTM+Dense, Model 1 represents CNN+3 Dense model, Model 2 represents 3CNN+2 Dense model and Model 3 represents the proposed model i.e. 4CNN+3 Dense model.

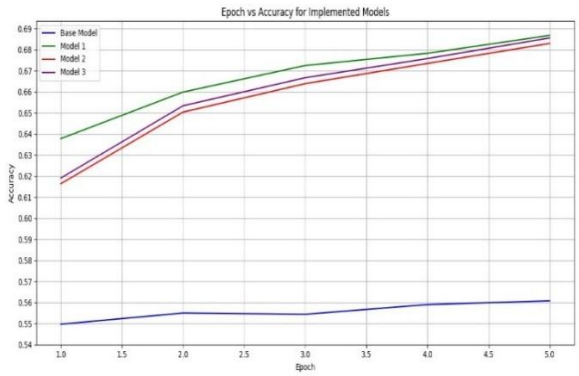

Fig. 12: Accuracy v/s Epochs

Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

6. CONCLUSIONS AND FUTURE WORK

In this paper, issue of identification of machine-generated text has been addressed. It is found that although machine-generated text improved significantly but it still poses challenges for the information security and authenticity verification areas. We have exploited various machine learning models to differentiate the human-generated and machine-generated text. Models with the different arrangements of convolution layers, LSTM layers and dense layers have been implemented and compared. Model proposed in [5] is taken as base model. Various strategies are employed to bolster the text detection model's accuracy in discerning human versus machine-generated content. Our proposed model shows the improved results over other considered models.

REFERENCES

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., ... Zheng, X. (2016). TensorFlow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)* (pp. 265–283).

[2] Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*

[3] Conneau, A., Kiela, D., Schwenk, H., Barrault, L., & Bordes, A. (2017). Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

[4] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[5] Fröhling, L., & Zubiaga, A. (2021). Feature-based detection of automated language models: Tackling GPT-2, GPT-3 and Grover. *PeerJ Computer Science, 7*, e443.

[6] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning* (Vol. 1, No. 2). MIT Press.

[7] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems* (pp. 2672–2680).

[8] Jawahar, G., Abdul-Mageed, M., & Lakshmanan, L. V. S. (2020). Automatic detection of machine generated text: A critical survey. In *Proceedings of the 28th International Conference on Computational Linguistics* (pp. 2296–2309).

[9] Katib, I., Assiri, F. Y., Abdushkour, H. A., Hamed, D., & Ragab, M. (2023). Differentiating Chat Generative Pretrained Transformer from humans: Detecting ChatGPT-generated text and human text using machine learning. *Mathematics, 11*(15), 3400.

[10] Khanmohammadi, R. (2023). Human vs machine generated text detection in Persian. *ResearchGate*.

[11] Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

[12] Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R. S., Torralba, A., Urtasun, R., & Fidler, S. (2015). Skip-thought vectors. In *Advances in Neural Information Processing Systems* (pp. 3294–3302).

[13] Liao, W., Liu, Z., Dai, H., Xu, S., Wu, Z., Zhang, Y., Huang, X., Zhu, D., Cai, H., Liu, T., & Li, X. (2023). Differentiate ChatGPT-generated and human-written medical texts. *arXiv preprint arXiv:2304.11567*.

[14] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[15] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems* (pp. 3111–3119).

[16] Mindner, L., Schlippe, T., & Schaaff, K. (2023). Classification of human- and AI-generated texts: Investigating features for ChatGPT. In *International conference on artificial intelligence in education technology* (pp. 152-170). Singapore: Springer Nature Singapore.

[17] Nguyen-Son, H.-Q., & Echizen, I. (2018). Detecting computer-generated text using fluency and noise features. In K. Hasida & W. Pa (Eds.), *Computational Linguistics. PACLING 2017. Communications in Computer and Information Science* (Vol. 781, pp. 315–328). Springer.

[18] Oghaz, M., Dhame, K., Singaram, G., & Saheer, L. (2023). Detection and classification of ChatGPT generated contents using deep transformer models. *Frontiers in Artificial Intelligence*, *8*, 1458707.

Research Article

[19] Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pretraining. *OpenAI*.
[20] Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). SQuAD: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.
[21] Shijaku, R., & Canhasi, E. (2023). ChatGPT generated text detection. *ResearchGate*.
[22] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems* (pp. 5998–6008).
[23] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., & Le, Q. V. (2019). XLNet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.