MetaHate: Text-based Hate Speech Detection for Metaverse Applications using Deep Learning

Judith Nkechinyere Njoku *, Anthony Uchenna Eneh †, Cosmas Ifeanyi Nwakanma‡,
Jae-Min Lee *, and Dong-Seong Kim*
*IT Convergence Engineering, ‡ICT Convergence Research Center, Kumoh National Institute of Technology, Korea
†Frontend Engineering Team, Tech and Product Department, Africhange Technologies Ltd, Nigeria

***Abstract*—The rise of digital communication and the metaverse has revolutionized interaction paradigms, while also introducing challenges in ensuring safe engagements. Hate speech, pervasive in digital spaces, threatens inclusivity. This research introduces a tailored hate speech detection system for the metaverse. Through a comprehensive evaluation of deep learning models, effective real-time detection approaches are identified. To guarantee reliable deployment in the metaverse, models are subjected to explainability assessments using Local Interpretable Model-Agnostic Explanations (LIME). A lightweight Convolutional Neural Network (CNN) model is developed and deployed on the Roblox server, exhibiting commendable accuracy and efficiency. Remarkably, quantization reduces the CNN model size by 93.59%. This pioneering study addresses the dearth of metaverse-focused hate speech research, fostering secure and inclusive virtual spaces.**

***Index Terms*—hate speech, metaverse, deep learning, offensive language, GloVe, explainableai**

I. Introduction

The rise of digital communication and virtual platforms has transformed how people interact and share information. With the advent of the metaverse, an interconnected virtual space, collaboration and engagement possibilities have reached unprecedented levels [1], [2]. However, this advancement has also brought challenges regarding safe interactions in immersive digital realms. Social media allows free expression, but this has led to the spread of harmful and abusive content [3].

Hate speech, involving derogatory language and discriminatory remarks based on ethnicity, gender, race, etc., poses a threat to online inclusivity [4]. Metaverse platforms face similar challenges, with incidents of harassment, assault, and hate speech reported [5]. Even prominent platforms like *Meta* suffer from issues like virtual stalking and lack of moderation [6]. Despite the importance, limited work addresses hate speech in the metaverse. Some studies focus on audio-based detection [7], while others explore the impact without solutions [8]. This reveals a gap in protecting metaverse users from abuse.

Developing text-based hate speech detection models requires choosing appropriate deep-learning models and word embeddings. Various models like convolutional neural networks (CNN), and multi-layered perceptron (MLP) have been used [8] and have shown various levels of robustness. In the context of the metaverse, deploying trustworthy AI models is paramount. To ensure user safety and well-being, the deployed models must not only be accurate but also transparent and interpretable. Users need to understand how decisions are made to establish trust in the AI's actions.

This study pioneers hate speech detection in the metaverse, offering:

- Comprehensive evaluation of deep learning models for metaverse hate speech detection.
- Creation of a lightweight model suitable for the metaverse.
- Deployment of the model for real-time testing on the Roblox server, emphasizing transparency and interpretability for user trust.

This model not only detects hate speech effectively but also provides insights into its decision-making process, fostering user trust.

The rest of this paper is structured as follows: Section II summarizes foundational literature, Section III presents the dataset and methodologies, Section IV showcases experimental results, and Section V concludes the paper.

II. Background & Literature Review

This section provides some background for this study.

A. Hate Speech in the Metaverse

The metaverse's immersive nature transforms communication [1]. Users engage through avatars in simulated environments [2], magnifying the impact of interactions. This unique blend of reality and virtuality necessitates appropriate behavior. Hate speech's emergence in digital spaces extends to the metaverse, threatening inclusive communities. Prominent platforms like *Meta* are not immune, facing virtual stalking, assault, and content moderation challenges [6]. Ensuring user safety and well-being in these immersive environments becomes paramount.

B. Trustworthy AI Models in the Metaverse

In the context of the metaverse, where the fusion of real and virtual experiences is prominent, the significance of explainable AI is amplified, particularly when dealing with sensitive matters such as hate speech. Emphasizing the development of ethical and interpretable AI models becomes essential to establish fairness and engender trust [9]. Among the techniques, Local Interpretable Model-Agnostic Explanations (LIME) stands

out by offering insights into AI decision-making processes, thereby enhancing comprehension and facilitating positive interactions [10].

C. Word Embeddings

Word embeddings are vital for representing words as continuous vectors, capturing semantic connections [11]. In hate speech models, two types stand out: (i) pre-trained embeddings (e.g., Word2Vec, GloVe) for strong basics, and (ii) contextual embeddings (e.g., BERT) for nuanced context understanding [3], [12]. Experiments in [3] favor GloVe's efficiency with specific DL models, while [13] highlights BERT's robustness. However, BERT's fine-tuning, preprocessing, and resource demands [13] make it less computationally efficient than GloVe, posing metaverse platform challenges [13].

D. Deep Learning Models

Hate speech detection employing text classification tasks has been effectively tackled by DL models [14]. DL models, including CNN and MLP, are explored for text-based hate speech detection in [3], with favorable accuracy and F1 score. CNN outperformed the long-short-term-memory (LSTM) model in [13]. [4] considers emojis and hashtags, training an MLP for hate speech detection. DL models, especially CNNs and MLPs, efficiently detect text-based hate speech [15], [16]. Computational expense hampers models like Bi-directional Gated recurrent Units (BiGRU) and BiLSTM [17], focusing on less costly CNN, MLP, or their variants [18].

III. Methodology

This section presents details of the system model illustrated in Fig. 1.

A. Problem Definition

Let $\mathcal{T} = \{t_1, t_2, t_3, \ldots t_n\}$ be the set of texts and $\mathcal{L} = \{l_1, l_2, l_3\}$ be the corresponding $\mathbf{n}$ labels for the input sample $\mathcal{T}$, where $\mathcal{T} \in \{\text{Hate, Offensive, Normal}\}$ represents the presence and absence of hate and offensive speech. The aim of the models is to predict the conditional label $\mathbf{l}$ for the given input sample $\mathbf{s}$, as $P(l|s)$.

B. Dataset Description and Preprocessing

The dataset used in the experiment is the Davidson dataset and was obtained from [19]. The dataset was obtained from tweets and was categorized into hate speech, offensive language, and normal speech. The statistics of this dataset are described in Table I. The dataset was preprocessed by tokenizing using a Tensorflow-based word tokenizer. The tokenized sequences were then padded to have the same length as the longest sequence. GloVe pre-trained word embeddings composed of 100-dimensional vectors were used to create an embedding matrix for the words in the dataset. The dataset was then split into train, test, and validation sets, using a split of $75:15:10$.

TABLE I
STATISTICS OF DATASET

Speech category	Hate	Offensive	Normal	Total
Number	1430	19190	4163	24783
Percentage (%)	5.77	77.43	16.80	100

C. Models

The following models from the literature were explored in this paper.

1) Convolutional neural network (CNN): The CNN model employed in this study was adapted from [3] and includes two Conv1D layers with 32 and 64 filters, followed by GlobalMaxPooling1D for feature extraction. A dense layer with ReLU activation aids in learning complex patterns. Dropout mitigates overfitting, and a final softmax layer provides class probabilities. Compiled with sparse categorical cross-entropy loss and Adam optimizer, the architecture balances convolutional and dense operations, allowing the model to capture diverse features for effective hate speech classification. The model architecture is illustrated in Fig. 2.

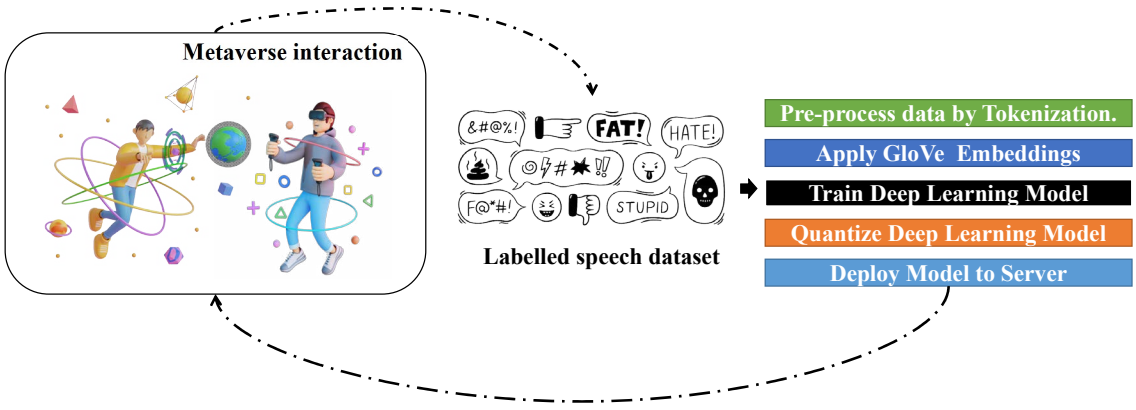


Fig. 1. System model showing the proposed hate speech detection approach

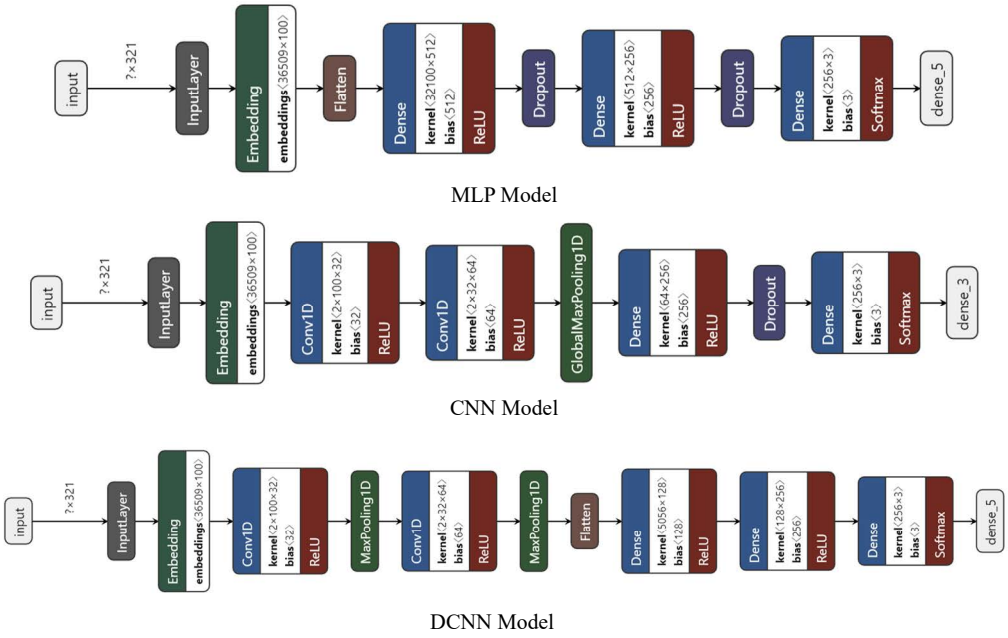


Fig. 2. The Models employed

2) Deep Convolutional neural network (DCNN): This model employs pre-trained GloVe word embeddings in an embedding layer. Conv1D layers follow, with 32 and 64 filters, kernel size 2, and ReLU activation. MaxPooling1D layers retain patterns and reduce dimensionality. A Flatten layer reshapes data for dense layers. Two dense layers with 128 and 256 units, using ReLU activation, capture intricate relationships. The final dense layer with softmax activation produces class probabilities for hate speech, offensive language, and non-offensive content. Compiled with sparse categorical cross-entropy loss and the Adam optimizer, this architecture detects patterns at various levels. See Fig. 2 for illustration.

3) Multi-layered perceptron: Starting with pre-trained GloVe word embeddings, the model transforms input text into vectors. These vectors are then flattened to connect with dense layers. The model includes two dense layers (512 and 256 units) with ReLU activation. Dropout layers (0.2 rate) follow each dense layer to reduce overfitting by deactivating neurons during training. The final dense layer uses softmax activation for class probabilities (hate speech, offensive language, non-offensive). Using sparse categorical cross-entropy and the Adam optimizer, this architecture captures text relationships. The structure of embedding, dense, and dropout layers facilitates learning patterns, promising for hate speech detection.

D. LIME Explainability for Hate Speech Detection

Local Interpretable Model-agnostic Explanations (LIME) enhance model interpretability by creating simpler, faithful models around individual predictions. For text-based tasks, LIME perturbs input text and observes prediction changes to provide insights into complex model decisions. In this study, LIME explains the model predictions by perturbing text instances. This identifies influential words or phrases, improving transparency and trust in the system.

E. Model Quantization

Quantization maps continuous values to discrete ones, reducing bits for neural network weights and activations. This conserves memory and computation while retaining accuracy. It can happen during or after training. Here, TensorFlow's post-training quantization was used, optimizing the model for inference on resource-limited devices. The model was first converted to TensorFlow Lite format before quantization. The following quantization equation governs the process:

$$QW = round(W/S), \tag{1}$$

where S is the scaling factor that maps floating-point values to the quantized integer range, and ensures that the range of floating-point values is mapped effectively to the quantized range. W is a given floating-point weight value. $round()$ rounds the scaled value to the nearest integer.

During post-training weight quantization, the scaling factor can be calculated as:

$$S_{weight} = \frac{Max_{abs}}{(2^{B-1} - 1)}, \tag{2}$$

Where Max_{abs} represents the maximum absolute value of weights in the layer, and the denominator term $(2^{B-1} - 1)$ is used to ensure that the quantized value can represent the range [-1, 1].

For post-training activation quantization, the scaling factor can be calculated as:

$$S_{act} = \frac{Max_{abs-a}}{(2^{B-1} - 1)}, \tag{3}$$

Where Max_{abs-a} represents the maximum absolute value of activations in the layer, and the denominator term $(2^{B-1} - 1)$ is used to ensure that the quantized value can represent the range [-1, 1].

F. Model Deployment

The quantized model is integrated into a web server, facilitating real-time analysis of text messages exchanged between players. Emoji icons are employed to visually indicate predicted speech types: angry emojis for hate/offensive speech and happy emojis for normal speech. The Roblox API integration connects the web server to the metaverse environment, enabling seamless communication. Utilizing the HTTP Service, POST requests are sent from Roblox to the server, transmitting input for hate speech detection. The Chat Service is employed to filter and control messages based on model predictions, ensuring a safe environment.

G. Model Performance and Explainability Metrics

This section evaluates models using the performance metrics and explainability metrics introduced below.

1) FLOPS (Floating-Point Operations per Second): FLOPS quantifies computational efficiency by measuring the rate of arithmetic operations involving floating-point numbers. Higher FLOPS values indicate better processing capability.

2) Memory Usage: Memory usage indicates a model's memory footprint during execution. It assesses memory consumption for inference in resource-constrained environments.

3) Model Parameters: Model parameters reflect the complexity and the ability to learn patterns. Balancing parameter count with generalization ability is crucial, preventing overfitting and computational overhead.

4) F1 Score: The F1 score evaluates classification performance by considering precision and recall balance. Computed as:

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

5) Computation Time: Computation time measures inference speed, reflecting input processing and prediction generation duration. Lower times benefit real-time applications, meeting latency requirements in contexts like the metaverse.

6) Proximity Score: The proximity score evaluates the fidelity of an explanation by measuring how closely it approximates model behavior. Calculated using the proximity measure:

$$\text{Proximity Score} = 1 - \frac{d_e(x)}{d_{\max}} \tag{5}$$

where $d_e(x)$ represents the explanation's distance to the decision boundary, and $d_{\max}$ is the maximum possible distance.

7) R-squared (Coefficient of Determination): R-squared assesses the proportion of variance in the output explained by the model's inputs:

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2} \tag{6}$$

where y_i is the observed output, $\hat{y}_i$ is the predicted output, and $\bar{y}$ is the mean of the observed outputs.

8) Prediction Difference: Prediction difference quantifies the impact of perturbing instance features on the model's prediction for that instance:

$$\text{Prediction Difference} = \hat{y}\text{perturbed} - \hat{y}\text{original} \tag{7}$$

where $\hat{y}$perturbed is the prediction after feature perturbation, and $\hat{y}$original is the original prediction.

IV. Experimental Setup and Results

All model training was conducted in the Google Colaboratory platform, using different device specifications, including CPU, GPU, and TPU. This is to enable a thorough evaluation of all models. Table II presents a comprehensive comparison of various models based on key performance metrics: FLOPS, memory usage, and model parameters. The table provides insights into the computational efficiency, memory footprint, and complexity of each model. Among the models, the MLP exhibits the highest computational efficiency, surpassing both the CNN and the DCNN with a value of 0.03313 GFLOPS.

However, it's important to note that the memory usage values in the table are higher for the MLP compared to the other models. The MLP employs 77.128 bytes of memory, whereas the CNN and DCNN models demand 14.03 bytes and 16.56 bytes, respectively. This indicates that the MLP consumes more memory than the other models. The MLP, while excelling in computational efficiency, exhibits greater complexity with 20,218,711 parameters. In contrast, both the CNN and DCNN models display reduced parameter counts of 3,678,903 and 4,342,583, respectively.

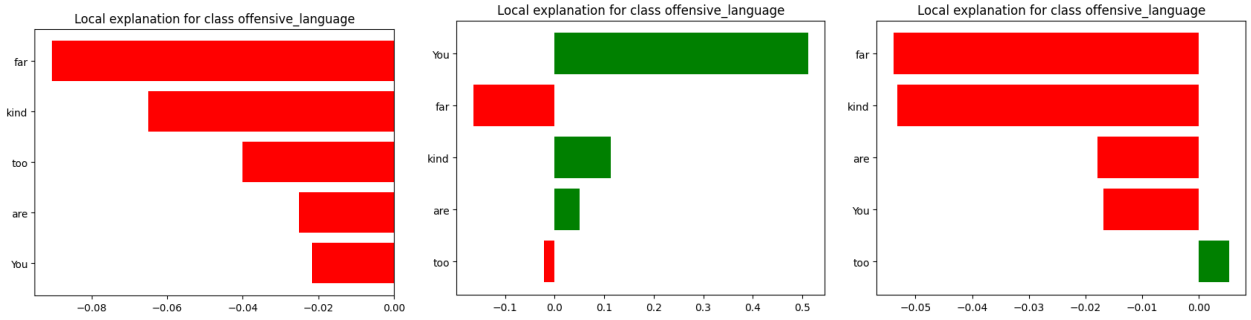
Table III unveils a comprehensive comparison of three models' performance metrics under varying processors: CPU, GPU, and TPU. This analysis critically evaluates the models' effectiveness in terms of F1 Score, Recall, Precision, and Time (in seconds) across these processing units. Notably, the MLP consistently achieves competitive F1 Scores across all processors, indicating its robust predictive capabilities. In contrast, the CNN and DCNN models exhibit varying F1 Scores under different processors, suggesting their sensitivity to processing unit selection. While the MLP showcases consistent inference times across processors, the CNN and DCNN

TABLE II
Model Comparison Metrics

Model/Metrics	FLOPS (GFLOPS)	Memory Usage (bytes)	Model Parameters
MLP	0.03313	77.128	20,218,711
CNN	0.00368	14.03	3,678,903
DCNN	0.0068	16.56	4,342,583

TABLE III
MODEL PERFORMANCE COMPARISON

Processor	CPU				GPU				TPU			
Model/Metrics	**F1 Score**	**Recall**	**Precision**	**Time (s)**	**F1 Score**	**Recall**	**Precision**	**Time (s)**	**F1 score**	**Recall**	**Precision**	**Time (s)**
MLP	0.84	0.66	0.63	8.755	0.85	0.64	0.65	0.602	0.85	0.66	0.65	10.49
CNN	0.73	0.73	0.64	5.45	0.85	0.71	0.66	0.685	0.81	0.69	0.66	14.03
DCNN	0.85	0.69	0.66	5.17	0.84	0.70	0.66	1.50	0.82	0.68	0.66	2.11



(a) CNN (b) MLP (c) DCNN

Fig. 3. Performance of LIME Explainability of all model predictions

TABLE IV
COMPARISON OF EXPLAINABILITY RESULTS OF THE DIFFERENT MODELS

Model/Metrics	Proximity Score	R-Squared	Prediction Diff
MLP	0.7216	0.5207	1
CNN	0.6997	0.4896	0
DCNN	0.2972	0.0883	0

models exhibit differential inference performance, influenced by the processing unit's capabilities.

Table IV and Fig. 3 illustrate the explainability of all models. Explainability was conducted on a sample text *"You are far too kind"* and while the CNN and DCNN models were quite accurate in explaining the predictions, the MLP model was quite erroneous and the explanation too. Hence the CNN model was selected as the most robust and trustworthy model to be deployed to the metaverse.

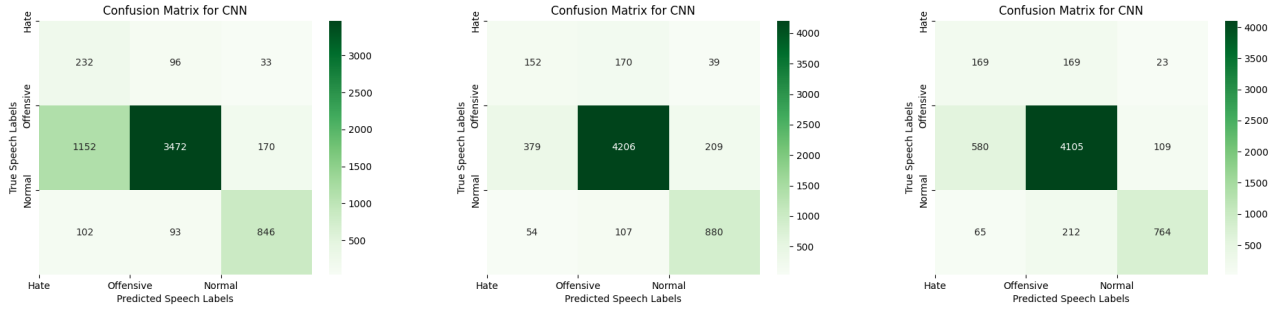
Fig. 4 represents the confusion matrixes of the CNN models on the different processors, showing the robustness of the model in accurately classifying hate speech.

The CNN model was selected for further processing and to be deployed to the web server. Thus it was quantized and deployed to a web server hosted on Replit [20]. By quantizing the CNN model, the size reduces from a whooping 36.981 MB to 2.37 MB. Fig. 5 shows the interface of the web server [1] and the Roblox platform for hate speech detection.

V. CONCLUSIONS AND FUTURE WORKS

This paper showcases a successful hate speech detection model in Roblox's metaverse. The quantized explainable CNN model, trained for efficient inference, is integrated into a web

[1]https://metaverse-va.judithnjoku.repl.co/



(a) CPU (b) GPU (c) TPU

Fig. 4. CNN Model Performance on Different Processors

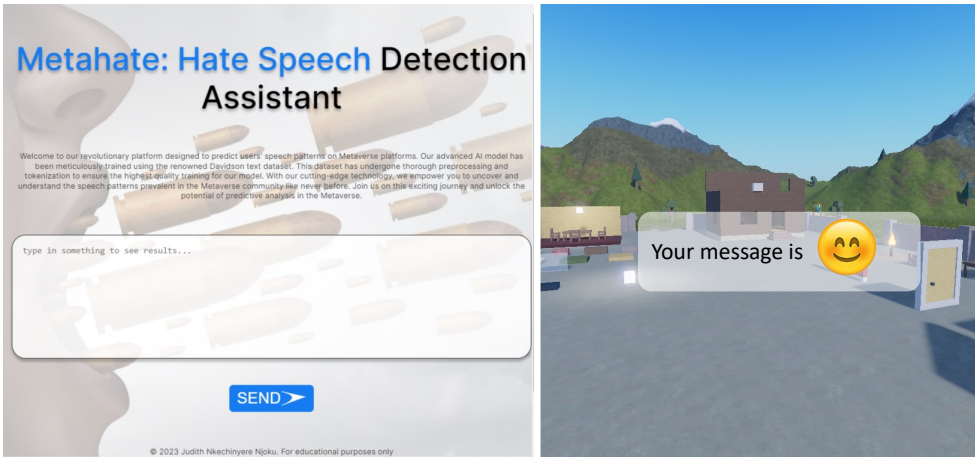


Fig. 5. Sample Demo of Server and Roblox interface

server. The server, connected to the Roblox API, enables real-time text analysis between players. Our approach uses HTTP for POST requests and Chat Service for filtering, ensuring user safety. Integrating quantized models into interactive platforms like Roblox highlights AI's potential for virtual interactions. Future work aims to develop better models for hate speech detection in the metaverse.

ACKNOWLEDGMENTS

This research was supported by the Priority Research Centers Program through the NRF funded by the MEST (2018R1A6A1A03024003), MSIT Korea (1711175292/2022-IT-RD-0084-01), and by MSIT under the Innovative Human Resource Development for Local Intellectualization support program (IITP-2023-2020-0-01612) supervised by the IITP.

REFERENCES

[1] J. N. Njoku, C. I. Nwakanma, G. C. Amaizu, and D.-S. Kim, "Prospects and challenges of metaverse application in data-driven intelligent transportation systems," *IET Intelligent Transport Systems*, vol. 17, no. 1, pp. 1–21, 2023.

[2] C. I. Nwakanma, J. N. Njoku, J. Jo, C. Lim, and D. Kim, ""creativia" metaverse platform for exhibition experience," in *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*, 2022, pp. 1789–1793.

[3] J. S. Malik, G. Pang, and A. van den Hengel, "Deep learning for hate speech detection: A comparative study," 2022.

[4] S. G. Roy, U. Narayan, T. Raha, Z. Abid, and V. Varma, "Leveraging multilingual transformers for hate speech detection," 2021.

[5] F. Sheera and K. Browning, "The metaverse's dark side: Here come harassment and assaults," *The New York Times*, August 2021, (Accessed August 22, 2023). [Online]. Available: https://www.nytimes.com/2021/12/30/technology/metaverse-harassment-assaults.html)

[6] K. Miller, "Report: The metaverse is already plagued by hate speech and sexual assault," *InsideHook*, June 2022, (Accessed August 22, 2023). [Online]. Available: https://www.insidehook.com/daily_brief/tech/metaverse-report-hate-speech-assault)

[7] R. M. Medina, J. N. Njoku, and D.-S. Kim, "Audio-based hate speech detection for the metaverse using cnn," in *Korean Institute of Communications and Information Sciences (KICS Fall 2022), Gyeongju, South Korea*, 2022.

[8] I. Trauthig and S. Woolley, "Addressing hateful and misleading content in the metaverse," *Journal of Online Trust and Safety*, vol. 1, no. 5, Apr. 2023. [Online]. Available: https://tsjournal.org/index.php/jots/article/view/109

[9] S. Glen, "Why we can't trust ai to run the metaverse," *Data Science Central*, February 2022, (Accessed August 22, 2023). [Online]. Available: https://www.datasciencecentral.com/why-we-cant-trust-ai-to-run-the-metaverse/

[10] C. I. Nwakanma, L. A. C. Ahakonye, J. N. Njoku, J. C. Odirichukwu, S. A. Okolie, C. Uzondu, C. C. Ndubuisi Nweke, and D.-S. Kim, "Explainable artificial intelligence (xai) for intrusion detection and mitigation in intelligent connected vehicles: A review," *Applied Sciences*, vol. 13, no. 3, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/3/1252

[11] H. S. Alatawi, A. M. Alhothali, and K. M. Moria, "Detecting white supremacist hate speech using domain specific word embedding with deep learning and bert," *IEEE Access*, vol. 9, pp. 106 363–106 374, 2021.

[12] N. Prasad, S. Saha, and P. Bhattacharyya, "A multimodal classification of noisy hate speech using character level embedding and attention," in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1–8.

[13] Y. Yadav, P. Bajaj, R. K. Gupta, and R. Sinha, "A comparative study of deep learning methods for hate speech and offensive language detection in textual data," in *2021 IEEE 18th India Council International Conference (INDICON)*, 2021, pp. 1–6.

[14] N. Ousidhoum, Z. Lin, H. Zhang, Y. Song, and D.-Y. Yeung, "Multilingual and multi-aspect hate speech analysis," 2019.

[15] S. Banerjee, B. Raja Chakravarthi, and J. P. McCrae, "Comparison of pretrained embeddings to identify hate speech in indian code-mixed text," in *2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, 2020, pp. 21–25.

[16] S. Biradar, S. Saumya, and A. Chauhan, "Hate or non-hate: Translation based hate speech identification in code-mixed hinglish data set," in *2021 IEEE International Conference on Big Data (Big Data)*, 2021, pp. 2470–2475.

[17] M. Mozafari, R. Farahbakhsh, and N. Crespi, "Cross-lingual few-shot hate speech and offensive language detection using meta learning," *IEEE Access*, vol. 10, pp. 14 880–14 896, 2022.

[18] H. Sohn and H. Lee, "Mc-bert4hate: Hate speech detection using multi-channel bert for different languages and translations," in *2019 International Conference on Data Mining Workshops (ICDMW)*, 2019, pp. 551–559.

[19] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language," *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 11, no. 1, pp. 512–515, May 2017. [Online]. Available: https://ojs.aaai.org/index.php/ICWSM/article/view/14955

[20] "Replit," https://replit.com, accessed: Aug 22, 2023.