A Comparative study of BERT-CNN and GCN for Hate Speech Detection

SHASANK SEKHAR PANDEY, University of Twente, The Netherlands

Social media has become not only a medium for like-minded people to connect but also a platform where anyone can freely express their thoughts and opinions. However, its widespread nature has not only led to an immeasurable impact on society but has also presented some important challenges. Online hate speech is one such challenge. Consequently, the identification of hate speech on online platforms has gained much traction recently. Different methods ranging from reactive approaches like using Natural Language Processing (NLP) for classifying individual posts to proactive approaches like using contextual information and predicting when a discussion advances towards hatefulness have been tried in the domain of Hate Speech Detection. In this paper, we perform an in-depth comparison of two such techniques of Hate Speech Classification, namely BERT-CNN and Graph-based Graph Convolutional Network (GCN). Our findings show that when developed on the same dataset from Twitter, the BERT-CNN model requires fewer computational resources compared to the GCN model. Moreover, the BERT-CNN model achieves a macro F1 score of 0.81 outperforming the GCN model with a macro F1 score of 0.48.

Additional Key Words and Phrases: Hate Speech, Natural Language Processing, Graph Convolutional Network, Graph Neural Network, BERT, Convolutional Neural Network, Twitter.

1 INTRODUCTION

The use of social media has grown tremendously, with about 59% of the world using it daily for an average of 2 hours and 31 minutes [13]. This use of social media has produced both positive and negative impacts on society. One such negative impact is the publishing of hateful comments i.e., comments targeted at individuals or groups based on ethnicity, national background, gender identity, sexual orientation, societal class, or disability on social media platforms [16].

These hateful comments, commonly known as "hate speech" have been shown to have substantial negative effects on victims' mental health. For example, in a survey focused on understanding the impact of online and offline hate speech on the LGBTQ+ community in Ukraine and Moldova, it has been shown that hate speech can cause emotional distress, depression, sleep disturbances, exhaustion, panic attacks, and feelings of social isolation [26]. These ill effects of hate speech have motivated social media platforms to deploy automated and manual detection and moderation mechanisms to prevent further harm and limit hatefulness on their platforms.

This domain of hate speech detection has also gained a lot of attention from researchers, who have experimented with different methods like Natural Language Processing (NLP) or Deep Learning Models (DLMs) such as Support Vector Machines (SVMs), Random Forests, Deep Neural Networks (DNNs) [3] for identifying abusive and offensive content.

TScIT 39, July 7, 2023, Enschede, The Netherlands

Moreover, Graph Neural Networks (GNNs), another type of DLM, have recently received wide attention in the domain of text classification because of their effectiveness at classification tasks thought to have rich relational structures [23]. In this paper, we have taken up the task of comparing BERT-based Convolutional Neural Network (BERT-CNN) and Graph Convolutional Network (GCN), from the domains of NLP and DLM respectively.

Both these models have been used for text classification; however, their designs are starkly different. BERT-CNN uses the pre-trained BERT model to obtain the vector representation of words, extracting features like sentence sequence [12], which is used as input to the CNN model that extracts high-level features and performs the classification. GCN on the other hand, relies on the relationships between words and documents they occur in, along with global relationships between words, converting the text-classification problem into a node-classification problem [23].

Studies have been conducted on the usability of these models in the Hate Speech Classification domain, but due to the differences in the training dataset, we believe, research is required in comparing the two methods and their capabilities. Therefore, this research provides both a detailed understanding and a comparison of the two methods.

2 PROBLEM STATEMENT

Although there has been prior research [4, 14, 16, 18, 19] in developing different models and techniques for the analysis of hate speech, the domains of NLP-based approaches and Graph-Based DLM approaches have rarely been compared in similar contexts. To address this, we have developed the following research questions to better understand and answer this problem.

RQ 1: What are the differences in computational resources required for the creation and training of BERT-CNN and GCN?

RQ 2: What is the difference in performance between BERT-CNN and GCN when trained on the same dataset?

By answering these research questions, we aim to provide not only a comparison between the performance of the two models but also a comparison of computational resources required to obtain that level of performance.

3 BACKGROUND

3.1 BERT

BERT stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers and was developed by scientists from Google [7]. BERT makes use of an attention mechanism called a Transformer that learns contextual relations between words in a text. It allows for deep preliminary learning of bidirectional text representation for subsequent use in machine learning models [15]. For our research, we aim to use this bi-directional ability of BERT to extract contextual information [1], before passing it to the CNN for classification.

3.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) were first introduced as a mechanism of visual pattern recognition [9], but since have been used in various application areas, including but not limited to, Activity Recognition, Text Recognition, Face Recognition, and Natural Language processing [8].

The basic design of a CNN consists of an input layer, an output layer, and multiple hidden layers that may or may not include convolutional layers, pooling layers, fully-connected layers, and various normalization layers [8]. For our research, we aim to use CNN to learn features from word vectors produced using BERT and classify them.

3.3 Graph Convolutional Networks

Graph Convolutional Networks (GCNs), are a special type of Graph Neural Networks [24], based on Convolutional Neural Networks on graphs [22]. They have achieved state-of-the-art results in various application areas such as Natural language processing, applied chemistry, computer vision, and citation networks [22].

GCNs allow for the exploitation of non-Euclidean characteristics i.e. the irregular structure of graphs [24] and learn node features by aggregating information in its neighbourhood [17].

For our research, we aim to convert the dataset into one graph and use GCN's ability to preserve global information structure and learn from rich relational structure [4] to classify tweets.

4 CLASSIFICATION MECHANISM OF BERT-CNN

BERT is trained on plain text for masked word and next-sentence prediction tasks [7]. Therefore, to apply the capabilities of BERT for the task of text classification, it must be fine-tuned using task-specific training data [25]. Furthermore, additional task-specific layers can be applied in combination with the pre-trained BERT model to further improve its capabilities [25].

Using a CNN in combination with BERT allows for obtaining local information in the text more effectively [25].

In the framework of BERT-CNN shown in Figure 1, there are 6 layers: BERT-embedding layer, Convolutional layer(s), Pooling layer, Dense layer, Dropout layer and Output layer. The BERT model layer is used to convert input text i.e., a tweet into word vectors and to create a primary input matrix.

This input matrix is fed to the convolutional layers, which create feature maps which are converted to max-value feature vectors using the pooling layer. The results from the pooling layer are passed onto the fully connected dense layer for dimensionality reduction. We also utilize a dropout layer to reduce overfitting by dropping the forward and backward connections of certain neurons, thus preventing co-adaptation. Lastly, a final fully connected output layer is used for classification.

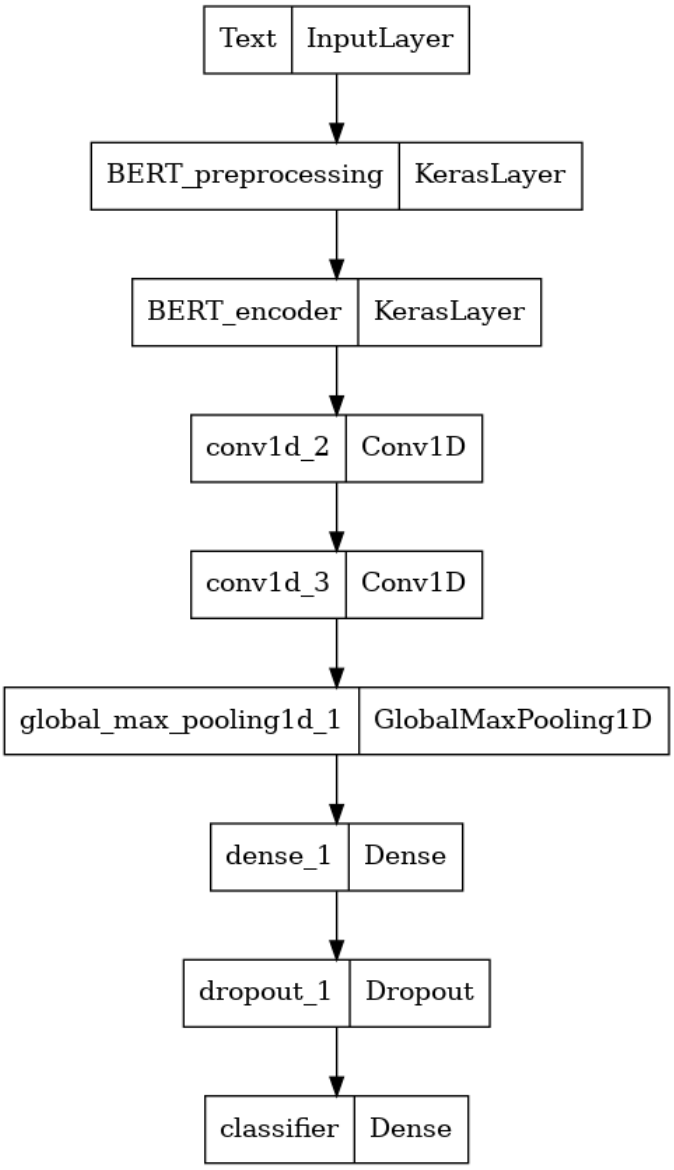

Fig. 1. BERT-CNN Model Structure

5 CLASSIFICATION MECHANISM OF GCN

As mentioned in Section 3.3, GCNs are a special type of CNN that allow for the exploitation of irregular structure of graphs.

The layers in a GCN are Graphs containing nodes whose features are learned based on their local neighbourhood. This mechanism of GCNs learning attributes of nodes based on their neighbourhood, allows us to use a Graph containing tweets and words and learn the hatefulness of words and tweets based on their relationships to one another. This type of GCNs, where features of nodes are learnt based on their local neighbourhood are also referred to as Spatial GCNs.

For this study, we design a 2-layer GCN as described in [23]. The two-layer structure allows message passing among nodes that are at maximum 2 steps away [23]. Therefore, in our design where we only have tweet-word and word-word edges, relationships between tweet-tweets can be learnt because of this 2-layer structure.

6 RELATED WORK

6.1 Hate Speech Detection

Hate speech detection has gained traction in the research community as it has far-reaching impacts on society. Various ways of performing it have been explored. The two main categories of hate speech detection include the use of lexicons [10] and the use of machine learning [2]. The machine learning approach relies on the extraction of features from text such as Term-Frequency Inverse-Document-Frequency (TF-IDF) or Bag of word vectors [2], with some utilizing network and user features to determine hate and abuse [5, 20].

The lexicon-based approach utilizes the domain of sentiment analysis to determine the polarity of text and combine it with features of hate speech to create robust classifiers for hate speech.

6.2 Hate Speech Detection Using BERT

BERT provides a Transfer learning approach to hate speech detection, as it can be fine-tuned and applied in combination with other deep learning models for hate speech detection [18].

This transfer-learning approach has been utilized by various researchers. The authors in [14] have finetuned BERT with Masked Rationale Prediction (MRP) to increase the model's explainability and have obtained a macro F1 score of 0.699. The DictNN solution in [16] combined BERT with a 3-layer CNN along with a dictionary approach in the preprocessing stage, to obtain a macro F1 score of 0.61.

Lastly, in [18], the authors tried different combinations of BERT and DLMS to create models such as BERT + Non-linear layers with an F1-score of 0.92, BERT + LSTM with an F1 score of 0.88 and BERT+CNN with an F1 score of 0.92.

6.3 Hate Speech Detection Using Graph Convolutional Networks (GCN)

Conversational hate speech has a deeply contextual nature, requiring an understanding of both the text and its context for proper classification [11]. GCNs allow for the capturing of this contextual information [19], which is why they have been gaining traction in the domain of hate speech detection [11].

The authors in [11], have used GCNs to create a framework to detect hate speech and predict if conversations are steering towards hate speech by taking into account the conversational history of comments. In [4], the authors created a simple GCN to detect hate speech on Twitter, by converting tweets into a graph of words and tweets, allowing for the capture of word co-occurrence relations, and achieved an F1 score of 0.8215. Lastly, the authors of [19], used both the context of tweets and user relations on Twitter, to model a GCN with features such as retweet count and follower-followee relations between users.

7 METHODOLOGY

7.1 Pre-processing

Since the training dataset is sourced from Twitter, and as tweets contain unstructured information like special characters, punctuation marks, Twitter usernames, and more, normalizing the information is important [4].

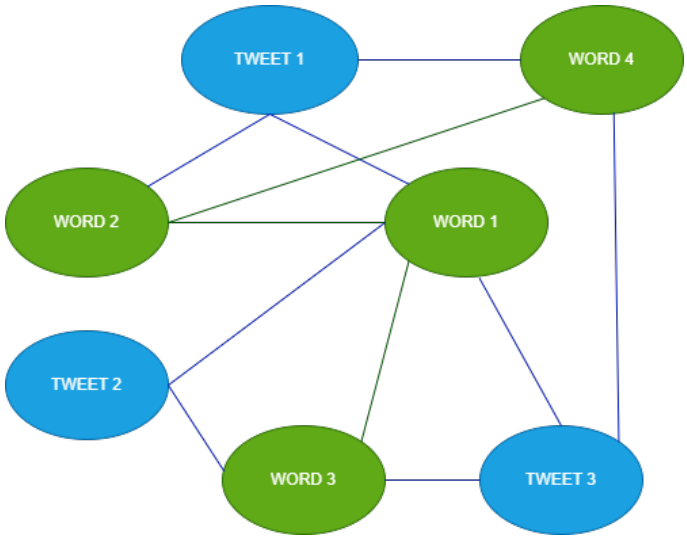

Fig. 2. Graph Structure [4]

For normalization, we use the Ekphrasis[1] Library as it allows for pre-processing of text from social networks and performs functions such as tokenization, word normalization, word segmentation, and spell correction.

7.2 Models

7.2.1 Graph Convolutional Network. For the GCN we followed the structure presented in [4], under which the graph G = (N, E, W), as shown in Figure 2, has the following properties:

(1) Nodes (N): The graph consists of word and tweet nodes, with the total number of nodes in the graph equal to the number of tweets + total number of unique words.
(2) Edges (E): The graph consists of 2 types of edges, tweet-word edges created using word occurrence in tweets and word-word edges created using word co-occurrence.
(3) Weights (W): The weight of a tweet-word edge is determined using the term frequency-inverse document frequency (TF-IDF) of the word in the tweet, where TF is the frequency of the word in the tweet and IDF is the logarithmically scaled inverse fraction of the number of tweets containing the word [23]. For a word-word edge, Point-wise Mutual Information (PMI) values are used, which are calculated using a fixed sliding-window approach on all documents, allowing the capture of global word co-occurrence [23].

The graph is used in a simple two-layer GCN, where the second-layer node embeddings are of the same size as the output label set and are fed into a SoftMax classifier [23].

7.2.2 BERT-CNN. This model is designed with 2 parts, the first being the pre-trained BERT base model used for the conversion of words in the tweet into contextualized vector representations [21].

[1] ekphrasis · PyPI Last visited 24/04/2023.

The second part is the CNN layer, which is used as a classifier. There have been different methods of designing the CNN classifier, from using 3 convolutional layers with increasing output channels [16], to using 4 parallel convolutional filters of different sizes[21]. However, due to the increased complexity of using parallel convolutional filters, we chose to use the structure described in [16], but with only 2 convolutional layers.

8 EXPERIMENT

8.1 Training Dataset

The dataset used for training was created by the authors of [6], containing 24,783 Tweets with their distributions shown in Table 2. Each tweet in the dataset was manually coded by 3 or more people, using the CrowdFlower platform under strict criterions [6].

To train the models, we divided the dataset into 90% Train and 10% Test splits. We used a larger train dataset to counter the imbalanced nature of the dataset, allowing for more samples of each class to be available during training. Moreover, we stratified the splits based on classes to ensure the availability of all classes in both datasets.

Table 1. Distribution of classes in Dataset

Label	Class	No. of Instances
Hate Speech	0	1430
Offensive	1	19190
Neither	2	4163

8.2 Pre-processing

We normalized each tweet to remove usernames, and URLs and correct spelling errors as can be seen in Table 2, using the Ekphrasis library.

Table 2. Tweet before and after pre-processing

Tweet	!!! RT @mayasolovely: As a woman you shouldn't complain about cleaning up your house. & as a man you should always take the trash out...
Processed Tweet	! <repeated> rt <user>: as a woman you should not complain about cleaning up your house. & as a man you should always take the trash out. <repeated>

8.3 GCN Model

8.3.1 Graph. Using the method described in Section 7.2.1, we created a set of nodes, containing 22,304 tweet-word nodes and 25,854 word-word nodes. We used the SK-learn's vectorizer to construct a unique vocabulary from the tweets and calculate the TF-IDF scores between each word in the vocabulary and all tweets.

For the calculation of PMI-values, used to determine word-word edges, we used a sliding window of size 10 as it was the largest sliding window that could be accommodated on the available hardware. The final graph as shown in figure 3, consisted of 48,158 nodes and 1,186,730 edges.

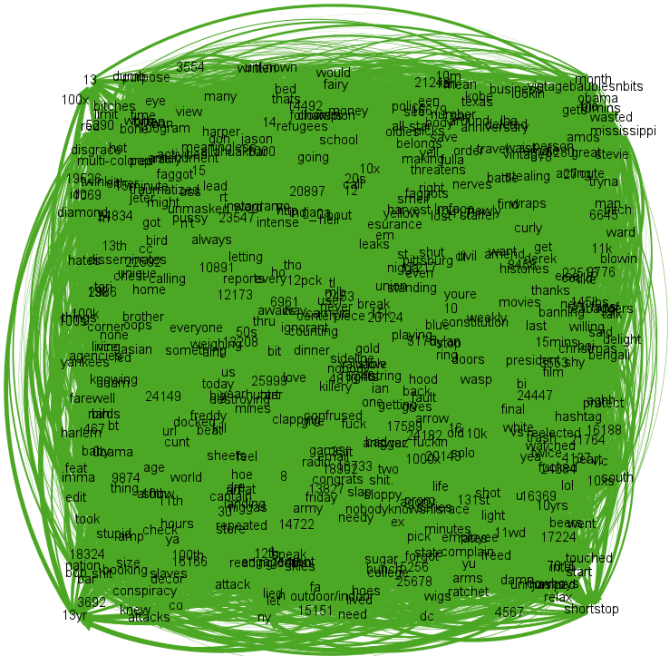

Fig. 3. 1% preview of the Graph

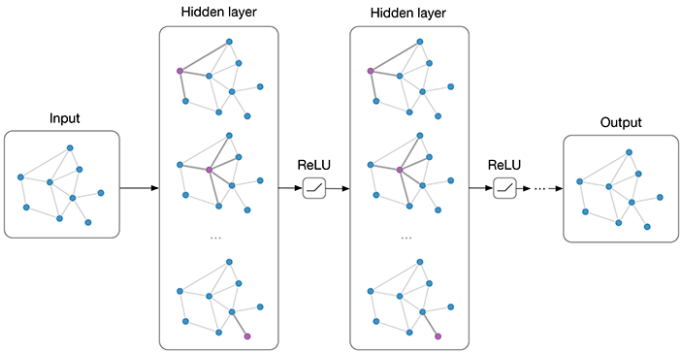

Fig. 4. GCN Model Structure [4]

8.3.2 Model. The architecture of the GCN as shown in Figure 3, consists of 2 hidden layers. The model was built with PyTorch[2], using Adam optimization, and trained with the following parameters, Hidden Layer 1 Size = 330, Hidden Layer 2 Size = 130, learning rate =0.4, and epochs = 100.

8.4 BERT-CNN Model

We used TensorFlow[3] to create the BERT-CNN model. We used the pre-trained Small-BERT model, consisting of 4 Hidden layers with size = 512 and 8 Attention Heads. We also utilized the available pre-processing model for Small-BERT, to provide it with the desired inputs.

[2]PyTorch last visited 19/5/2023.
[3]TensorFlow last visited 19/5/2023.

We batched the dataset into batches of size 32, to allow for GPU utilization for training. The architecture of the model is shown in Figure 1, with two convolutional layers being utilized for the CNN followed by a Global Max Pooling data to downsample the inputs. All layers except for the last Dense layer (classifier), are implemented with the Rectified Linear Unit (ReLU) activation function. The final Dense layer implements a sigmoid activation function. We also experimented with a SoftMax activation function for the last layer but chose against it due to overfitting and lower F1 score for the "Hate Speech" label.

The final model was built using Adam Optimization and trained with the following parameters, epochs = 120 and learning rate = 3 x 10^{-7}. Learning rates of 3 x 10^{-6}, 3 x 10^{-5}, 0.1, 0.03 were also experimented with, along with various epoch lengths such as 10,20,25,40 and 80. However, the model overfits the training dataset with higher learning rates and shorter epochs. Lastly, the final model utilizes un-processed tweets and only relies on BERTs pre-processing, as that provided the best accuracy and F1 score.

8.5 Model Computational Results

The GCN model required firstly the creation of the graph. This task took 4 hours and 13 minutes. Following this, the GCN model was created which required another 13 hours and 27 minutes due to calculations required to create the normalized adjacency matrix proposed in [23].

Such pre-processing was not required for BERT-CNN. The training times for the final models are shown in Table 3. A difference between the two model's training is that the GCN model was trained using the CPU while BERT-CNN utilized the GPU.

Table 3. Training times of BERT-CNN and GCN

Model	Epochs	Time Taken
GCN	100	3 hours and 30 minutes
BERT-CNN	120	17 hours

8.6 Model Results

For comparison of the models, we use the values of precision, recall, and F1-score per class. Precision allows for visualizing the reliability of the model and is calculated by the following formula.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

Recall allows for measuring the ability of the model to detect positive samples and is calculated by the following formula.

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

Lastly, we also calculate F1-score because it provides us with the accuracy of the model by combining precision and recall of the model in the following formula.

$$\begin{aligned} F1\ score &= \frac{2 * Precision * Recall}{Precision + Recall} \\ &= \frac{TP}{TP + 0.5 * (FP + FN} \end{aligned} \tag{3}$$

Where TP denotes True Positive, FP denotes False Positive, TN denotes True Negative and FN denotes False Negative.

The resulting values of Precision, Recall and F1-score of the two models can be seen in Table 4.

Table 4. Per class Precision, Recall, and F1 score

Model		Precision	Recall	F1-Score
GCN	Hate-Speech	0.59	0.23	0.33
	Offensive	0.77	0.95	0.85
	Neither	0.47	0.17	0.25
BERT-CNN	Hate-Speech	0.76	0.57	0.65
	Offensive	0.94	0.96	0.95
	Neither	0.78	0.86	0.82

We also calculated the accuracy and macro F1-score of the models as shown in Table 5. Accuracy allows us to get a general understanding of how many labels are correctly classified by the model. Accuracy is calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

Macro F1-score is an arithmetic mean of the F1 scores of all the labels and allows for an understanding of the model's performance specifically when trained on imbalanced datasets like ours.

Table 5. Accuracy, Macro F1, and Testing Loss

Model	Accuracy	Macro F1-score	Test Loss
GCN	0.74	0.48	0.779
BERT-CNN	0.90	0.81	0.291

BERT-CNN achieved substantially higher F1-scores of 0.65 and 0.82 for the "Hate speech" and "Neither" labels compared to the 0.33 and 0.25 of the GCN for the same. The general accuracy of the BERT-CNN model was also higher than the GCN model.

9 DISCUSSION

To create the models, we utilized a setup consisting of an Intel i7-11800H, 32GB RAM and an RTX 3050 4GB GPU. However, even with this strong setup, both our models were computationally restricted. The graph creation for the GCN model required additional swap memory, as it exhausted 100% of both the CPU performance and the available RAM. The time required for the creation of the graph along with the complete GCN model was 17 hours and 40 mins. Training the model for 100 epochs required another 3 hours and 30 mins. The GCN model could not be trained on GPU due to its large size, therefore our durations are calculated for the model running solely on CPU.

In comparison, the BERT-CNN model was smaller in size and could be trained on GPU. However, to train the model on GPU, we had to limit the BERT-CNN model to use the "Small BERT" model instead of a larger model. Furthermore, the data could only be batched into a smaller batch size of 32, as larger batch sizes could not be accommodated on the GPU. The total time taken to train the BERT-CNN model on GPU for 120 epochs was around 15 hours.

Our final GCN model followed the design proposed in [4], however, in our study the macro F1 score is 0.48 in comparison to their 0.8215. We attribute this difference to the hardware limitations of our study, as the model could not be trained for 200 epochs as in [4], due to lack of memory. Another pitfall is the imbalance in the training dataset, with the total number of hate speech labels only comprising 5.7% of the dataset. Therefore, a balanced dataset should allow for better learning of features.

Another improvement to the GCN could be the use of a larger sliding window like 20, as used in [23] when computing the PMI scores. This could not be done by us, because the increase in sliding window increases the number of edges exponentially, requiring more memory than we had available.

However, even with a simple architecture for the GCN, the model was able to gain 70% accuracy, showing that adding more hidden layers and increasing the complexity of the graph along with a larger training dataset can help further improvement.

Secondly, our BERT-CNN model achieved a lower F1 score when compared to the BERT-CNN model developed in [18]. We attribute this to the use of the small-BERT model, due to hardware limitations, instead of the larger BERT models that provide additional hidden layers and attention heads, allowing for the capture of more contextual features. Moreover, we attribute the lower F1-scores of BERT-CNN for the "Hate speech" and "Neither" labels to the imbalance in the dataset. A balanced dataset should allow for better learning of these features and a higher F1 score.

10 CONCLUSION

In this study, we compared the performance of Graph-based GCN with NLP-based BERT-CNN. The experiment results show that BERT-CNN outperforms the simple 2-layer GCN model, however, the accuracy of the GCN model for its simple architecture suggests that greater results can be achieved with more hidden layers and a more complex graph structure.

We also provided new insights into the computational requirements of these models. In our experiment, both our models were limited by hardware, but the GCN model was more computationally limited than BERT-CNN. We attribute this to a large amount of memory required for the Graph used in the GCN model.

Due to these limitations in computational capacity, we believe further research is required in this domain, to compare these models without any computational limitations.

REFERENCES

[1] Francisca Adoma Acheampong, Henry Nunoo-Mensah, and Wenyu Chen. 2021. Transformer models for text-based emotion detection: a review of BERT-based approaches. *Artificial Intelligence Review* 54, 8 (Dec. 2021), 5789–5829. https://doi.org/10.1007/s10462-021-09958-2

[2] Aymé Arango, Jorge Pérez, and Barbara Poblete. 2019. Hate Speech Detection is Not as Easy as You May Think: A Closer Look at Model Validation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Paris France, 45–54. https://doi.org/10.1145/3331184.3331262

[3] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep Learning for Hate Speech Detection in Tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion*. 759–760. https://doi.org/10.1145/3041021.3054223 arXiv:1706.00188 [cs].

[4] Necva Bölücü and Pelin Canbay. 2021. Hate Speech and Offensive Content Identification with Graph Convolutional Networks. In *Working Notes of FIRE 2021 - Forum for Information Retrieval Evaluation, Gandhinagar, India, December 13-17, 2021 (CEUR Workshop Proceedings, Vol. 3159)*, Parth Mehta, Thomas Mandl, Prasenjit Majumder, and Mandar Mitra (Eds.). CEUR-WS.org, 44–51. https://ceur-ws.org/Vol-3159/T1-4.pdf

[5] Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Emiliano De Cristofaro, Gianluca Stringhini, and Athena Vakali. 2017. Mean Birds: Detecting Aggression and Bullying on Twitter. In *Proceedings of the 2017 ACM on Web Science Conference (WebSci '17)*. Association for Computing Machinery, New York, NY, USA, 13–22. https://doi.org/10.1145/3091478.3091487

[6] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. *Proceedings of the International AAAI Conference on Web and Social Media* 11, 1 (May 2017), 512–515. https://doi.org/10.1609/icwsm.v11i1.14955 Number: 1.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[8] Anamika Dhillon and Gyanendra K. Verma. 2020. Convolutional neural network: a review of models, methodologies and applications to object detection. *Progress in Artificial Intelligence* 9, 2 (June 2020), 85–112. https://doi.org/10.1007/s13748-019-00203-0

[9] Kunihiko Fukushima. 1980. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics* 36, 4 (April 1980), 193–202. https://doi.org/10.1007/BF00344251

[10] Njagi Dennis Gitari, Zuping Zhang, Hanyurwimfura Damien, and Jun Long. 2015. A Lexicon-based Approach for Hate Speech Detection. *International Journal of Multimedia and Ubiquitous Engineering* 10, 4 (April 2015), 215–230. https://doi.org/10.14257/ijmue.2015.10.4.21

[11] Liam Hebert, Lukasz Golab, and Robin Cohen. 2023. Predicting Hateful Discussions on Reddit using Graph Transformer Networks and Communal Context. https://doi.org/10.48550/arXiv.2301.04248 arXiv:2301.04248 [cs].

[12] Kamaljit Kaur and Parminder Kaur. 2023. BERT-CNN: Improving BERT for Requirements Classification using CNN. *Procedia Computer Science* 218 (2023), 2604–2611. https://doi.org/10.1016/j.procs.2023.01.234

[13] Simon Kemp. 2023. Digital 2023: Global Overview Report — DataReportal – Global Digital Insights. https://datareportal.com/reports/digital-2023-global-overview-report

[14] Jiyun Kim, Byounghan Lee, and Kyung-Ah Sohn. 2022. Why Is It Hate Speech? Masked Rationale Prediction for Explainable Hate Speech Detection. http://arxiv.org/abs/2211.00243 arXiv:2211.00243 [cs] version: 1.

[15] M. V. Koroteev. 2021. BERT: A Review of Applications in Natural Language Processing and Understanding. https://doi.org/10.48550/arXiv.2103.11943 arXiv:2103.11943 [cs].

[16] Maximilian Kupi, Michael Bodnar, Nikolas Schmidt, and Carlos Eduardo Posada. 2021. dictNN: A Dictionary-Enhanced CNN Approach for Classifying Hate Speech on Twitter. https://doi.org/10.48550/arXiv.2103.08780 arXiv:2103.08780 [cs].

[17] Jiali Liang, Yufan Deng, and Dan Zeng. 2020. A Deep Neural Network Combined CNN and GCN for Remote Sensing Scene Classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 13 (2020), 4325–4338. https://doi.org/10.1109/JSTARS.2020.3011333 Conference Name: IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing.

[18] Marzieh Mozafari, Reza Farahbakhsh, and Noël Crespi. 2019. A BERT-Based Transfer Learning Approach for Hate Speech Detection in Online Social Media. *CoRR* abs/1910.12574 (2019). arXiv:1910.12574 http://arxiv.org/abs/1910.12574

[19] Seema Nagar, Ferdous Ahmed Barbhuiya, and Kuntal Dey. 2023. Towards more robust hate speech detection: using social context and user data. *Social Network Analysis and Mining* 13, 1 (March 2023), 47. https://doi.org/10.1007/s13278-023-01051-6

[20] Etienne Papegnies, Vincent Labatut, Richard Dufour, and Georges Linarès. 2017. Graph-Based Features for Automatic Online Abuse Detection. In *Statistical Language and Speech Processing (Lecture Notes in Computer Science)*, Nathalie Camelin, Yannick Estève, and Carlos Martín-Vide (Eds.). Springer International Publishing, Cham, 70–81. https://doi.org/10.1007/978-3-319-68456-7_6

[21] Ali Safaya, Moutasem Abdullatif, and Deniz Yuret. 2020. KUISAIL at SemEval-2020 Task 12: BERT-CNN for Offensive Speech Identification in Social Media. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. International Committee for Computational Linguistics, Barcelona (online), 2054–2059. https://doi.org/10.18653/v1/2020.semeval-1.271

[22] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying Graph Convolutional Networks. In *Proceedings of the 36th International Conference on Machine Learning*. PMLR, 6861–6871. https://proceedings.mlr.press/v97/wu19e.html ISSN: 2640-3498.

[23] Liang Yao, Chengsheng Mao, and Yuan Luo. 2018. Graph Convolutional Networks for Text Classification. *CoRR* abs/1809.05679 (2018). arXiv:1809.05679 http://arxiv.org/abs/1809.05679

[24] Si Zhang, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. 2019. Graph convolutional networks: a comprehensive review. *Computational Social Networks* 6, 1 (Nov. 2019), 11. https://doi.org/10.1186/s40649-019-0069-y

[25] Shaomin Zheng and Meng Yang. 2019. A New Method of Improving BERT for Text Classification. In *Intelligence Science and Big Data Engineering. Big Data and Machine Learning (Lecture Notes in Computer Science)*, Zhen Cui, Jinshan Pan, Shanshan Zhang, Liang Xiao, and Jian Yang (Eds.). Springer International Publishing, Cham, 442–452. https://doi.org/10.1007/978-3-030-36204-1_37

[26] Oana Ștefăniță and Diana-Maria Buf. 2021. Hate Speech in Social Media and Its Effects on the LGBT Community: A Review of the Current Research. *Romanian Journal of Communication and Public Relations* 23, 1 (April 2021), 47–55. https://doi.org/10.21018/rjcpr.2021.1.322