Cyberbullying Detection using Attention-based Deep Neural Networks

Edwin Rangga Ardhana[1], Mohd Asyraf Zulkifley[1,*], Martin Spraggon[2], Siti Raihanah Abdani[3]

[1] Department of Electrical, Electronic and Systems Engineering, Faculty of Engineering & Built Environment, Universiti Kebangsaan Malaysia (UKM), 43600 UKM, Bangi, Selangor, Malaysia

[2] Rabdan Academy, 65, Al Inshirah, Al Sa'adah, Abu Dhabi, 22401, UAE. PO Box: 114646, Abu Dhabi, UAE

[3] School of Computing Sciences, College of Computing, Informatics and Media, Universiti Teknologi MARA, Shah Alam 40450, Malaysia

* Correspondence: asyraf.zulkifley@ukm.edu.my

Abstract

Social media is an online tool that connects different individuals from around the world and enables everyone to share and interact with their status as well as updates on their daily lives. However, social media interaction needs to be handled carefully, especially when cyberbullying cases have been on the rise in recent years. It has become a trend that the more time spent on social media, the more likely an individual will experience cyberbullying cases. In general, cyberbullying is a malicious act of causing distress and worry through social media platforms. To address this worrying issue, this paper proposes a cyberbullying detection method using attention-based deep neural networks. Text-based input will be used to validate the proposed method, whereby five classes of major cyberbullying incidents will be the main focus. Firstly, this paper explores the optimal base architecture by determining a set of combined compact layers for a classification network. Secondly, the base network hyperparameters need to be optimized, whereby they play a crucial role in determining the performance of the model. Thirdly, an attention mechanism will be embedded into the base network to put more emphasis on unique terms that represent various types of cyberbullying. Moreover, it allows the network to focus on key phrases and important features. The results show that the proposed attention mechanism-based network produced an accuracy of 0.8311 and a loss of 0.4880. In conclusion, the attention mechanism has managed to improve the performance of a text-based classifier in detecting cyberbullying incidences of various categories, which are due to religion, ethnicity, gender, and age.

Keywords: Cyberbully Detection; Artificial Intelligence; Machine Learning; Deep Learning; Classification.

1 Introduction

Cyberbullying is a significant issue in today's society all over the world. The reported number of individuals attempting suicide because of cyberbullying is increasing at an alarming rate in Malaysia (Kee et al., 2024). The negative consequences of cyberbullying come in various forms and are surely detrimental to the victim. There are multiple types of cyberbullying such as trolling, harassment, flaming, cyberstalking, denigration, frapping, outing, and masquerade that are used to victimize and threaten the victims (Macaulay et al., 2022). The consequences of these bullying actions play a big role in the mental and physical wellbeing of the victim which can result in self-harm and poor quality of life. Victims of cyberbullying face a wide range of effects from poor mental health, self-harming behavior, and poor workplace performance (Li et al., 2023). The victims can feel various categories of negative emotions from depression, hopelessness, worry, and some of the victims might even resort to self-harm or suicidal acts in order to stop the pain (Nikolaou, 2017).

A challenge faced by any intelligent detector based on text input is to understand the emotions solely from the sentence, and it is especially hard if there is a limited length, such as a tweet from Twitter (Nisha and Jebathangam, 2022). Furthermore, it becomes more challenging when the text contains sarcasm or other natural language ambiguities (Vitman et al., 2023). Fortunately, advanced deep neural networks provide an opportunity to mitigate these issues. Various types of deep neural networks through different variants of convolutional neural network (CNN) and recurrent neural network (RNN have been previously developed to identify cyberbullying based on text input, captured from social media platforms (Chan et al., 2023). Generally, a deep learning network consists of multiple layers, including an initial layer for input, various hidden layers, and a final output layer (Thanoon et al., 2023). These models can achieve good results in text classification applications by optimizing the model hyperparameters that suit the respective problems. Each deep model variant has its own strengths and weaknesses according to the input type and size, and an ensemble approach of two or more models is also more likely to improve the classification performance (Ganaie et al., 2022).

To further improve the classifier performance, an attention mechanism can be implemented to increase the precision and accuracy of the network. The attention mechanism aims to improve the model concentration by focusing on selective parts of features that are closely related to the targeted problem (Niu et al., 2021). In this paper, the attention mechanism will be implemented so that the

model can focus on selective parts of the text that might contain cyberbullying intentions, which should be assigned greater weights. This work focuses on cyberbullying detection based on a tweet's text input using a 1D attention-based CNN model. The tweets are first pre-processed and cleaned, turned into vectors, and split into respective training and test datasets, before being fed into the classification network. The original dataset consists of 47,000 tweets that contain six different classes of cyberbullying cases that are equally distributed among them. The optimized base classification architecture must be developed first before the optimal set of hyperparameters can be determined. Then, the attention mechanism will be embedded to further increase the accuracy and precision of the classification network for cyberbullying detection.

2 Related works

In general, social media connects people together, whether among existing friends or making new friends. However, there is a critical issue that needs community attention, whereby it is reported that frequent contact with strangers online will inevitably increase the risk of cyberbullying (Chan et al., 2021). It is also reported that with more exposure to various platforms of social media, the higher the likelihood that someone will fall prey to cyberbullying (Craig et al., 2020).

There are multiple factors that contribute to why an individual becomes a cyberbully (Balakrishnan, 2017). To be specific, an internal factor of low self-esteem is a main contributor to cyberbullying cases, whereby an individual with low self-esteem has a higher likelihood of cyberbullying a victim just to make him feel proud and good (Shaikh et al., 2021). Another reason is to maintain a certain ideal body image, whereby the bully tends to execute cyberbullying activities just for the sake of maintaining a certain expected impression from his closed circle (Berne et al., 2014). A study by Ganapathy et al. (2019) shows that bullying is frequently associated with individuals who are obese or perceived to be obese. Besides that, gender is also one of the main factors when discussing about cyberbullying and bullying. Research by Johansson and Englund (2021) found that there are no clear negative impacts between different genders for both physical and verbal bullying, however, females do experience more cyberbullying as compared to males.

Various deep learning models have been applied to detect cyberbullying on social media platforms such as CNN, RNN, long short-term memory (LSTM), gated recurrent unit (GRU), and Bi-LSTM (Balakrisnan and Kaity, 2023). In its simplest form, these models can be trained and used to predict new text data whether it contains cyberbullying or not. CNN-based approaches have shown some promising results in various applications (Elizar et al., 2022). Previously, CNN has shown

effectiveness for forest detection (Ru et al., 2023), tide level prediction (Su and Jiang, 2023), landslide susceptibility mapping (Hakim et al., 2022), industrial-based quality checking (Zhang et al., 2023) and Covid-19 screening (Zulkifley et al., 2020). CNN's basic architecture consists of multiple convolutional and pooling layers with activation functions, which are coupled with a set fully connected layer in the end. The convolutional layer extracts the comprehensive important features from the input and converts them into numerical feature maps to allow the model to capture low and high-level features (Jayasree and Rao, 2022). Generally, for a text classification task implemented in tweet-based cyberbullying, the extracted features are in the form of a word vector (Zhou, 2020). Then, after subsequent feature extraction process, pooling layers are applied to shrink the feature maps to create smaller feature maps while preserving the important feature information. It is worth to note that CNNs perform well in situations where the detection of position-invariant and local patterns is crucial, hence making them great at identifying patterns such as used keywords. Minaee et al. (2021) attempted to improve their CNN model so that it can capture additional fine-grained characteristics from various sections of the document by using a dynamic max-pooling strategy. Besides that, there are several factors that affect training time such as the number of hidden layers (density) and the number of neurons. An increased number of hidden layers may increase the level of precision during the learning process but will consequently affect the training time as well increasing the likelihood of overfitting (Sadiq et al., 2021). Figure 1 illustrates the CNN model accepting text characters as an input, where the inputs are then subjected to convolutional operations, proceeded by a max-pooling operation to extract its most notable features, followed by multiple convolutional and max-pooling operations, and finally passed onto 3 fully connected layers before the output.

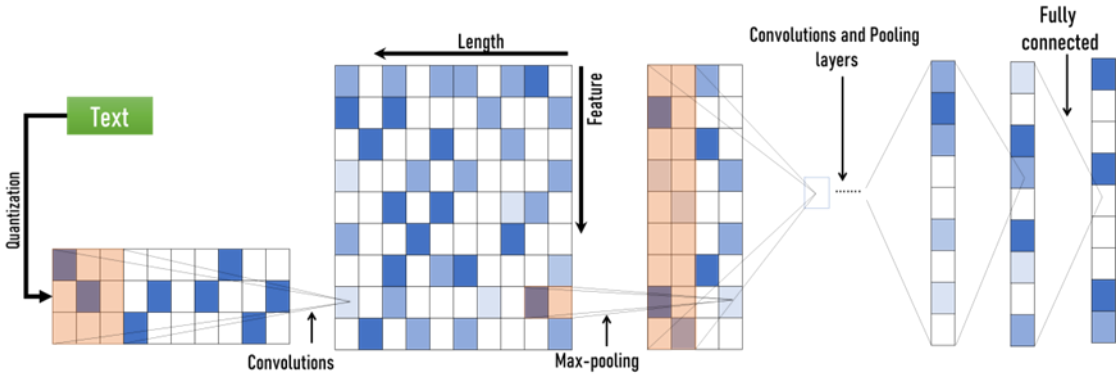

Figure 1 The architecture of a character level CNN model

The concept of an attention mechanism is to put more emphasis on a certain aspect of features or key phrases which in this case, phrases that represent cyberbullying (Wu et al, 2021). This mechanism attempts to concentrate on a few relevant things while ignoring others. The attention vector is then used to predict how strongly a word is correlated to another word. The sum of the words is taken and weighted by the attention vector as the approximation of the focus is combined with the respective word (Minaee et al., 2021). Wang et al. (2019) used an attention mechanism for text classification by using a model with small *n*-grams features that were able to generate larger *n*-grams features from the dense connections. Their multi-scale feature attention has successfully adapted to select an effective set of unique features. After their model had been evaluated, it was found that the model was able to select the optimal scale to form meaningful representations for text classification. Figure 2 illustrates a situation where the deeper the black-colored text, the more attention weights have been placed on it. The difference in color can be attributed to the number of weights that have been placed on it by the decoder in the attention mechanism. It can be noted that the attention mechanism focuses on verbs and nouns that are related to swearing or adjectives and adverbs that are related to emotional expression (Fang et al., 2021).

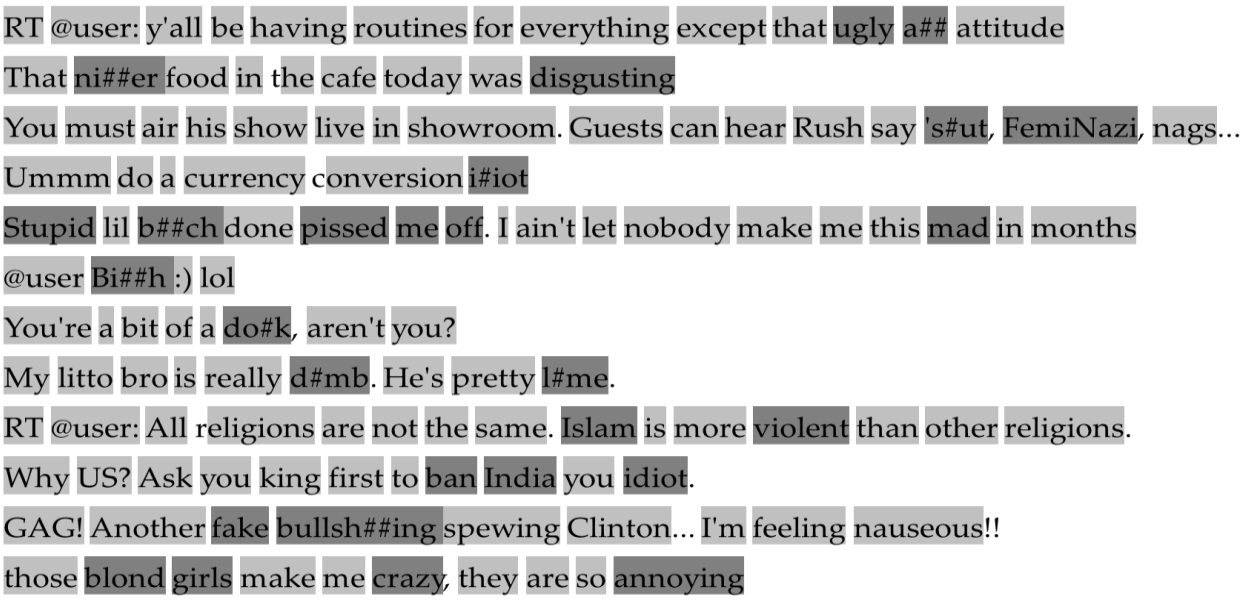

Figure 2 Attention Mechanism on Posts with Self-attention mechanism

2.1 Base deep neural networks used in text analysis applications

There are multiple variants of deep neural networks that are available for text classification tasks and each network has its own strengths and weaknesses (Iqbal and Qureshi, 2022). Some of the base networks that are frequently used for text classification are CNN, RNN, LSTM, and GRU. According to Zulkifley et al. (2023), a good base deep model needs to be selected carefully so that any additional module can be embedded strategically to further improve the model's performance. Furthermore, model complexity also needs to be observed as slight improvement at the cost of large additional computation cost is not an ideal situation. All these different networks have their advantages and disadvantages and need to be analyzed to find the best model configurations that suit the targeted problem (Chai et al, 2021).

The general CNN consists of multiple convolutional and pooling layers with an activation function and a fully connected layer at the end. Important features are extracted by the multiple convolutional layers, which are great for identifying keywords used in the text (Kiranyaz et al., 2021). Normally, more layers are added to produce greater accuracy and precision but at the cost of model complexity as well as difficulty in optimal training of the model (Wang et al., 2018). On the other hand, the RNN uses time series or sequential data and it is a defacto model used in natural language processing (NLP), image translation, and speech recognition (Sherstinsky, 2020). The RNN network has “memory” as the output depends on the prior elements within the sequence. It is good for capturing the local structure of the word sequence but faces difficulties in remembering long-range dependencies (Schaefer et al., 2008). Furthermore, it also faces some generic problems during the training process such as vanishing and exploding gradients (Majumdar and Gupta, 2019).

To overcome the previously mentioned problem, LSTM mitigates the vanishing and exploding gradients by applying memory gates (Qin et al, 2023). Consequently, many predictions, processing, and classification of handwriting recognition are all based on this model. It can effectively capture the contextual information of the text fed into it. The network does not require much fine-tuning for the hyperparameters, but its main downside is heavy computational cost. GRU is another variant of the RNN which also solves the issues of vanishing and exploding gradient problems faced by the RNN (Shen et al, 2018). Compared to the LSTM network, GRU uses fewer parameters and can be processed at a relatively faster speed. GRU uses only two gates which are the update and forget gates that

determine the information that is allowed to be passed, which helps to retrain information to help in making better decisions (Mirzaei et al, 2022).

Bidirectional LSTM (Bi-LSTM) can also be implemented to obtain information from the backward and forward directions of the sentence by summing up and merging the information from both directions (Li et al., 2020). This is possible as Bi-LSTM is made from two independent units of unidirectional LSTMs. The main advantage of the Bi-LSTM network is that it connects to the future as well as the past which enables the network to look into the past and the future to create context for each character inside the text. With this capability, the network can capture more information hence improving the context for the text processing algorithm (Ofori-Boateng, 2023).

2.2 Cyberbullying detection using artificial intelligence approach.

Atoum (2021) devised several applications with the CNN, support vector machines, and naïve Bayes classifiers tested on a collection of tweets to detect cyberbullying incidents. The author found out that the CNN classifier performed the best with the highest accuracy compared to the support vector machines and naïve Bayes classifiers. Badjatiya and Gupta (2017) proposed the approach of using various variants of CNN, LSTM, and FastText for hate speech detection. Based on their testing, it was found that using CNN with random embedding initialization performed better compared to LSTM, which was better than FastText. Finally, the authors found that the addition of gradient-boosted decision trees learned through deep neural networks performed the best.

Iwendi and Srivastava (2023) tested multiple different types of deep neural networks to determine the performance and effectiveness of deep learning algorithms in detecting cyberbullying. The authors used RNN, LSTM, GRU, and Bidirectional LSTM as their deep learning models to benchmark their proposed method. They found out that a modified LSTM with doubled input gates, forget gates, and output gates achieved better accuracy compared to the traditional LSTM network. However, they also noticed a trade-off whereby the proposed model requires more computational complexity and cost in performance.

Zhou (2020) investigated different types of deep neural networks to find the most optimal model for text classification. He has applied CNN, recurrent convolutional neural network (RCNN),

CNN with LSTM (C-LSTM), and FastText to compare the effectiveness of each deep neural network. He found out that CNN managed to successfully capture local correlations, and when it is used for text classification tasks, the CNN was able to extract key information automatically. When learning word representations with repeated structures, his RCNN was found to be able to capture a lot of contextual information and with the help of CNN, it is found that an optimal text representation model can be produced.

3 Methodology

This section presents the dataset and methodology for text preprocessing steps, several variants of the deep neural networks, optimal base architecture using convolutional neural network architecture, hyperparameters tuning of the base model, and attention mechanism embedding into the optimized base deep neural network model. Figure 3 illustrates the flowchart of the overall workflow from the beginning until the embedding of the attention mechanism. The text data of various cyberbullying incidents and normal text was saved in the CSV format, which was retrieved from Twitter, where the dataset is then preprocessed before they are tokenized. The data is then split into 75% of training data and 25% of testing data which were randomized at each run of training the model. The tokenized data is then fed into the deep neural network model where the training process is executed until the model loss converges to zero. The optimal base architecture is first determined by analyzing several different architectures, especially the number of layers and the building block of the model. The optimal hyperparameters are determined by testing individual hyperparameters in sequence by assuming a greedy search approach, which will be optimized separately and carried forward for each subsequent test. Finally, an attention mechanism will be embedded to enable the model to focus on key phrases in the input sequence to increase the model's classification accuracy.

3.1 Cyberbully dataset

The dataset used compromises of around 49,000 tweets which are classified into six categories, which are gender, ethnicity, religion, age, other types of cyberbullying, and not cyberbullying (Wang et al., 2019). The dataset is distributed almost equally among the categories that results in around 8,000 tweets per class. Some of the tweet samples are shown in Table 1 for each cyberbullying category. The tweets under the gender class contain words or sentences that use gender information of an individual

as a slur or part of the bullying context. Tweets that are categorized under the “ethnicity” class contain words or sentences that incline towards racist remarks or use ethnic information to bully another individual. Tweets that are categorized under “religion” contain sentences that slander other individuals based on their religious beliefs, while tweets that are categorized under “age” contain words or phrases that signify that the individual involved in cyberbullying is of different ages or bullying based on the victim’s age. Bullying tweets that are not categorized under “age”, “religion”, “gender”, or “ethnicity” will be categorized as “other cyberbullying” as they do not contain words or phrases that meet the criteria for the previously defined cyberbullying categories. Tweets that are categorized as “not cyberbullying” are tweets that do not have any cyberbullying intent inside the texts. 75% of the dataset is assigned as training data, while 25% of the dataset is assigned as testing data. The splitting of data into two subsets is done randomly every time the code is executed. This process is done to ensure that the training and testing dataset can be representative of the entire dataset and would help prevent the model from overfitting issue that may occur during training process.

Table 1 Samples of Cyberbullying Tweets

Category	**Tweets**
Gender	all my bitches bad, you'll never see me around ugly females
Ethnicity	Get the fuck out of my country dumb nigger whore
Religion	I really hate Muslim twitter. You guys are idiots.
Age	Me when I bullied tf out of this girl in high school for dying her hair half red and black like mine.
Other cyberbullying	I like bullying my sister’s friend. :))
Not cyberbullying	I hate art. It's my hardest class.

3.2 Text Preprocessing

The raw data from the dataset contains a lot of noise information and is arranged in unstructured condition which needs to be preprocessed through several steps. Therefore, text preprocessing aims to transform the text into a clean and consistent format that can be fed into the deep neural network for the model training and learning. Some of the text preprocessing steps that are executed in this work are tokenization, lemmatization, stopword removal, punctuation removal, and lower casing adjustment so that the data is ready for the training process (Uysal and Gunal, 2014). These steps were taken so that the input texts would be free of unwanted information as well as downsizing the input size.

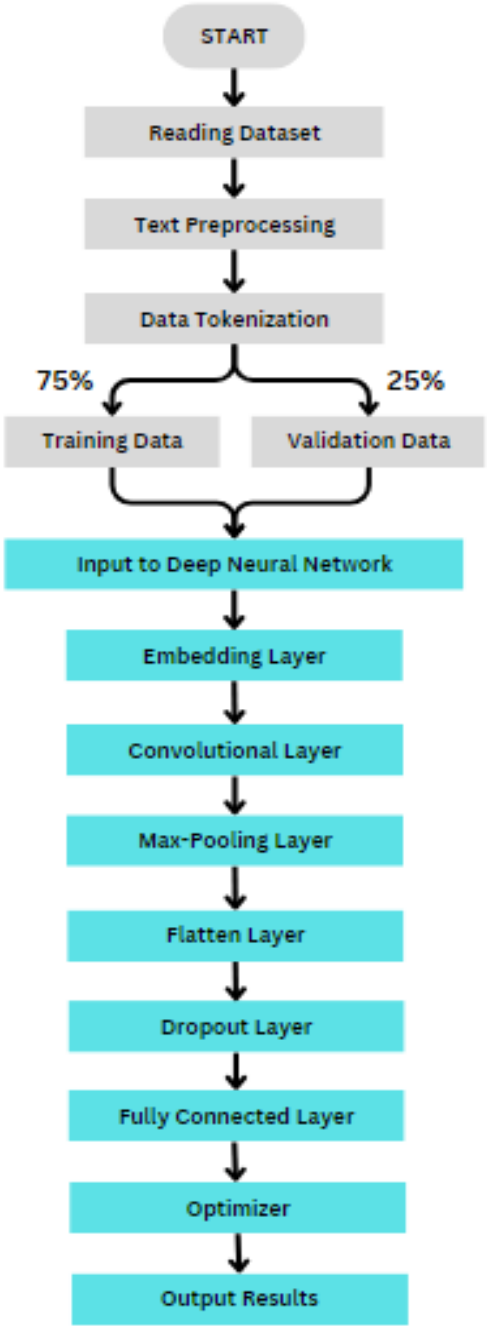

Figure 3 Flowchart of the Program

The first step is tokenization that separates the phrases and sentences inside of the text into smaller units which are known as tokens, whereby in this scenario, the tokens take the form of words (Muller et al., 2022). All the punctuation from the text will be removed and replaced by space-separated sequences of words. The different sequences of words will be split into different tokens which will be then indexed or vectorized.

Then, the lemmatization process will be executed to convert a word into its meaningful base form while still maintaining the relevancy and consideration of the token within its context (Freihat et al, 2018). In the work, Wordnet Lemmatizer with NLTK has been used. It is a large and publicly available lemmatization database of the English language whose goal is to establish structured semantic relationships between the words.

Continuing from lemmatization, stopword removal is then performed to remove commonly used stopwords such as “if” and “it” from the sentence or text. These stopwords take up space inside of the database and would take a longer processing time due to the increased number of tokens involved in the training (Sarica and Luo, 2021). The stopwords do not give much insight into the context and they can normally be removed to enable the extraction of more relevant words.

3.3 Base Convolutional Neural Network Architecture

In this work, four base network architectures are explored to determine the best network to embed an attention mechanism. The first base architecture known as architecture A consists of an input layer, embedding layer, convolutional layer, max-pooling layer, flatten layer, dropout layer, fully connected layer, dropout layer, and fully connected layer for output. Figure 4 illustrates the network design for architecture A. The other three variants would have additional convolutional layers, applied subsequently after the first convolutional layer. Architecture B, C, and D respectively have 2, 3, and 4 layers of convolutional layers and max-pooling layers, respectively. All the hyperparameter values are kept constant during the evaluation of these four variants to ensure consistent and accurate performance results to determine which of the four architectures would produce the best base model. It is important to analyze the number of convolutional layers as the input for our case is a tweet with a maximum limit of 280 characters (the original limit of the tweet set by Twitter).

3.4 Hyperparameter Tuning

Hyperparameters control the learning process of deep neural network models and have a significant impact on the performance of the models (Yang and Shami, 2020). In this work, the hyperparameters that are going to be evaluated are the selection of the optimizer, the learning rate of the optimizer, the filter size of the convolutional layer, the dropout value, the number of nodes in fully connected layers, and the number of epochs.

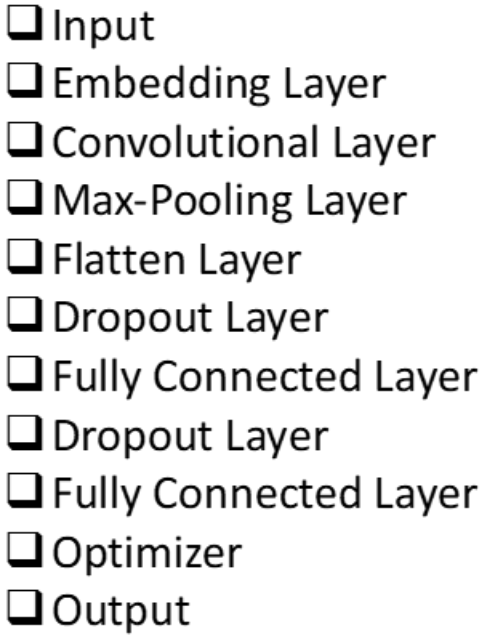

Figure 4 Base Convolutional Neural Network Architecture

The preprocessed data (punctuation removal, stopwords removal, lowercasing all the words, emoji removal, lemmatization, tokenized) from the Twitter dataset will become the input data. The dataset is initially split into 75% of training data and 25% of testing data. The split percentage between training and testing is kept constant throughout the entire experiment. The data is split randomly so that the factor of luck can be reduced as well as allowing the model to be trained by using a diverse set of data. Then, the data is passed to the embedding layer that turns each input, either a token or a word, into a fixed-length vector to represent the words optimally while reducing the dimension of the representative vector. The embedding layer produces a matrix of $X * Y$ whereby X is the number of tokens and Y is the length of the vector for each word. The input dimension of the embedding layer will be the length of the vocabulary and the output dimension will be 100. The input and output dimensions of the embedding layer will be kept constant throughout this paper.

The optimizer will update the weights of the convolutional kernels as well as the weights in the fully connected layers. The goal of the optimizer is to update the weights according to the convergence of the lost function. This paper tested a few optimizers that include Adam, Adamax, Adadelta, Adagrad, Nadam, and SGD (Sengupta et al., 2020). All these optimizers will be tested with their default learning rates and the optimizer that yields the best value will be carried over to the next tests.

In the second optimization process, the learning rate that controls how fast the model can learn will be optimized. A low learning rate value would cause the model to learn slowly but the chance of overfitting is reduced. A high learning rate value would cause the model to learn at a bigger step, which would increase the chance of making the model overfit the training data. The set of learning rates that are tested is its default learning rate plus 0.01, 0.001, 0.0001, and 0.00001. Every learning rate value will be evaluated, and their accuracy and loss will be recorded and analyzed, so that the learning rate that yields the best accuracy and lowest loss will be brought forward for the next hyperparameter optimization process.

The next optimization process is the number of convolutional filters in the convolutional layers, which will determine how much data characteristic will be captured. The convolutional filter will be slid over the row or column of the input matrix to learn various features from the input. Figure 5 illustrates how a CNN model extracts different features from the input text through multiple convolutions, then passes it onto the max-pooling layer and finally to the fully connected layer. The size of the filter will be varied from 8, 16, 32, 64 to 128. Each of them will be tested for the best accuracy and will be carried to the next hyperparameter optimization process.

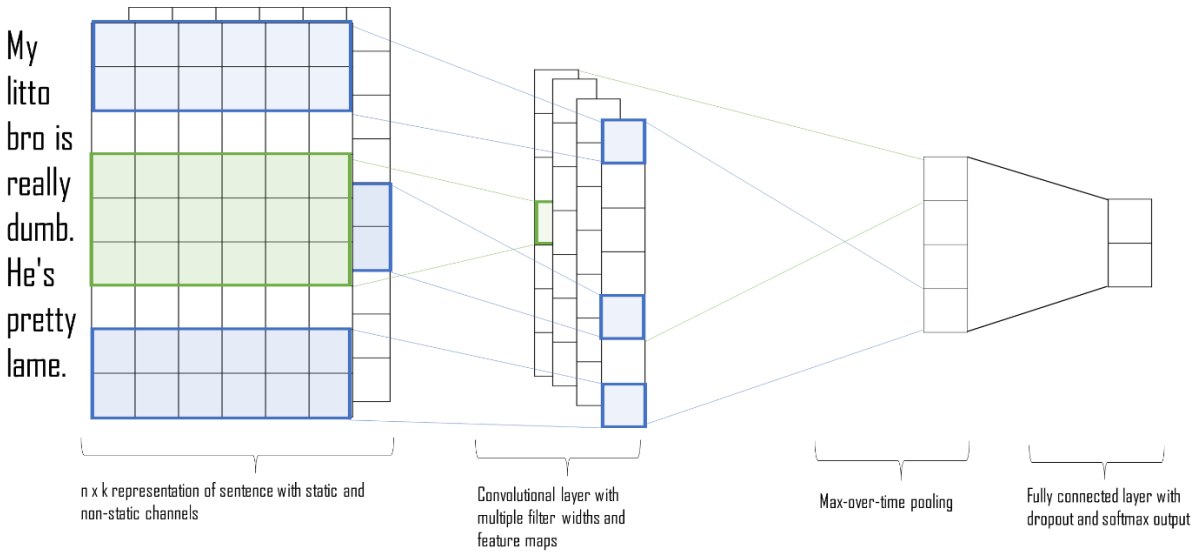

Figure 5 Text CNN Model

The dropout layer “drops” or ignores neurons at random during training, which function is to reduce the likelihood of model overfitting. In this paper, the dropout values that are tested include 0.1, 0.2, 0.3, 0.4, and 0.5. A drop rate of more than 50% is not considered as the network will drop more

than 50% of the kernels, which is not optimal in this problem. The dropout value that produces the best accuracy and the lowest loss will be carried forward to the next hyperparameter optimization.

The last hyperparameter that will be optimized is the number of epochs that determines the number of times the learning algorithm must work through the entire training dataset. A very large number of epochs may cause the model to overfit the data which means that the model has learned the training data too well and does not capture the essence of the data. On the other hand, a very small number of epochs may cause the training process has not converged yet. The perfect number of epochs is determined based on the training and validation loss values, which can be observed if there is a divergence pattern between these two values. If the validation loss keeps increasing while the training loss decreases or stagnates, this could indicate that the model is starting to overfit. On the contrary, if the training loss and the validation loss are both high, this could indicate that the model might suffer from underfitting. The number of epochs tested varies from 5, 10, 25, 35, 70, 80, 90, and 100, whereby the number of epochs that produce the lowest loss and the highest accuracy will become the ideal number of epochs.

3.5 Attention Mechanism

The attention mechanism enables the deep learning model to selectively focus on more relevant parts of the input text sequence. This focus-based approach will improve the performance of the model as well as increasing the efficiency of the model to learn cyberbullying tweets. There are several network configurations that will be adjusted for the optimal attention mechanism embedding such as the number of encoders and decoders, the activation function inside of the attention mechanism, the optimal number of filters of the convolutional layers inside the attention mechanism, and the optimal number of epochs to run the model with the embedded attention mechanism. The max-pooling layers and up-sampling layers in the attention mechanism are used to focus on key phrases and features of the input sequence, which indirectly will reduce the size of effective feature maps. The up-sampling layers are used to increase the size of the feature map to allow the attention mechanism to understand the overall context of the phrase from the input text sequence. The combination of max-pooling and up-sampling layers inside of the attention mechanism enables the attention mechanism to focus on the key important phrases and information of the sentence and also to understand the bigger picture and overall

context of the phrase. The number of max-pooling layers and up-sampling layers inside of the attention mechanism is kept constant throughout the experiments to ensure consistency.

The purpose of the encoders and decoders in an attention mechanism is to effectively enable sequence-to-sequence modeling. This helps the model in capturing the relevant information from the input sequence and produces the contextually relevant outputs. The goal of the encoder is to read the input sequence token step by step and generate a sequence of context vectors or hidden states that capture the input sequences' of the relevant information. The decoder allocates a specific amount of weight to each of the hidden states from the encoder to indicate how much attention should be placed on it. The context vector and internal state of the decoder are used to generate each element for the output sequence. The number of encoder and decoder layers will be varied from 2, 3, and 4, in order to determine which configuration would result in the best accuracy and the lowest loss value.

The activation function in the attention mechanism is used to introduce nonlinearity and control the flow of information. There are various types of activation functions such as the sigmoid activation function, which produces non-linear outputs a value between 0 and 1. Another function, the rectified linear unit (ReLu) activation function outputs either; the original value if the input value is positive, or zero if the input value is negative. The tanh activation function is a non-linear function and produces an output value of between -1 and 1. In general, the sigmoid activation function puts higher weights onto more important features, the ReLu activation function focuses more on the most important features, while the tanh activation function is less sensitive to the vanishing gradient problem. These three activation functions will be tested in order to design the optimal attention mechanism for the cyberbullying classifier.

Besides that, the optimal number of convolutional filters will also be determined by varying the convolutional layer filter size from 16, 32, 64, and 128. The greater the filter size, the more input characteristics can be captured by the model. An attention mechanism contains a few convolutional layers, whereby all these convolutional kernels will be varied using the previously mentioned values. Finally, the number of epochs for evaluating the model with the embedded attention mechanism will be experimented on, based on the following set, 5, 15, 35, and 55. This set was selected because the initial test on 35 epochs looks promising enough to produce a convergence-trained model. When the training plot is analyzed, the intersection between the loss of the training and validation datasets is a good indicator to determine the optimal number of epochs for the best model performance.

4 Results and Discussion

This section presents all the results obtained from testing all the variants of architectures, different configurations of hyperparameters, optimal embedding of the attention mechanism into the network, and benchmarking comparison to demonstrate the benefits of embedding an attention mechanism. All tested models were implemented on Python version 3.8.5 with TensorFlow GPU version 2.10.1. The data has been pre-processed and cleaned before it is fed inside of the deep neural network to reduce the input size while ensuring important information remains intact.

4.1.1 Optimal convolutional neural network architecture

Upon testing the proposed four base architectures using the same set of hyperparameters to ensure experiment consistency between the architectures, the best result is produced by architecture A which consists of 1 convolutional layer and 1 max-pooling layer with an accuracy performance of 0.7944 and a loss value of 1.6066. Then, architectures B, C, and D achieved a lesser accuracy performance of 0.7844, 0.7860, and 0.7751, respectively. Table 2 displays all of the loss and accuracy results from the proposed four base architectures.

From the results obtained, it can be said that one-layer CNN is the optimal model to extract features from a limited 280 characters of tweet compared to multiple convolutional layers. As the number of convolutional layers are added, the accuracy of the model decreases and the loss of the model increases. The decrease in accuracy when more convolutional layers are added may be attributed to the increasing complexity of the model and causing it to overfit. given the limited number of text characters in a tweet. Having a model that is too complex given the size of input data may cause the model to start memorizing the training data as compared to learning the essence and generalizing the patterns from it. Another possibility of reduced accuracy as we increase the amount of convolutional layer is that the training data is not varied enough, whereby some of the repeated bullying terms look similar to each other. There may be a large amount of training data from 48,000 tweets but the amount of data within the individual tweets itself may be too small which causes the model to struggle to capture the characteristics of the data given from the limited information available. Hence, increasing the number of layers will not increase the effective learned patterns that lead to a lower classification accuracy.

Table 2 Performance of the base architecture of the cyberbullying classifiers.

Architecture	Loss	Accuracy
A – One CNN layer	1.6066	0.7944
B – Two CNN Layers	1.7769	0.7844
C – Three CNN layers	2.1802	0.7860
D – Four CNN layers	2.4782	0.7751

4.1.2 Optimal hyperparameter tuning

To determine the optimal optimizer for the proposed model, six different optimizers, which are Adamax, Adam, Adagrad, Adadelta, Nadam, and SGD have been tested with their default learning rates. It is found that the SGD optimizer performed the best with its default learning rate yielding a loss of 0.4724 and an accuracy of 0.8006. When ranking the performance of these different optimizers, the order from the best-performing optimizer to the worst-performing optimizer is SGD, Adamax, Adam, Nadam, Adagrad, and finally Adadelta. Table 3 displays the loss and accuracy results of all the tested optimizers. The main reason why the SGD optimizer may have performed better as compared to the other optimizers is that the dataset does not rely much on the momentum data, which makes SGD, which is the simplest among them work the best. The Adadelta optimizer performed the most poorly as compared to the others as the process of its update window is not optimal for this work dataset, which makes the gradient accumulation not optimal.

Table 3 Performance of the base architecture with varying optimizers for the cyberbullying classifiers.

Optimizer	Loss	Accuracy
Adagrad	0.9604	0.6111
Adam	1.6066	0.7944
Adamax	0.7257	0.7955
Adadelta	1.7354	0.307
Nadam	1.8757	0.7824
SGD	0.4724	0.8006

Hence, the SGD optimizer is determined to be the best optimizer for this work based on the previous evaluation. Then, the learning rate will be varied to determine the most optimal value to obtain the lowest loss and the highest accuracy. The learning rate determines how quickly a model can learn and adapt to the problem. A larger learning rate would result in rapid changes in the weight values and hence the training process may require a fewer number of epochs. However, if the learning rate is too large, it would result in the model converging too quickly which would lead to suboptimal results. A

small learning rate would result in smaller changes in the weights for each update that generally will require more numbers of epochs, and if the learning rate is too small it could result in the model getting stuck to a local extremum. After executing all the experiments, it was found that the learning rates of 0.001, 0.0001, and 0.00001 were too small which resulted in very low accuracy and a large loss. One of the patterns that have been observed for these small valued learning rates is the performance metrics do not change much during the training even after a large number of epochs The learning rate of 0.01 yielded a good accuracy but it is determined that the default learning rate yields the best results with a loss of 0.4723 and an accuracy of 0.8006, which is the lowest loss and highest accuracy among the tested learning rates. Table 4 displays the results of the loss and accuracy of the different learning rates that have been tested. The default learning rate produced the best balance between the training speed and the accuracy.

Table 4 Performance of the base architecture with varying learning rates for the cyberbullying classifiers.

Learning Rate	**Optimizer**	**Loss**	**Accuracy**
Default Rate	SGD	0.4724	0.8006
0.01	SGD	0.4803	0.7996
0.001	SGD	1.5961	0.2732
0.0001	SGD	1.7881	0.2419
0.00001	SGD	1.7915	0.1876

The number of convolutional filter sizes determines how many data characteristics and patterns can be extracted and learned by the model. The larger the filter size, the more feature possibility that it can capture from neighboring words and the more information it can extract from the data. The evaluation of the different numbers of convolutional filters was performed using five different configurations and it was found that the filter size of 32 yields the best result. It obtained a loss value of 0.4687 and an accuracy value of 0.8142. When increasing the filter size from 8 to 32, the accuracy of the model keeps increasing and the loss keeps decreasing. When the filter reaches 64 kernels, the accuracy of the model is nearly identical to when the filter size is 32 but with a higher loss and a slightly lower accuracy. Continuing from before, the filter size of 128 resulted in a loss that was much greater than the previous tests and also with a lower accuracy performance. From these results, the filter size of 32 convolutional kernels is deemed to be the most optimal selection for cyberbullying classifier as shown in Table 5.

Table 5 Performance of the base architecture with varying numbers of convolutional filters for the cyberbullying classifiers.

Filter Size	Loss	Accuracy
8	0.8374	0.7949
16	0.4724	0.8006
32	0.4687	0.8142
64	0.935	0.8128
128	1.0931	0.8074

The dropout unit operates by randomly ignoring some of the neurons (weights) during the training process which results in those particular dropped weights from being updated in that respective pass. The dropout unit helps prevent the model from overfitting by basically simulating the original network with slightly different network structures which will result in a more robust learning process. The dropout rates that have been tested in this work are 0.1, 0.2, 0.3, 0.4, and 0.5. After testing each individual dropout rate, the dropout configuration of 0.5 yielded the best result with a loss value of 0.7965 and an accuracy performance of 0.8210. Table 6 displays the loss and accuracy results when testing different dropout rate values. As the dropout rate increased from 0.1 to 0.4, the accuracy of the model also increased slowly. An anomaly occurred at 0.4 whereby the accuracy decreased but the loss also decreased as compared to the dropout rate values of 0.1, 0.2, and 0.3. When the dropout unit is set at 0.5, it indicates that half of the input will be ignored during the update process, which provides the opportunity for the model to not overfit and allows the model to learn more from the training data through different paths. The dropout rate values of 0.1, 0.2, 0.3, and 0.4 may be too low which is not enough to prevent overfitting from happening, which results in the model not being able to learn as well.

Table 6 Performance of the base architecture with varying drop rates for the cyberbullying classifiers.

Dropout	Loss	Accuracy
0.5	0.7965	0.8210
0.4	0.4930	0.7823
0.3	0.8932	0.8157
0.2	0.4687	0.8142
0.1	0.7880	0.8097

The number of nodes in the fully connected layers was varied from 10, 20, 32, 64, until 128. After conducting all the experiments using all the suggested values, a fully connected layer with 32 nodes results in the lowest loss of 0.4356 and an accuracy of 0.8226. As the fully connected layer node

values were increased from 10 to 32, the loss decreased, and the accuracy increased. When the values of 64 and 128 nodes were tested, the loss decreased but the accuracy of the model also decreased. Table 7 displays the loss and accuracy results of the different configurations of nodes in fully connected layers. It can be said that when the number of nodes in fully connected layers is small, it indicates that the model has simpler connections which makes it capture less information. Hence, when the number of nodes increases, the accuracy of the model keeps increasing. After a certain threshold, the accuracy of the model starts to decrease which would indicate that the model is too complex and starts to show overfitting behavior. Choosing the fully connected layer with 32 nodes balances the model of being not too complex that it would start to overfit while ensuring that the model is not too simple that preventing the model from capturing enough information.

Table 7 Performance of the base architecture with varying fully connected layers for the cyberbullying classifiers.

Fully Connected Layers	Loss	Accuracy
10	0.7965	0.8210
20	0.4284	0.8192
32	0.4356	0.8226
64	0.4161	0.8211
128	0.4084	0.8241

The number of epochs during the training process tells the learning algorithm how many times it must work throughout the training dataset. In this work, the number of epochs has been varied from 5, 10, 25, 35, 70, 80, 90, and 100. After conducting all the simulations with different numbers of epochs, there is a trend when the number of epochs was increased from 5 to 35, the loss of the model decreases while the accuracy of the model increases. However, when the simulations were set with a training epoch of more than 35, the model loss kept increasing while the accuracy plateaued and slowly decreased at the end. Therefore, the optimal number of epochs is set to 35 to yield the best loss of 0.4356 and an accuracy of 0.8226. Table 8 displays the accuracy and loss values from testing various configurations of the epoch. From the results, it can be said that when the number of epochs increases, the accuracy of the model increases up to a certain point, but after this point has been reached, the model will start to overfit, and the accuracy of the model starts to decrease. A low number of epochs will result in the model to underfit, consequently yielding a high loss and low accuracy.

Table 8 Performance of the base architecture with varying numbers of epoch for the cyberbullying classifiers.

Epochs	Loss	Accuracy
5	0.9611	0.6263
10	0.5165	0.7635
25	0.4447	0.8193
35	0.4356	0.8226
70	0.5056	0.8137
80	0.5640	0.8153
90	0.8141	0.7815
100	0.6543	0.8055

4.1.3 Loss and accuracy performance of the optimized base architecture

By using the base architecture with the optimal hyperparameter settings, the proposed model produced an accuracy of 0.8226 and a loss of 0.4356, considerably better performance compared to the unoptimized version of the base architecture. Figure 6 shows the loss plot of the base architecture with the optimal hyperparameter configurations. There are six classes, where classes 0, 1, 2, 3, 4, and 5 represent age, ethnicity, gender, religion, other types of cyberbullying, and not cyberbullying, respectively. The analysis reveals that the proposed model performs the best at detecting cyberbullying due to the factors of age, ethnicity, gender, and not cyberbullying tweets but it struggles to identify cyberbullying tweets due to religion and other types of cyberbullying factors. Table 9 displays the confusion matrix of the proposed base model with the optimal hyperparameters. The overall F1 score and accuracy of the optimized convolutional neural network model is 0.820, a much greater improvement over the original CNN model.

Table 9 Confusion matrix of the optimal hyperparameters

Class	Precision	Recall	F1-Score
0	0.97	0.97	0.97
1	0.97	0.98	0.97
2	0.89	0.83	0.86
3	0.59	0.46	0.52
4	0.58	0.72	0.64
5	0.94	0.97	0.95
Accuracy			0.82
Marco Average	0.82	0.82	0.82
Weight Average	0.82	0.82	0.82

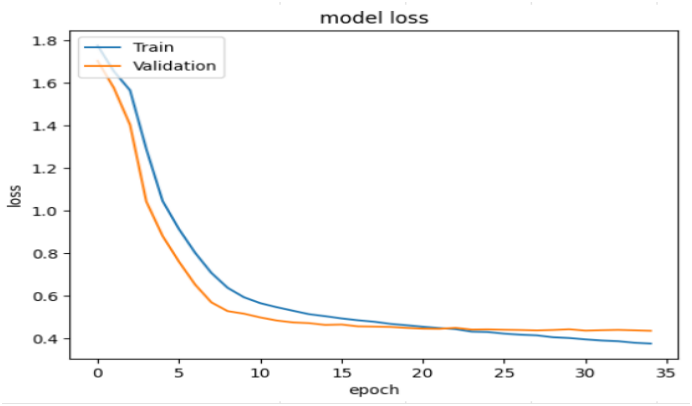

Figure 6 Graph of the loss plots for the base architecture with optimal hyperparameter configurations

4.1.4 Attention mechanism embedding

The first configuration of the attention mechanism that needs to be optimized is the number of encoder and decoder layers, which are varied from 2 to 4 layers. The numbers of encoder and decoder layers need to be finetuned according to the problem, which is in our case cyberbullying tweets. Increasing the number of encoder and decoder layers might improve the model's accuracy but may also result in the model being more difficult to train optimally. Therefore, after running the simulations, the results show that 2 layers of encoder and decoder were deemed to be too simple, which resulted in the model being unable to capture good data representation. On the other hand, a set of 4 layers of encoder and decoder resulted in an increase in accuracy but it was too difficult to train optimally. Hence, the findings show that a set of 3 layers of encoder and decoder proved to be the best by producing an accuracy of 0.8077, whereby it managed to capture complex representations but not too deep. Table 11 demonstrates the different results of the loss and accuracy performance when testing with different numbers of layers of encoder and decoder for the attention mechanism.

Table 10 Loss and accuracy performance by varying the numbers of layers in the encoder and decoder modules of the attention mechanism.

No. of Encoder and Decoder Layers	Loss	Accuracy
4	0.7143	0.8048
3	0.7574	0.8077
2	0.7009	0.7996

The second optimization stage of the attention mechanism architecture is to find the optimal activation function. There are three activation functions that were considered including sigmoid, ReLu, and tanh activation functions. They were all placed inside the attention mechanism to determine which of these activation functions would produce the best performance result. The loss and accuracy results of the different activation functions can be found in Table 12., whereby the ReLu activation function performed the best yielding an accuracy of 0.8151 and a loss of 0.5751. The sigmoid activation function produced a lower accuracy of 0.8104 and had the highest loss of 0.5924, while the tanh activation function yielded the lowest accuracy of 0.8094 and a loss of 0.5446. The main reasons for the better performance of ReLu activation function compared to the sigmoid and tanh activation functions can be attributed to better linearity, making ReLu a much faster activation function. Besides, ReLu is also known to be more robust to the vanishing gradient problem. This means that ReLu‘s gradient will not approach zero despite the input being or approaching infinity. However, this statement is not true for the other two activation functions, making them less optimal for the attention mechanism design.

Table 11 Loss and accuracy performance by varying the activation function of the attention mechanism.

Activation Function	Loss	Accuracy
Sigmoid	0.5924	0.8104
ReLu	0.5751	0.8151
Tanh	0.5446	0.8094

The third optimization stage for the attention mechanism is to analyze the usage batch normalization. It is applied to allow a set of data to be fetched at one time and this respective data will be stretched to the full range, which enables a higher likelihood of learning due to variation in a small set of sampled data. One of the benefits of using batch normalization is it utilizes a wider range of features, which may make data division among classifiers clearer. The results of the loss and accuracy of the tested models, which are with and without applying batch normalization can be found in Table 13. However, the results show that a better classifier is obtained without applying batch normalization. Hence, batch normalization may not be necessary for the proposed model as the dataset used for the model as previously said may not contain enough information variation, which makes scaling to the full range less effective.

Table 12 Loss and accuracy performance between with and without batch normalization applied to the attention mechanism.

Batch Normalization	Loss	Accuracy

With	1.0285	0.7756
Without	0.5751	0.8151

The fourth optimization stage of the attention mechanism is to determine the number of convolutional filters in the network that determines how many data characteristics can be captured by the model. The more convolutional filters used, the more patterns can be captured by linking the neighboring words and the more information can be extracted from the input data. Table 14 displays the results of different loss and accuracy results of the proposed model when tested with different numbers of convolutional filters, which are 16, 32, 64, and 128. The filter size of 32 performed the best by giving an accuracy of 0.8270 and a loss of 0.5305. The filter size of 16 is deemed to be too small and unable to capture enough information from the tweet data, resulting in lower performance accuracy. On the other hand, filter sizes of 64 and 128 captured redundant information from the tweet data which resulted in overfitting of the model and yielded lower accuracy scores. Therefore, an attention mechanism with 32 convolutional filters is the optimal configuration for text-based cyberbullying detection.

Table 13 Loss and accuracy performance by varying the numbers of convolutional filters in the attention mechanism.

Filter Size	**Loss**	**Accuracy**
16	0.5349	0.8089
32	0.5305	0.8270
64	0.5751	0.8151
128	0.5869	0.8092

The total number of training epochs signifies the iteration in which the training data will be used to update the model's parameters. In this work, the set of epoch values has been set to 5, 15, 35, and 55. The performance results of testing several configurations of epochs are shown in Table 15. The results show that training with 5 epochs produced a lower performance before it started to increase until the number of epochs reached 55 epochs based on training data. However, the trend for validation data differs slightly, whereby the performance increases until 15 epochs and then, the performance starts to plateau. The loss plots for the training and validation dataset are shown in Figure 7, whereby the loss lines intersect at 15 epochs and the validation loss continued to decrease after that, which indicates the possibility of overfitting. Based on the performance of 15 epochs, the model yielded an accuracy of 0.8311 and a loss of 0.5305. Comparatively, the performance for a low number of epochs, which is 5 epochs produced an accuracy of 0.8134 and a high loss of 0.6170. On the other hand,

performance results of 55 epochs produced lower accuracy but still a higher loss of 0.5869. Therefore, the value of 15 epochs is selected as the best configuration to train the embedded attention mechanism.

Table 14 Loss and accuracy performance by varying the number of epochs in training the attention mechanism model.

Epochs	Loss	Accuracy
5	0.6170	0.8134
15	0.5305	0.8311
35	0.5751	0.8270
55	0.5869	0.8092

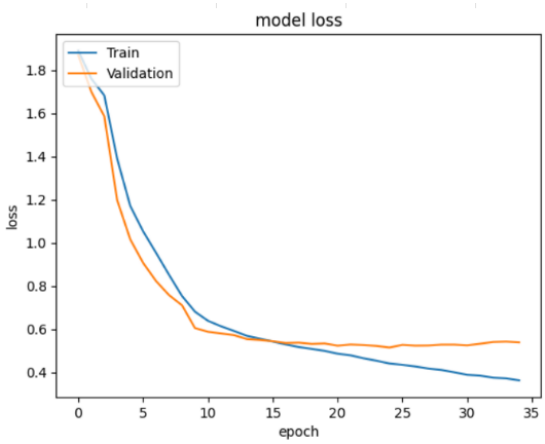

Figure 7 Attention Mechanism Model loss plot at 35 Epochs

Using the optimal configuration of the attention mechanism, the module onto the optimized base network managed to increase the accuracy of the model, yielding a higher accuracy of 0.8311 and a lower loss of 0.4880. The embedded attention mechanism yielded an improvement of 1.034% over the optimized base model without implementing the attention mechanism. Comparatively, the embedded attention mechanism yielded an improvement of 5.33% over the unoptimized base model. Table 10 displays the confusion matrix of the optimized attention mechanism embedded into the optimized base network model.

Table 15 Confusion matrix of the optimized attention mechanism embedded to the base model.

Class	Precision	Recall	F1-Score
0	0.97	0.97	0.97
1	0.98	0.98	0.98
2	0.86	0.87	0.87
3	0.62	0.49	0.55

4	0.59	0.71	0.65
5	0.95	0.95	0.95
Accuracy			0.83
Marco Average	0.83	0.83	0.83
Weight Average	0.83	0.83	0.83

4.1.5 Performance benchmarking with other models

There are multiple types of deep layers based on Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), and Gated Recurrent Unit (GRU) have been tested to benchmark the classification performance. The basic layout of the architecture using these different layers was kept the same, whereby each of these layers will only replace one layer of CNN. The results show that the CNN-based layer performed the best with an accuracy of 0.7944, followed by BiLSTM, BiGRU, LSTM, RNN, and GRU. The performance of the BiLSTM model is relatively similar to the CNN model with a slightly lower accuracy of 0.7938. Contrary to the BiLSTM layer, the worst performing variant is by using GRU with an accuracy of 0.7133. Table 16 demonstrates all of the loss and accuracy performance of the different deep layers. It is important to note that the architectures and hyperparameters of each model were kept consistent throughout the experiments.

Table 16 Performance benchmarking compared to the other deep learning models.

Model	**Loss**	**Accuracy**
CNN	0.4724	0.7944
RNN	0.5483	0.7630
LSTM	0.5373	0.7691
GRU	0.8205	0.7133
BiLSTM	0.5041	0.7938
Optimized Attention-CNN	0.4356	0.8226

In addition to the analysis performance of various basic layers used in the classification model, this research also delves into the impact of embedding an attention mechanism to these various variants. Knowing that an attention mechanism is able to allocate better weight distribution to the specific parts of the feature maps, thereby allowing a better set of features to capture the cyberbullying cases. Therefore, the attention mechanism has also been embedded into the LSTM, GRU, BiLSTM, and BiGRU-based models. All the architectures and hyperparameters of the tested variants are kept the same to ensure consistency of the performance comparison. After testing the addition of embedding the attention mechanism to all different variants, it was found that the embedding of an attention

mechanism into the CNN-based model achieved the highest accuracy of 0.8010, followed by BiGRU, LSTM, GRU, and BiLSTM. The results also showed that all variants with the embedded attention mechanism produced increased accuracies compared to the models without attention mechanisms. Table 17 displays the performance of loss and accuracy of all variants with an embedded attention mechanism. Based on the results, we can safely say that embedding the attention mechanism in the optimized base model increases the accuracy by 2.31%.

Table 17 Loss and accuracy comparison of the benchmarked models with embedded attention mechanism.

Model	Loss	Accuracy
Attention Mechanism + CNN	2.5927	0.8010
Optimized Attention Mechanism + CNN	0.4880	0.8311
Attention Mechanism + LSTM	0.7526	0.7898
Attention Mechanism + GRU	0.7822	0.7860
Attention Mechanism + Bi LSTM	0.5145	0.7801
Attention Mechanism + Bi GRU	0.5398	0.7929

4.2 Conclusion

The implementation of an optimized attention mechanism to focus on specific phrases in the sentence managed to produce the best classification performance. The evaluation of multiple different architecture depths has resulted in the best-reported performance through the simplest architecture (Architecture A) with one convolutional layer. The other architectures that contain more than one convolutional layer tend to lower performance due to overfitting issues. This research then conducted further testing to enhance the performance of Architecture A by optimizing the hyperparameters. By systematically testing each individual hyperparameter, the resultant output yielded a notable increase in accuracy of 5.34% over the network without an optimal set of hyperparameters. Additionally, the optimized embedded network with an attention mechanism managed to produce an accuracy of 0.8226 and a loss value of 0.4356. The integration of an attention mechanism is also proven to work well for various variants of deep models that are based on LSTM, GRU, BILSTM, and BIGRU, whereby these models reported an improved performance compared to their variants without an attention mechanism. This finding showcases the impact of the attention mechanism and demonstrates its potential as an enhancement to deep classification neural networks. It is important to acknowledge that the limitation

of this research is the usage of a single spatial-based attention mechanism, whereas channel-based attention mechanism and dual-attention mechanism have not been explored in this study (Liao et al., 2022). For future work, an investigation on stacked attention mechanisms by Sergio and Lee (2021) can be examined to analyze the effect of multiple attention mechanisms either in sequence or parallel arrangements can be designed to further improve the text-based cyberbullying detector.

Acknowledgement

This research was funded by Universiti Kebangsaan Malaysia under Dana Padanan Kolaborasi with grant number DPK-2023-005 and ISIF-Asia Grant with grant number KK-2022-018 (M-202205-01161).

References

Atoum, J.O. (2021) Cyberbullying Detection Neural Networks using Sentiment Analysis. *International Conference on Computational Science and Computational Intelligence, (CSCI)*, pp.158–164.

Badjatiya, P., Gupta, S., Gupta, M. and Varma, V. (2017) Deep Learning for Hate Speech Detection in Tweets. *Proceedings of the 26th international conference on World Wide Web companion*, pp.759–760.

Balakrishnan, V. (2017) Unraveling the underlying factors SCulPT-ing cyberbullying behaviours among Malaysian young adults. *Computers in Human Behavior*, 75, pp.194-205.

Balakrisnan, V. and Kaity, M. (2023) Cyberbullying detection and machine learning: a systematic literature review. *Artificial Intelligence Review*, pp.1-42.

Berne, S., Frisén, A. and Kling, J. (2014) Appearance-related cyberbullying: A qualitative investigation of characteristics, content, reasons, and effects. *Body image*, 11(4), pp.527-533.

Chai, J., Zeng, H., Li, A. and Ngai, E. W. (2021) Deep learning in computer vision: A critical review of emerging techniques and application scenarios. *Machine Learning with Applications*, 6, pp.100134.

Chan, K.Y., Abu-Salih, B., Qaddoura, R., Ala'M, A.Z., Palade, V., Pham, D.S., Del Ser, J. and Muhammad, K. (2023). Deep Neural Networks in the Cloud: Review, Applications, Challenges

and Research Directions. *Neurocomputing*, pp.126327.

Chan, T. K., Cheung, C. M. and Lee, Z. W. (2021) Cyberbullying on social networking sites: A literature review and future research directions. *Information & Management*, 58(2), pp.103411.

Craig, W., Boniel-Nissim, M., King, N., Walsh, S.D., Boer, M., Donnelly, P.D., Harel-Fisch, Y., Malinowska-Cieślik, M., Gaspar de Matos, M., Cosma, A., Van den Eijnden, R., Vieno, A., Elgar, F.J., Molcho, M., Bjereld, Y. and Pickett, W. (2020) Social Media Use and Cyber-Bullying: A Cross-National Analysis of Young People in 42 Countries. *Journal of Adolescent Health*, 66(6), pp.S100–S108.

Elizar, E., Zulkifley, M.A., Muharar, R., Hairi, M. and Zaman, M. (2022) A Review on Multiscale-Deep-Learning Applications.

Fang, Y., Yang, S., Zhao, B. and Huang, C. (2021) Cyberbullying detection in social networks using bi-gru with self-attention mechanism. *Information*, 12(4), pp.1–18.

Freihat, A. A., Abbas, M., Bella, G. and Giunchiglia, F. (2018) Towards an optimal solution to lemmatization in Arabic. *Procedia computer science*, 142, pp.132-140.

Ganaie, M. A., Hu, M., Malik, A. K., Tanveer, M. and Suganthan, P. N. (2022). Ensemble deep learning: A review. *Engineering Applications of Artificial Intelligence*, 115, pp.105151.

Ganapathy, S.S., Tan, L.A., Sooryanarayana, R., Hashim, M.H., Saminathan, T.A., Ahmad, F.H., Salleh, R. and Abdul Aziz, N.S. (2019) Body Weight, Body Weight Perception, and Bullying Among Adolescents in Malaysia. *Asia-Pacific Journal of Public Health*, 31, pp.38S-47S.

Hakim, W.L., Rezaie, F., Nur, A.S., Panahi, M., Khosravi, K., Lee, C.-W. and Lee, S. (2022) Convolutional neural network (CNN) with metaheuristic optimization algorithms for landslide susceptibility mapping in Icheon, South Korea. *Journal of Environmental Management*, 305(June 2021), p.114367.

Iqbal, T. and Qureshi, S. (2022) The survey: Text generation models in deep learning. *Journal of King Saud University-Computer and Information Sciences*, 34(6), pp.2515-2528.

Iwendi, C., Srivastava, G., Khan, S. and Maddikunta, P.K.R. (2023) Cyberbullying detection solutions based on deep learning architectures. *Multimedia Systems*, 29(3), pp.1839–1852.

Jayasree, M. and Rao, L.K. (2022) A Deep Insight into Deep Learning Architectures, Algorithms and Applications. *International Conference on Electronics and Renewable Systems (ICEARS)*, pp.1134–1142.

Johansson, S. and Englund, G. (2021) Cyberbullying and its relationship with physical, verbal, and relational bullying: a structural equation modelling approach. *Educational Psychology*, 41(3), pp.320–337.

Kee, D. M. H., Anwar, A., and Vranjes, I. (2024). Cyberbullying victimization and suicide ideation: The mediating role of psychological distress among Malaysian youth. Computers in Human Behavior, 150, 108000.

Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M. and Inman, D. J. (2021) 1D convolutional neural networks and applications: A survey. *Mechanical systems and signal processing*, 151, pp.107398.

Li, J., Wu, Y. and Hesketh, T. (2023). Internet use and cyberbullying: Impacts on psychosocial and psychosomatic wellbeing among Chinese adolescents. Computers in Human Behavior, 138, 107461.

Li, W., Qi, F., Tang, M. and Yu, Z. (2020) Bidirectional LSTM with self-attention mechanism and multi-channel features for sentiment classification. *Neurocomputing*, 387, pp.63-77.

Liao, S., Liu, H., Yang, J. and Ge, Y. (2022) A channel-spatial-temporal attention-based network for vibration-based damage detection. *Information Sciences*, 606, pp.213-229.

Macaulay, P. J., Betts, L. R., Stiller, J., & Kellezi, B. (2022). Bystander responses to cyberbullying: The role of perceived severity, publicity, anonymity, type of cyberbullying, and victim response. Computers in Human Behavior, 131, 107238.

Majumdar, A. and Gupta, M. (2019) Recurrent transform learning. *Neural Networks*, 118, pp.271-279.

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M. and Gao, J. (2021) Deep Learning-Based Text Classification: A Comprehensive Review. *ACM Computing Surveys (CSUR)*, 54(3), pp.1–40.

Mirzaei, S., Kang, J. L. and Chu, K. Y. (2022) A comparative study on long short-term memory and

gated recurrent unit neural networks in fault diagnosis for chemical processes using visualization. *Journal of the Taiwan Institute of Chemical Engineers*, 130, pp.104028.

Muller, M., Longard, L. and Metternich, J. (2022) Comparison of preprocessing approaches for text data in digital shop floor management systems. *Procedia CIRP*, 107, pp.179-184.

Nikolaou, D., (2017) Does cyberbullying impact youth suicidal behaviors?. *Journal of health economics*, 56, pp.30-46.

Nisha, M., and Jebathangam, J. (2022) Deep KNN Based Text Classification for Cyberbullying Tweet Detection. *Proceedings of the 11th International Conference on System Modeling & Advancement in Research Trends (SMART)*, pp. 1550-1554.

Niu, Z., Zhong, G. and Yu, H. (2021). A review on the attention mechanism of deep learning. *Neurocomputing*, 452, pp. 48-62.

Ofori-Boateng, R., Aceves-Martins, M., Jayne, C., Wiratunga, N. and Moreno-Garcia, C. F. (2023) Evaluation of Attention-Based LSTM and Bi-LSTM Networks For Abstract Text Classification in Systematic Literature Review Automation. *Procedia Computer Science*, 222, pp.114-126.

Qin, C., Chen, L., Cai, Z., Liu, M. and Jin, L. (2023) Long short-term memory with activation on gradient. *Neural Networks*, 164, pp.135-145.

Ru, F.X., Zulkifley, M.A., Abdani, S.R. and Spraggon, M. (2023) Forest Segmentation with Spatial Pyramid Pooling Modules: A Surveillance System Based on Satellite Images. *Forests*, 14(2), pp.1–20.

Sadiq, S., Mehmood, A., Ullah, S., Ahmad, M., Choi, G.S. and On, B.W. (2021) Aggression detection through deep neural model on Twitter. *Future Generation Computer Systems*, 114, pp.120–129.

Sarica, S. and Luo, J. (2021) Stopwords in technical language processing. *Plos one*, 16(8), pp.e0254937.

Schaefer, A. M., Udluft, S. and Zimmermann, H. G. (2008) Learning long-term dependencies with recurrent neural networks. *Neurocomputing*, 71(13-15), pp.2481-2488.

Sengupta, S., Basak, S., Saikia, P., Paul, S., Tsalavoutis, V., Atiah, F., Ravi, V. and Peters, A. (2020)

A review of deep learning with special emphasis on architectures, applications and recent trends. *Knowledge-Based Systems*, 194, pp.105596.

Sergio, G. C. and Lee, M. (2021) Stacked DeBERT: All attention in incomplete data for text classification. *Neural Networks*, 136, pp.87-96.

Shaikh, F.B., Rehman, M., Amin, A., Shamim, A. and Hashmani, M.A. (2021) Cyberbullying Behaviour: A Study of Undergraduate University Students. *IEEE Access*, 9, pp.92715–92734.

Shen, G., Tan, Q., Zhang, H., Zeng, P. and Xu, J. (2018) Deep learning with gated recurrent unit networks for financial sequence predictions. *Procedia computer science*, 131, pp.895-903.

Sherstinsky, A. (2020) Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404, pp.132306.

Su, Y. and Jiang, X. (2023) Prediction of tide level based on variable weight combination of LightGBM and CNN-BiGRU model. *Scientific Reports*, 13(1), pp.1–13.

Thanoon, M.A., Zulkifley, M.A., Mohd Zainuri, M.A.A. and Abdani, S.R. (2023) A Review of Deep Learning Techniques for Lung Cancer Screening and Diagnosis Based on CT Images. *Diagnostics*, 13(16).

Uysal, A. K. and Gunal, S. (2014) The impact of preprocessing on text classification. *Information processing & management*, 50(1), pp.104-112.

Vitman, O., Kostiuk, Y., Sidorov, G. and Gelbukh, A. (2023) Sarcasm detection framework using context, emotion and sentiment features. *Expert Systems with Applications*, 234, pp.121068.

Wang, J., Fu, K. and Lu, C.T. (2019) SOSNet: A Graph Convolutional Network Approach to Fine-Grained Cyberbullying Detection. *Proceedings of the 2020 IEEE International Conference on Big Data (IEEE BigData 2020)*, pp.1699–1708.

Wang, J., Peng, B. and Zhang, X. (2018) Using a stacked residual LSTM model for sentiment intensity prediction. *Neurocomputing*, 322, pp.93-101.

Wang, R., Li, Z., Cao, J., Chen, T. and Wang, L. (2019) Convolutional recurrent neural networks for text classification. *International Joint Conference on Neural Networks (IJCNN)*, pp.1–6.

Wu, P., Li, X., Ling, C., Ding, S. and Shen, S. (2021) Sentiment classification using attention mechanism and bidirectional long short-term memory network. *Applied Soft Computing*, 112, pp.107792.

Yang, L. and Shami, A. (2020) On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415, pp.295-316.

Zhang, W., Feng, W., Cai, Z., Wang, H., Yan, Q. and Wang, Q. (2023) A deep one-dimensional convolutional neural network for microplastics classification using Raman spectroscopy. *Vibrational Spectroscopy*, 124, pp.103487.

Zhou, Y. (2020) Review of text classification methods on deep learning. *Proceedings of the 2020 3rd international conference on geoinformatics and data analysis*, pp.132–136.

Zulkifley, M.A., Abdani, S.R. and Zulkifley., N.H. (2020) COVID-19 Screening Using a Lightweight Convolutional Neural Network with Generative Adversarial Network Data Augmentation. *Symmetry*, 12(9), p.1530.

Zulkifley, M.A., Munir, A.F., Sukor, M.E.A. and Shafiai, M.H.M. (2023) A Survey on Stock Market Manipulation Detectors Using Artificial Intelligence. *Computers, Materials and Continua*, 75(2), pp.4395–4418.