A picture containing logo

Description automatically generated

**ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ**

**ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ**

**Πτυχιακή Εργασία**

|  |  |
| --- | --- |
| Τίτλος Πτυχιακής Εργασίας | Ταξινόμηση του διαδικτυακού εκφοβισμού χρησιμοποιώντας μοντέλα BERT  Cyberbullying Classification using BERT Models |
| Ονοματεπώνυμο Φοιτητή | Μαυρομμάτης Βασίλειος |
| Πατρώνυμο | Χριστόδουλος |
| Αριθμός Μητρώου | Π15177 |
| Επιβλέπων | Δρ. Διονύσιος Ν. Σωτηρόπουλος, PhD |

Ημερομηνία Παράδοσης Μήνας Έτος

Περίληψη

Ο διαδικτυακός εκφοβισμός (μπούλινγκ) είναι συνηθισμένο φαινόμενο στη σύγχρονη κοινωνία. Όπως και ο εκφοβισμός, ο διαδικτυακός εκφοβισμός μπορεί να πλήξει την κοινωνική, εκπαιδευτική και επαγγελματική ζωή κάποιου ανθρώπου. Δύο κλάδοι της τεχνητής νοημοσύνης είναι η μηχανική μάθηση και η επεξεργασία φυσικής γλώσσας. Με τη χρήση προηγμένων αλγορίθμων μηχανικής μάθησης, οι δυνατότητες της επεξεργασίας φυσικής γλώσσας έχουν προχωρήσει σημαντικά. Σε αυτήν την εργασία με τη χρήση της επεξεργασίας φυσικής γλώσσας και νέων μοντέλων που έχουν αναπτυχθεί για την επίλυση προβλημάτων της σύγχρονης κοινωνίας, θα γίνει προσπάθεια να αναγνωριστούν διάφορα είδη διαδικτυακού εκφοβισμού στα μέσα κοινωνικής δικτύωσης.

Για να επιτευχθεί ο στόχος της εργασίας, δημιουργείτε ένα νευρωνικό δίκτυο με χρήση του BERT, ένα μοντέλο επεξεργασίας φυσικής γλώσσας, βασισμένο σε Transformers, και ανεπτυγμένο από τη Google. Για την εκπαίδευση του νευρωνικού δικτύου χρησιμοποιείται ένα σύνολο δεδομένων από το Twitter. Το σύνολο δεδομένων περιέχει περίπου 47000 "tweets". Πριν εμβαθύνουμε στα βήματα προ επεξεργασίας και στη ρύθμιση του νευρωνικού δικτύου, είναι σημαντικό να κατανοήσουμε τον τρόπο λειτουργίας του BERT. Το BERT λαμβάνει μια ακολουθία από tokens εισόδου και παράγει ενσωματώσεις (embeddings) που αποτυπώνουν τη συμφραστική σημασία κάθε token. Αυτές οι ενσωματώσεις τροφοδοτούνται στη συνέχεια σε ένα επίπεδο ταξινόμησης, όπως ένα νευρωνικό δίκτυο, για να κάνουν προβλέψεις με βάση τις αναπαραστάσεις που έχουν μάθει. Η αρχιτεκτονική των Transformer αναλύεται περισσότερο ώστε να μπορεί να είναι πιο αντιληπτό πως λειτουργεί το BERT. Το πρώτο βήμα πριν από τη χρήση του νευρωνικού δικτύου, είναι η προ-επεξεργασία των δεδομένα καθαρίζοντας το κείμενο και εφαρμόζοντας αποκοπή καταλήξεων (stemming). Επίσης, αφαιρούνται διπλότυπες τιμές και λέξεις με λίγα ή πολλά γράμματα. Στο κείμενο γίνεται η χρήση συμβόλων και η αντικατάσταση λέξεων με σύμβολα (tokenization) χρησιμοποιώντας τον tokenizer "bert-base-uncased". Τα αποτελέσματα, δηλαδή οι πίνακες "input\_ids" και οι "attention masks", μαζί με τις ετικέτες χρησιμοποιούνται για να δημιουργήσουν ένα αντικείμενο συνόλου δεδομένων TensorFlow (TensorFlow Dataset), το οποίο θα τροφοδοτηθεί στο επίπεδο εισόδου (input layer) του νευρωνικού δικτύου.

Τα δεδομένα χωρίστηκαν σε σύνολα εκπαίδευσης (training), ελέγχου (testing) και επικύρωσης (validation). Μετά τη δημιουργία του νευρωνικού δικτύου και την ρύθμιση των υπερπαραμέτρων, το μοντέλο εκπαιδεύτηκε στο σύνολο εκπαίδευσης (training set). Τέλος, το μοντέλο δοκιμάστηκε στο σύνολο δοκιμής (test set) και τα αποτελέσματα του αναλύθηκαν χρησιμοποιώντας ένα πίνακα σύγχυσης (confusion matrix).

*Λέξεις Κλειδιά: διαδικτυακός εκφοβισμός, μηχανική μάθηση, επεξεργασία φυσικής γλώσσας, νευρωνικό δίκτυο, BERT, μετασχηματιστές, πίνακας σύγχυσης*

Abstract

Cyberbullying is something familiar in modern society. Like bullying, cyberbullying can hurt someone's social, educational, and work life. Machine learning and natural language processing are both branches of Artificial Intelligence. With the use of more advanced machine learning algorithms, the capabilities of natural language processing have significantly advanced. In this work, using natural language processing and new models developed to solve problems of modern society, an attempt will be made to identify different types of cyberbullying in social media.

To achieve the goal of the work, a neural network based on BERT is created, a natural language processing model based on transformers developed by Google. This work uses a dataset with "tweets" from Twitter to train the neural network. The dataset has around 47000 "tweets". Before delving into the preprocessing steps and the neural network setup, it is crucial to understand how BERT works. BERT takes in a sequence of input tokens and generates embeddings that capture the contextual meaning of each token. These embeddings are then fed into a classification layer, such as a neural network, to make predictions based on the learned representations. The Transformer architecture, a deep learning model, is further explained to understand better how BERT works. Before using the neural network, the first step is to preprocess the data by cleaning the text and applying stemming. Duplicate values and words with few or too many letters were also removed. The text is then tokenized using the tokenizer "bert-base-uncased". The output, which is the "input\_ids" and "attention masks", and the labels are used to create a TensorFlow Dataset object which will be fed into the input layer of the neural network.

The data was split into training, testing, and validation data. After the neural network creation and tuning of the hyperparameters, the model was trained on the training set. Finally, the model was tested using the test set, and its results were analyzed using a confusion matrix.

*Key Words: cyberbullying, neural network, natural language processing, BERT, tokenization, transformers, confusion matrix, machine learning*

Contents

[1. Introduction 1](#_Toc139048589)

[1.1 Cyberbullying in modern society 1](#_Toc139048590)

[1.2 Machine Learning and Natural Language Processing 2](#_Toc139048591)

[1.3 Using Machine Learning for Cyberbullying Detection 2](#_Toc139048592)

[2. BERT a new linguistic model 3](#_Toc139048593)

[2.1 BERT’s Architecture 4](#_Toc139048594)

[2.1.1 BERT as a Model 5](#_Toc139048595)

[2.1.2 The Transformer Model Architecture 6](#_Toc139048596)

[2.2 Performance of BERT on Benchmarks 8](#_Toc139048597)

[2.3 BERT’s Text Classification Process 9](#_Toc139048598)

[2.3.1 Tokenization and Special Token Embeddings 9](#_Toc139048599)

[2.3.2 Segmentation and Attention Mechanism 10](#_Toc139048600)

[2.3.3 Encoding and Pooling 10](#_Toc139048601)

[2.4 BERT for Cyberbullying Classification 10](#_Toc139048602)

[3. Dataset 11](#_Toc139048603)

[3.1 Dataset Description 11](#_Toc139048604)

[3.2 Preprocessing of Dataset 12](#_Toc139048605)

[3.2.1 Text Preprocessing and Cleaning 13](#_Toc139048606)

[3.2.2 Target Column Encoding 14](#_Toc139048607)

[4. Artificial Neural Network Setup 14](#_Toc139048608)

[4.1 Input Data Preparation 14](#_Toc139048609)

[4.2 Model Architecture and Optimization 15](#_Toc139048610)

[4.2.1 Input Layers 15](#_Toc139048611)

[4.2.2 BERT Layer 15](#_Toc139048612)

[4.2.3 Classifier Head 15](#_Toc139048613)

[4.2.4 Optimization 15](#_Toc139048614)

[4.2.5 Parameter Tuning 16](#_Toc139048615)

[4.3 Training, Testing, and Validation 16](#_Toc139048616)

[4.4 Predictions 17](#_Toc139048617)

[4.4.1 Confusion Matrix and single predictions 17](#_Toc139048618)

[4.4.2 Results 19](#_Toc139048619)

[Conclusion and future work 19](#_Toc139048620)

[Acronyms Table 21](#_Toc139048621)

# 1. Introduction

Bullying is a problem in a society's development from the early years. As time passed and new technological advances like the internet came, bullying spread there. With the widespread use of digital communication platforms and social media, peoaple are increasingly exposed to online harassment, intimidation, and verbal abuse. That is how cyberbullying came to be. The impact of cyberbullying can be severe, leading to psychological distress, social isolation, and even self-harm or suicide in extreme cases. Although it is easy to be recognized by a human, it is not easy to process and label all the comments and posts on the internet.

This work aims to recognize distinct types of cyberbullying across social media using a natural language processing model based on transformers and developed by Google named BERT. A neural network will be created using BERT (Bidirectional Encoder Representations from Transformers). The dataset used to train the neural network consists of different Twitter posts and comments categorized by cyberbullying type.

This work will first analyze different topics relevant to its objective, like what is cyberbullying, machine learning, and natural language processing. Afterward, the structure of BERT and its performance are discussed to understand better the goal and how the different technologies are used to achieve cyberbullying classification. In this part, the architecture of the Transformer, which is a basic part of BERT, is also explained. For a better understanding of later methods used for text preprocessing and how BERT’s classification works, each step of the text classification process used by BERT will be analyzed and explained.

Moreover, the steps to set up an artificial neural network, incorporating BERT as a critical component, are outlined. The input data preparation, model architecture, and optimization techniques will be detailed, covering input layers, BERT layer, classifier head, optimization, and parameter tuning. Finally, the training and testing results of the neural network will be discussed using a confusion matrix concluding what needs to be done for the model to achieve better results and a wider range of applications.

## 1.1 Cyberbullying in modern society

Technological advancement has progressed rapidly for many years, significantly impacting how people communicate and interact in modern societies. Gone are the days when making a phone call or meeting someone in person were the primary means of staying connected. Today, we can send messages or video calls to our friends and loved ones. Instead of physically meeting to catch up and exchange news, we can browse their social media posts and engage by commenting or reacting. However, along with these advancements, numerous issues have emerged. While communication between individuals has become more accessible and convenient, it has become more complex. People can now engage with others they have only met once or interact with strangers through comments on posts. These interactions range from upbeat and pleasant exchanges to hostile and aggressive encounters.

One unfortunate consequence of not seeing each other's faces during online interactions is that some individuals find it easier to express their opinions more aggressively. This expression of aggressive opinions can turn a conversation into a barrage of swearing and hostile comments. Many of these interactions can be categorized as disputes or cyberbullying. “Cyberbullying is defined as ‘…an aggressive intentional act carried out by a group or individual, using electronic forms of contact, repeatedly and over time against a victim who cannot easily defend him or herself’ “as mentioned in the following reference [1]. There are a lot of definitions for cyberbullying but all of them have the same meaning and give a good understating of what cyberbullying is.

Across social media platforms, it is evident that discussions and meaningful conversations are often bypassed in favor of using harmful and hateful comments toward the person who made the post or even towards unrelated topics. While bullying and cyberbullying can affect individuals differently, they generally harm mental health, leading to difficulties in academic studies, work, and interactions with other people. The following study mentions, “Bullying victimization was consistently and robustly associated with an increased likelihood of psychological distress across all measures from depressive symptoms and suicidal ideation to reports of self-injury and suicide attempts. Furthermore, the relationship between victimization and distress was strongest among students who were victims of both cyber and school victimization, followed by victims of cyberbullying only and then victims of school bullying only.” [2].

To prevent these unfortunate outcomes, the first crucial step is to identify and address what constitutes cyberbullying on social media platforms. By raising awareness, developing effective strategies, and promoting a culture of respect and empathy online, we can create a safer and more inclusive digital environment for everyone to enjoy. With machine learning, it is achievable to identify cyberbullying across all social media platforms and their vast number of posts and comments. Two basic technologies that are the core of this study are Machine Learning and Natural Language processing. These two are explained in detail in the next section to better understand how these technologies can help solve modern society’s problems.

## 1.2 Machine Learning and Natural Language Processing

Machine learning is a branch of Artificial Intelligence (AI) that tries to imitate how humans learn using algorithms and data. Machine learning offers valuable solutions across various domains, including making classifications or predictions and finding key insights in a considerable amount of data. By analyzing vast amounts of information, machine learning algorithms can identify and categorize objects, events, or phenomena with impressive accuracy. As the years have passed, technological advances in storage and processing power made it possible to use machine learning for even more challenging projects. With access to large datasets and improved computational resources, machine learning has expanded its reach and capabilities, enabling us to extract key insights and valuable knowledge from complex and unstructured data. The applications of machine learning vary. These applications include image recognition, where machine learning is used to identify objects, persons, places, and digital images. It is also used for speech recognition, medical diagnosis, and automatic language translation.

Natural Language Processing (NLP) as a branch of AI tries to give computers the ability to understand and analyze texts and spoken words. NLP is used for language translation, speech recognition, sentiment analysis, and natural language generation tasks. For example, text tokenization involves breaking down text into individual units, such as words or characters, for further analysis. The semantic analysis aims to understand the meaning and context of words, phrases, or sentences, enabling algorithms to grasp the intended message. Language translation with the use of statistical models and neural networks, automatically translates text from one language to another. Machine learning algorithms, including deep learning models like recurrent neural networks (RNNs) and transformer models, have significantly advanced the capabilities of NLP systems. In this work, a natural language processing model called Bidirectional Encoder Representations from Transformers (BERT) developed by Google is used to help us identify if a comment or post is considered cyberbullying and the type of bullying (age, ethnicity, religion, and gender).

## 1.3 Using Machine Learning for Cyberbullying Detection

Several machine-learning methods can be used to identify cyberbullying in social media posts and comments. In [3], “various classifiers have been used to classify whether the tweet is cyberbullying or non-cyberbullying. The classifier models constructed are LR, Light LGBM, SGD, RF, AdaBoost, naïve Bayes, and SVM.”. All these models mentioned are classification algorithms. These algorithms are used to classify data into different categories or classes based on the patterns and features present in the input data. In another study, SVM and a neural network were used. As mentioned in [4], “We used two classifiers, namely, SVM (Support Vector Machine) and Neural Network. The neural network contains three layers: Input, hidden, and output layer. In the input layer, it consists of 128 nodes. In the hidden layer, it contains 64 neurons. The output layer is a Boolean output.”. In both these studies, the first difference between these and this work is that the methods mentioned above were used to classify whether the input is cyberbullying or non-cyberbullying. As discussed in Section “3 Dataset”, this work’s goal is also to classify the type of cyberbullying.

As seen in both the studies above, the text preprocessing was done almost the same way in this work. Text preprocessing is used to make the text clearer and suitable for further analysis or machine learning tasks. More details are mentioned in section “3 Dataset”. For reference, the text preprocessing includes stemming, removal of stop words, special characters, and repeated words. What was needed for the classifiers used in the studies above and not in this work was feature extraction. Feature extraction refers to transforming raw data into representative features that can be used as inputs for machine learning algorithms or other data analysis tasks. In the context of text data, feature extraction involves converting textual information into numerical representations that capture relevant information about the text. A feature extraction method used in these studies was TF-IDF (Term Frequency-Inverse Document Frequency). TF-IDF is a widely used technique in information retrieval and text mining for evaluating the importance of words or terms in a document within a collection or corpus. No more details are mentioned here for the above methods because these methods and techniques will be explained in more detail in sections 2,3, and 4 based on this work’s target and methodology.

# 2. BERT a new linguistic model

“At the end of 2018, a group of scientists from the Google AI Language laboratory under the leadership of J. Devlin presented a new linguistic model called BERT” [5]. BERT’s model architecture is based on Transformers. The Transformer is “…a model architecture eschewing recurrence and instead relying entirely on an attention mechanism to draw global dependencies between input and output.” [6]. The attention mechanism is a crucial part of BERT and Transformers and will be explained in the section “2.1.2 The Transformer Model Architecture”.

Transformers, a deep learning architecture, have revolutionized natural language processing (NLP) tasks. They consist of an encoder-decoder structure, where the encoder processes the input text, and the decoder generates predictions for the given task. One prominent implementation of the Transformer architecture is BERT. BERT focuses explicitly on the encoder component of the Transformer, as its primary objective is to create a powerful language representation model that can be applied to various NLP tasks. By leveraging the encoder, BERT can capture and encode semantic and syntactic information in the input text. This capability allows BERT to generate high-quality contextualized word embeddings, crucial for performing a wide range of NLP tasks. The architecture of the Transformer is explained in the section titled “2.1.2 The Transformer Model Architecture”.

The encoder in BERT performs a deep bidirectional analysis of the input text. It processes the entire sentence or sequence of words, considering the surrounding context on both sides of each term. This bidirectional analysis enables BERT to capture the dependencies between words, contextual meanings, and relationships within the sentence, resulting in more nuanced and context-aware representations. During the training phase, BERT learns to predict missing words in a sentence using masked language modeling and to understand the relationships between sentences using next-sentence prediction. This self-supervised learning approach enables BERT to learn from vast amounts of unlabeled text, acquiring a comprehensive understanding of language patterns and structures.

The encoder in BERT consists of multiple layers of self-attention mechanisms, which allow the model to assign different weights to each word in the input sequence based on its importance and relevance to the context. BERT can capture long-range dependencies and relationships between terms by attending to different input parts. The resulting contextualized embeddings produced by the BERT encoder are helpful for specific NLP tasks such as sentiment analysis, text classification, question answering, and named entity recognition and downstream functions that require a deeper understanding of language semantics and syntax. By utilizing the encoder part of the Transformer architecture, BERT has become a groundbreaking model in the field of NLP. Its ability to encode rich contextual information in word embeddings has significantly advanced the performance of various NLP tasks, leading to breakthroughs in areas like natural language understanding and generation.

## 2.1 BERT’s Architecture

As we mentioned before, BERT uses the encoder part of Transformers. The image below shows (Image 2.1) we can see the encoder and decoder layers. We are interested in the encoder part because that is what BERT uses. As mentioned in [6], “The encoder is composed of a stack of N = 6 identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise, fully connected feed-forward network. We employ a residual connection around the two sub-layers, followed by layer normalization. That is the output of each sub-layer where , is the function implemented by the sub-layer itself. To facilitate these residual connections, all sub-layers in the model, and the embedding layers produce outputs of dimension = 512.”.

A screenshot of a computer

Description automatically generated with medium confidence

Image 2.1: The Transformer model architecture

Source: <https://doi.org/10.48550/arXiv.1706.03762>

More specifically, BERT’s architecture has two model sizes on which results were reported while testing the model. Those sizes are described here “…we denote the number of layers (i.e., Transformer blocks) as L, the hidden size as H, and the number of self-attention heads as A. 3 We primarily report results on two model sizes: BERTBASE (L=12, H=768, A=12, Total Parameters=110M) and BERTLARGE (L=24, H=1024, A=16, Total Parameters=340M).” [5].

Unlike directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words simultaneously. Therefore, it is considered bidirectional, though it would be more accurate to say it is non-directional. This characteristic allows the model to learn the context of a word based on all its surroundings (left and right of the word).

### 2.1.1 BERT as a Model

How BERT works shows how bidirectional methods are used to solve different NLP tasks. At first, every input sequence must be tokenized. Tokenization is further discussed in the section “2.3.1 Tokenization and Special Token Embeddings”. Then every word must be represented with an embedding. An embedding is a numerical representation of a word, sub-word, or linguistic unit. A dense vector format that machine learning models can process is used to capture the information of each term. There are three types of embeddings: Token, Segment, and Position embeddings. Token embeddings represent the word, sub-word, or linguistic unit using vectors. Segment embeddings are used when multiple sentences are present. By assigning different embeddings to each segment, BERT can recognize the sentence boundaries and consider the relationships between sentences. Position embeddings are added to the token embeddings to convey each token’s relative position or order in the input sequence, enabling BERT to capture sequential information.

Each embedding collectively allowed BERT to understand the contextual relationships and meanings of words within a given text. The token, segment, and position embeddings are then passed through a stack of transformer layers. Each transformer layer consists of self-attention mechanisms and feed-forward neural networks. These layers allow BERT to capture contextual relationships between tokens. Next, 15% of all tokens in each sequence are masked at random. The model then aims to predict the original value of these masked tokens based on the surrounding context. This procedure is called the "masked language model" (MLM). Following MLM, a next sentence prediction (NSP) task is used to help BERT understand the relationship between two sentences by predicting if a given sentence follows the previous one. In training, 50% of correct predictions are fixed with 50% random sentences to help BERT increase its accuracy.

By training on this modified dataset, BERT learns to discern the correct order of sentences and gain a deeper comprehension of sentence-level relationships. On top of the BERT model, other task-specific layers are added to solve a specific NLP task. The different parts of the BERT architecture, including the transformer layers, the masked language model (MLM), the next sentence prediction (NSP) task, and the task-specific layers, are depicted in Image 2.2, providing a visual representation of how these components fit together within the BERT framework.

A picture containing text, screenshot, font, parallel

Description automatically generated

Image 2.2: The BERT model architecture

### 2.1.2 The Transformer Model Architecture

To further understand how Transformers work, the different layers of it must be analyzed. As mentioned in section “2.1 BERT’s Architecture”, the Transformer encoder layer consists of two main sub-layers: a self-attention mechanism and a feed-forward neural network. The self-attention mechanism is a crucial component of a Transformer. It allows it to capture dependencies between different positions within the input sequence. It is also known as scaled dot-product attention. The attention mechanism was first introduced in the context of neural machine translation [7]. The study aimed to address the limitations of traditional sequence-to-sequence models by proposing a new architecture that incorporates an attention mechanism. The authors demonstrated the effectiveness of their approach on several machine translation tasks, showing significant improvements over previous models. Their attention mechanism proved to be a crucial innovation in neural machine translation, providing a more flexible and powerful way to align and translate source and target sequences. In simple terms, attention is a mechanism that allows a model to focus on specific parts of an input. It assigns different weights or importance scores to different words or token mimicking the way humans selectively pay attention to different aspects of information when processing data. Given an input sequence, the self-attention mechanism calculates the attention score between each pair of positions. This is done by taking the dot product between the query and key vectors, followed by a softmax operation to obtain the attention weights. Softmax is a mathematical function that converts a vector of numbers into a vector of probabilities, where the probabilities of each value are proportional to the relative scale of each value in the vector. The label with the highest probability is the predicted value of the neural network. The queries, keys, and values are obtained by applying linear transformations to the input sequence. These transformations project the input sequence into different vector spaces, resulting in each position's query, key, and value vectors. These vectors are then used to calculate attention scores, determining how much each position should attend to other positions in the sequence. The values associated with the attended positions are combined to generate the output representation for each position.

The Attention module repeats its computations multiple times in parallel. Each of these is called an Attention Head. The Attention module splits its Query, Key, and Value parameters N-ways and passes each split independently through a separate head. All these similar Attention calculations are combined to produce a final Attention score. This is called Multi-head attention and gives the Transformer greater power to encode multiple relationships and nuances for each word capturing both local and global dependencies, resulting in a more comprehensive understanding of the input.

Moreover, layer normalization is applied to normalize the output and improve training stability. The output of the self-attention sub-layer is combined with the input (via element-wise addition) to create a residual connection, allowing the model to retain important information from the original input. Dropout regularization is applied to the output of the self-attention sub-layer to prevent overfitting and improve generalization. The Images (Image 2.3, Image 2.4) show how the self-attention mechanism works and how the multi-head attention layer is implemented.

A picture containing text, screenshot, diagram, plan

Description automatically generated

Image 2.3: Matrix Calculation of Self-Attention

A picture containing text, screenshot, diagram

Description automatically generated  
Image 2.4: Multi-Head Self-Attention

Source: <http://jalammar.github.io/illustrated-transformer/>

The second main sub-layer is a feed-forward neural network. This sub-layer applies a simple feed-forward neural network to each position independently. It consists of two linear transformations with a non-linear activation function (e.g., ReLU) in between. Like the self-attention sub-layer, layer normalization is applied to normalize the output of the feed-forward sub-layer. The output of the feed-forward sub-layer is combined with the input (via element-wise addition) using another residual connection. Dropout regularization is applied to the output of the feed-forward sub-layer. Dropout is a regularization method that approximates training many neural networks with different architectures in parallel. During training, some number of layer outputs are randomly ignored or “dropped out.” This has the effect of making the layer look-like and be treated like a layer with a different number of nodes and connectivity to the prior layer. Dropout is commonly used in many neural networks and helps to prevent over fitting.

## 2.2 Performance of BERT on Benchmarks

When initially introduced, BERT significantly impacted natural language processing (NLP), showcasing exceptional performance across various NLP tasks. One notable achievement was its top-ranking position on the SQuAD 1.1 (Stanford Question Answering Dataset) Leaderboard in 2018, where BERT achieved an impressive F1 score of 93.16. This demonstrated its ability to comprehend and accurately answer questions based on given contexts. Furthermore, BERT exhibited remarkable results in other evaluation benchmarks, solidifying its position as a state-of-the-art linguistic model. It excelled in the SWAG (Situations with Adversarial Generations) evaluation, which involved predicting a sentence’s continuation in a multiple-choice format. BERT’s contextual understanding of language enabled it to perform exceptionally well in this task. Another prominent benchmark where BERT showcased its proficiency is the GLUE (General Language Understanding Evaluation) benchmark. GLUE encompasses a diverse set of NLP tasks, such as sentiment analysis, textual entailment, and question-answering. BERT’s ability to capture contextual representations and understand intricate relationships within text allowed it to achieve high scores across multiple GLUE tasks, demonstrating its versatility and effectiveness. However, it is worth noting that since the release of BERT, several new models have emerged with improved results on various NLP tasks, including the GLUE benchmark. The NLP research community has witnessed the development of models such as RoBERTa, ALBERT, and ELECTRA, which have surpassed BERT’s performance on specific tasks. In section “2.4. BERT for Cyberbullying Classification” it is shown that indeed these models can perform well on the task of text classification that is studied in this work, and exceed other classification models in accuracy for this specific task.

Image 2.5 showcases the current leaderboard for the GLUE benchmark, providing a comprehensive overview of the latest models and their performance. This dynamic landscape reflects the continuous advancements in NLP and the ongoing pursuit of achieving even higher benchmarks and pushing the boundaries of language understanding and processing.

A screenshot of a computer

Description automatically generated with medium confidence

Image 2.5: GLUE benchmark leaderboard. The image was taken on March 11, 2023.

Source: <https://gluebenchmark.com/leaderboard>

## 2.3 BERT’s Text Classification Process

In this thesis, we utilize BERT for classifying text as a type of cyberbullying by taking as input a Twitter post or comment. In this chapter, it is explained what the process for BERT is when used for text classification. It is essential to consider that BERT needs different fine-tuning for each other task. The steps below are the most common process for a text classification task using BERT. Some jobs require additional fine-tuning, incorporating additional layers, or modifying the pooling strategy. BERT, as a pre-trained language model, has demonstrated remarkable performance in various natural language processing (NLP) tasks, as seen in the section “2.2 Performance of BERT on Benchmarks”, by capturing the contextual representations of words. With steps, including data preparation, defining the model architecture, tokenization, input representation, fine-tuning, evaluation, and further optimization, the BERT model can be effectively leveraged to classify text and identify various types of cyberbullying.

### 2.3.1 Tokenization and Special Token Embeddings

Tokenization transforms the input text into a sequence of sub-word units called tokens. Each token is assigned a unique index, and the tokens are typically limited to a maximum length to ensure computational efficiency. The tokenization used is called WordPiece tokenization. WordPiece is the tokenization algorithm Google developed to pre-train BERT. It has since been reused in several Transformer models based on BERT, such as DistilBERT, MobileBERT, Funnel Transformers, and MPNET. Instead of treating each word as a single unit, WordPiece tokenization breaks them down into sub-words, as mentioned before. For example, the word "unhappiness" might be tokenized into ["un," "##happy", "##ness"]. The "##" prefix indicates that a sub-word is part of a more important word.

To provide additional information to the model, BERT uses special tokens. These tokens include [CLS], [SEP], and [PAD]. The [CLS] token is inserted at the beginning of the input sequence and represents the collective representation of the entire sequence for classification purposes. In the "next sentence prediction" (NSP) task mentioned in section “2.1 BERT’s Architecture”, there must be a way to inform the model where the first sentence ends and where the second sentence begins. Hence, another artificial token, [SEP], is introduced. Each input sample will contain only one sentence (or a single text input) to train a classifier. In that case, the [SEP] token will be added to the end of the input text. The BERT model receives a fixed length of sentence as input. Usually, the maximum length of a sentence depends on the data we are working on. For sentences shorter than this maximum length, we will have to add paddings (empty tokens) to the sentences to make up the length. In the original implementation, the token [PAD] represents paddings to the sentence.

### 2.3.2 Segmentation and Attention Mechanism

Once the input text is tokenized, BERT performs token segmentation. This step breaks long sequences into smaller segments to fit the model's maximum length. Segmentation ensures the model can efficiently process the input without exceeding its memory limitations. The tokenized sequence is divided into segments, with each segment typically representing a sentence or a document. After segmentation, by using the attention mechanism of the Transformer, the model learns the context of a word based on all its surroundings. The attention mechanism, which is explained in the section “2.1.2 The Transformer Model Architecture” allows each token to attend to other tokens in the sequence, capturing the dependencies and relationships between words. This mechanism helps BERT understand words' contextual meaning in the context of the entire input sequence.

### 2.3.3 Encoding and Pooling

After tokenization and attention mechanisms, BERT encodes the contextual information for each token. This encoding is achieved through a series of Transformer layers, which process the token embeddings in a self-attention fashion. The self-attention mechanism allows BERT to assign different weights to different tokens based on their relevance and contribution to the overall representation. The encoding process enables BERT to capture intricate relationships between words and contextualize their meaning effectively. Pooling is often applied after encoding to aggregate the information from the encoded tokens. Pooling helps condense the encoded representations into a fixed-length vector or a single representation, which can be fed into subsequent layers for classification purposes. Common pooling strategies include max pooling, average pooling, or using the [CLS] token representation as the aggregate representation for the entire text.

## 2.4 BERT for Cyberbullying Classification

In this work, BERT was chosen compared to other machine learning models seen in section 1.3. In many studies and various NLP tasks, BERT performed much better than Naïve Bayes or SVM. The following research [8] shows in comparison that BERT outperformed Naïve Bayes and SVM at the same task mentioning their accuracy percentages. Naïve Bayes and SVM had an accuracy of 52.70% and 71.25%, respectively, and BERT had an accuracy of 91.90% on the same dataset. The dataset that the models were trained in included tweets and posts from Twitter. Another study [9] using again data from Twitter tested and compared different models like SVM, XGBoost (Extreme Gradient Boosting), and BERT alongside some of its variations. BERT models had better results than the other models even when they used one dataset to train the models and another to test them on.

As shown in the next session, a similar dataset, as the one used in [8], is used in this work. There are many models based on BERT, with the most notable being, RoBERTa (Robustly Optimized BERT approach), ALBERT (A Lite BERT), ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately), and DistilBERT (Distilled BERT). These variations of BERT provide researchers and practitioners with options to choose models tailored to their specific requirements, such as performance, model size, efficiency, or multilingual support. Each variation has characteristics, enabling more effective text classification across different domains and languages. The original BERT model was chosen over these variations for three specific reasons for this work's purpose. First, BERT has been widely used and integrated into various NLP frameworks, libraries, and tools. This means extensive community support, documentation, and resources are available for BERT. Also, while the variations of BERT have improvements and specialized features for specific tasks, the original BERT model still performs exceptionally well on general text classification tasks. Lastly, BERT can be improved in future work because it provides a versatile and flexible architecture that can be easily fine-tuned for specific text classification tasks. Its architecture has been widely adopted and modified, allowing for custom adaptations and fine-tuning to suit particular requirements.

# 3. Dataset

To conduct a comprehensive analysis of cyberbullying classification, selecting an appropriate data source that captured a diverse range of posts and comments was crucial. Twitter was chosen as the primary data collection platform among the various social media platforms. Twitter was selected due to its vast user base and the wide variety of posts and comments available for analysis. Specifically, the aim was to gather data that covered different topics, utilized acronyms, and presented challenges in understanding the underlying context. Another viable choice to collect the data from could be Instagram.

The dataset for this work was taken from Kaggle. The following link is for our dataset source <https://www.kaggle.com/datasets/andrewmvd/cyberbullying-classification>. The source article describes the data collection methodology in detail [10]. The selected dataset consists of more than 47,000 tweets, each labeled according to the corresponding class of cyberbullying. The dataset labels for cyberbullying type are age, ethnicity, gender, religion, non-cyberbullying, and other types of cyberbullying. The dataset contains approximately 8,000 instances for each class for a balanced category representation. This balanced approach facilitates a more accurate evaluation of the model's performance and ensures that no particular class dominates the training process.

## 3.1 Dataset Description

The dataset used for this research was stored in a comma-separated values (CSV) file format. The first step to begin preprocessing was to utilize the pandas library in Python to read the CSV file efficiently. By previewing our data, two columns are noticed. The first column, titled "tweet\_text," contained the actual text of the tweets. These tweets serve as crucial sources of information, expressing various sentiments and opinions. The second column, labeled "cyberbullying\_type," provided the corresponding cyberbullying class assigned to each tweet. Notably, the text contained numerous symbols and the usage of shortened words, commonly found in online communication platforms. There are five classes: religion, age, gender, ethnicity, not\_cyberbullying, and other\_cyberbullying. The text contains many symbols and uses shortened words. The presence of an "@" symbol, which typically signifies a mention of a specific person in a tweet. To ensure proper processing and eliminate any potential bias caused by individual mentions, we implemented a preprocessing step that involved removing the "@" symbol from the text data. It is also confirmed that there are around 47000 tweets and the data exhibited a balanced distribution, with approximately 8,000 tweets available for each of the cyberbullying classes, as seen in images 3.1 and 3.2.

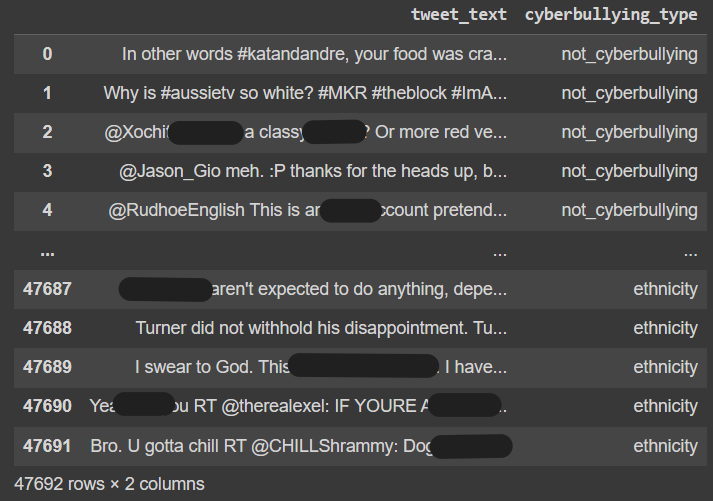


Image 3.1: Preview of the dataset

A screenshot of a computer

Description automatically generated

Image 3.2: Number of tweets for each cyberbullying class

## 3.2 Preprocessing of Dataset

The first step is for the dataset to be preprocessed to be ready for input in our neural network. During the training and testing phases, it became apparent that the predictions generated by the model for the "other\_cyberbullying" class were highly inaccurate. This class, which encompasses instances that do not fall into any specific category of cyberbullying, posed a challenge regarding accurate classification. Given the unreliable nature of the predictions for this class, the decision was made to exclude it from further analysis and focus solely on the remaining classes. By removing the "other\_cyberbullying" class from the analysis, the focus narrows down to the specific categories or types of cyberbullying that can be more accurately identified and classified by the model. This decision allows for a more targeted and precise dataset analysis, enhancing the model's overall performance and reliability in classifying the identified categories of cyberbullying.

### 3.2.1 Text Preprocessing and Cleaning

A few methods are created to preprocess the text. By importing the library “emoji,” we remove all the emojis because they are not crucial to the meaning of the text. The text is further cleaned by removing all new line characters, multiple spaces, links, symbols ("@,” "$,” "&"), characters that are not ASCII, and stop words. Non-ASCII characters refer to characters that do not belong to the standard ASCII character set, which represents the most used characters in the English language. Removing non-ASCII characters is often necessary to ensure compatibility and consistency when working with text data. Furthermore, common stop words, which do not contribute significantly to the semantics of the text, were removed to reduce noise and enhance the model's focus on meaningful content. Stop words are common words that do not carry significant meaning or contribute to the understanding of the text, such as "a," "the," "is," and so on. Removing stop words can help reduce the dimensionality of the text data, improve computational efficiency, and focus on more informative and contextually relevant words. Lastly, stemming, a process that reduces words to their root form, was applied as it yielded superior results compared to lemmatization. Duplicate entries within the dataset were also eliminated to prevent redundancy.

After cleaning and stemming the text, two things are noticed. There were many tweets with less than four words and a lot with more than 100 words. These tweets were removed from the data, and the duplicates were removed again.

A screenshot of a computer

Description automatically generated with medium confidence

Image 3.3: Count of text with less than ten words

A screenshot of a computer

Description automatically generated

Image 3.4: Count of tweets with a high number of words

### 3.2.2 Target Column Encoding

To facilitate the classification task, the target column is encoded using the following scheme: “'religion':0,'age':1,'ethnicity':2,'gender':3,'not\_cyberbullying':4”. This encoding scheme assigned unique numerical values to each class, enabling the model to classify the instances effectively. Finally, duplicate entries were removed from the dataset to ensure its integrity and eliminate potential biases. Retaining unique entries within the dataset, ensures that the model would not be influenced by redundant data, allowing it to focus on valuable and distinct information during the training and evaluation stages.

# 4. Artificial Neural Network Setup

## 4.1 Input Data Preparation

As mentioned, when BERT is described in the section “2.3.1 Tokenization and Special Token Embeddings”, the text needs to be tokenized. As a tokenizer, "bert-base-uncased" [4] is used. In this work, "bert-uncased" is used instead of cased because the case information of the text is not essential. After tokenizing the text and the outputs, we need the input\_ids and attention\_mask, which will be the model's inputs. These two are generated from the tokenizer. The labels, which are categorical data, will be one-hot encoded.

Categorical data refers to variables that are made up of label values. For example, in this dataset, the values are “religion,” “age”, “gender”, “ethnicity”, and “not\_cyberbullying". Some machine learning algorithms can work directly with categorical data depending on implementation, such as a decision tree. However, most require any input or output variables to be a number or numeric value. This means that any categorical data must be mapped to integers. One Hot Encoding is one method of converting data to prepare it for an algorithm and get a better prediction. One Hot Encoding converts each categorical value into a new categorical column and assigns a binary value of 1 or 0 to those columns. Each integer value is represented as a binary vector. All the values are zero, and the index is marked with a 1. The image below (Image 4.1) shows an example of converting a column named “color” using One Hot Encoding.

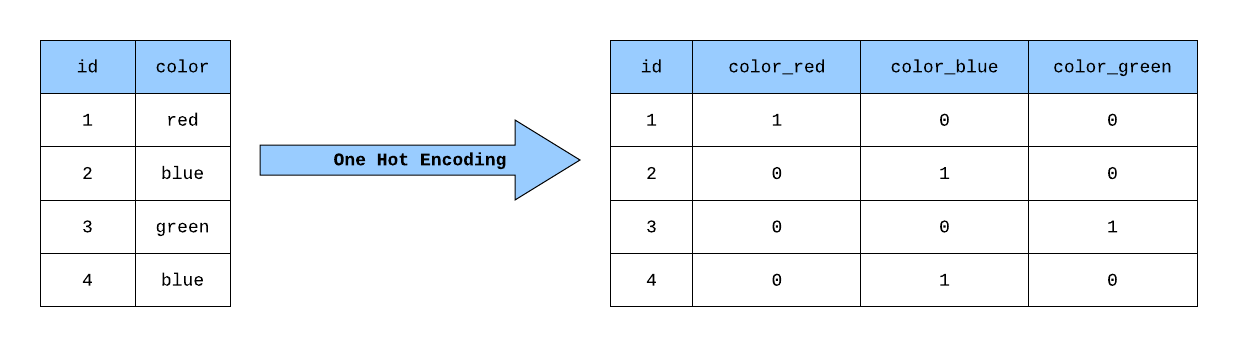


Image 4.1: Converting a column using One Hot Encoding

A TensorFlow Dataset object is created using the three arrays above (input\_ids, attention\_mask, labels). Each dataset element is a tuple containing an input ID, an attention mask, and a label. These three arrays have sizes 512, 512, and 5, respectively, indicating the number of elements in each array. Tuples are used to store multiple items in a single variable. However, a two-item tuple is needed in the "inputs, outputs" format to feed data into the model. The inputs are the “input\_ids” and the “attention\_mask," and the map function applies a transformation to each dataset element. In this case, the transformation involves creating a dictionary with two keys: "input\_ids" and "attention\_mask.” Using the map function, each dataset element is converted into a two-item tuple, where the first item is a dictionary containing the input IDs and attention masks, and the second item is the label. This transformed dataset can now feed data into the model, where each element follows the "inputs, outputs" tuple format, with inputs being a dictionary containing the input IDs and attention masks and outputs being the corresponding labels.

The last step of the data preparation is to split the dataset into train, test, and validation sets. The data is batched and shuffled in batches of 32. Batching involves grouping a certain number of examples and allows the model to process multiple examples simultaneously, optimizing computational efficiency during training. Shuffling randomizes the order of these batches. This process helps reduce bias and ensures that the model receives diverse examples during training. The split is that 80% of the data will be used for training, 10% for testing, and 10% for validation. Evaluating the model on the test set provides valuable insights into its generalization capabilities and helps identify any potential overfitting or underfitting issues. The validation set serves as an additional evaluation subset and aids in fine-tuning the model's hyperparameters.

## 4.2 Model Architecture and Optimization

### 4.2.1 Input Layers

The initial two layers of the model are the input layers used for receiving and encoding the input data. These layers are of size 512, accommodating the "input\_ids" and "attention\_mask" features. The "input\_ids" feature represents the tokenized input sequence, where each token is assigned a unique identifier. On the other hand, the "attention\_mask" feature assists in determining which tokens the model should pay attention to during processing. It is a binary vector of the same length as the input sequence, where each element indicates whether the corresponding token should be attended to. A value of 1 suggests that the token should be attended to, while a value of 0 indicates that the token should be ignored. The layer size is identical to the array size of “input\_ids” and “attention\_mask.” How the “input\_ids” and “attention\_mask” are generated is mentioned in the previous section “4.1 Input Data Preparation”.

### 4.2.2 BERT Layer

The third layer of the model utilizes BERT. The BERT layer uses a transformer architecture consisting of multiple self-attention mechanisms and feed-forward neural networks. The transformer architecture enables the BERT layer to process the input tokens in a bidirectional manner, considering both the preceding and following words when generating contextualized representations. In this architecture, BERT is employed to extract key features from the input sequence using a Max Pooling activation technique. Max Pooling is a pooling operation that calculates the maximum value within each patch of a feature map and uses it to create a down-sampled (pooled) feature map. This process helps to distill the encoded information into a more concise and representative form, emphasizing the most crucial aspects of the text.

### 4.2.3 Classifier Head

The fourth layer is the classifier head which has one Dense layer of size 1024 with a rectified linear unit (ReLU) activation function and another Dense layer of size 5 with a Softmax activation function. For more details of the Softmax function refer to chapter “2.1.2 The Transformer Model Architecture”. The output of ReLU is the maximum value between zero and the input value. Output equals zero when the input value is negative and the input value when the input is positive. Also, the ReLU function can accelerate the training speed of deep neural networks compared to traditional activation functions since the derivative of ReLU is 1 for positive input. Due to constant, deep neural networks do not need additional time to compute error terms during the training phase.

### 4.2.4 Optimization

To train the model effectively, appropriate optimization techniques are crucial. This study used Adam and AdamW optimizers to enhance the learning process. Adam is a popular optimization algorithm widely used in deep learning. It combines the advantages of two other optimization methods: AdaGrad, and RMSProp. AdaGrad adapts learning rates for each parameter based on their gradients. It gives more weight to parameters with small gradients and less weight to those with larger gradients. RMSProp, which applies an exponentially weighted moving average to gradients, is an extension of the Adagrad algorithm and aims to address some of its limitations, particularly the diminishing learning rates over time. Adam dynamically adjusts the learning rate during training, resulting in faster convergence and improved performance.

AdamW is a variant of the Adam optimizer that incorporates weight decay during optimization. Weight decay is a regularization technique that prevents the model from overfitting by adding a penalty term to the loss function. By applying weight decay, the optimizer encourages the model to learn smaller weights, reducing the risk of over-reliance on specific features and improving generalization capabilities. In section “4.3 Training, Testing, and Validation” it is shown that AdamW provided better results in this study’s classification task.

### 4.2.5 Parameter Tuning

During the experimental phase of this thesis, the learning rate and weight decay for AdamW were altered to investigate their impact on the model's performance. The learning rate determines the step size taken during optimization, affecting the speed and quality of convergence. A higher learning rate may lead to faster convergence but risks overshooting the optimal solution, while a lower learning rate may ensure more accurate convergence but at the expense of longer training time. On the other hand, weight decay controls the amount of regularization applied to the weights. Modifying these parameters makes it possible to fine-tune the model's behavior and explore the optimal configuration for achieving superior performance.

## 4.3 Training, Testing, and Validation

The training results were the same in both Adam and AdamW, with around 78% training accuracy and 80% validation accuracy with a learning rate of 1e-4 for Adam and AdamW. The weight decay for AdamW was 1e-8. At first, the third layer, BERT, was not trained for the data because it has 109482240 parameters, and the training would require many resources. With the AdamW optimizer, a learning rate set to 5e-5, a decay of 1e-8 at four epochs, and with the BERT layer set to not trainable, the training accuracy was 73.4%, and the validation accuracy was 76.3%. Even with six epochs, the results were 74.9% for training accuracy and 75.8% for validation accuracy, less than when the epochs were 4. The third layer was set to trainable to improve our results further, and although the training time was increased, the results were much better. With the AdamW optimizer and the same parameters as before but with the BERT layer set to trainable, the training accuracy increased to 97.6%, and the validation accuracy was 95.6%. Again, with 6 and 8 epochs the results were the same or worse. With these tests, it was concluded that 4 epochs and the BERT layer set to trainable using the AdamW optimizer with a learning rate of 5e-5 and a decay of 1e-8 produced the most accurate results in both the training and validation accuracy. In the section “4.4.1 Confusion Matrix and single predictions,” the difference between the BERT layer being set to trainable and not trainable is better illustrated using a confusion matrix.

Setting the BERT layer to trainable allows the optimization algorithm to update the weights and biases of the layer during training, potentially adjusting the model's learned representations. This can be useful if there is a relatively small dataset or the pre-trained layers need further fine-tuning for your specific task. Training the entire BERT model from scratch requires substantial labeled data and computational resources, as it restarts the training process. It may also increase the risk of overfitting if the dataset is small. Therefore, fine-tuning with trainable task-specific layers is generally recommended. However, experimenting with different approaches can help you understand the impact of training the entire BERT model for your specific use case.

Using TensorBoard, it is shown below how the model performed during each epoch. Image 4.2 shows how the accuracy changed after each epoch and the second loss after every epoch with the BERT layer set to trainable. Image 4.3 shows the same accuracy and loss, but the BERT layer is not trainable. The BERT layer being set to trainable provided a lot better results (Image 4.2). A comparison between the BERT layer being set to trainable and not trainable is also made using a confusion matrix in the section “4.4.1 Confusion Matrix and single predictions”.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Image 4.2: Epoch accuracy and epoch loss with AdamW optimizer on four epochs and BERT layer set to trainable.

(Blue Line is evaluation, and the orange is training)

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Image 4.3: Epoch accuracy and epoch loss with AdamW optimizer on four epochs and BERT layer set to not trainable.

(Blue Line is evaluation, and the orange is training)

## 4.4 Predictions

As mentioned previously, in the section titled "4.1 Input data preparation" the model was tested on 10% of the dataset. The results obtained from the testing phase demonstrated that the model performed at a comparable level of accuracy to that achieved during validation. This outcome indicates the model's ability to generalize well and make accurate predictions on new, unseen data. Furthermore, predictions were made on sentences outside the dataset to see how the model would perform. The ability to make accurate predictions on sentences outside the dataset is a crucial aspect of the model's effectiveness in real-world cyberbullying detection scenarios, where it encounters new and evolving forms of cyberbullying. These testing efforts contribute to the overall assessment of the model's effectiveness and provide valuable insights into its performance in real-world settings.

### 4.4.1 Confusion Matrix and single predictions

A confusion matrix is a performance measurement for a machine learning classification problem where the output can be two or more classes. It is a table with four different combinations of predicted and actual values. The image below (Image 4.4) explains what each column means. In the image’s example, the real value is 3. If the predicted value is 3, then it is a true positive. Table 1, which is shown below, helps to understand the different values present in a confusion matrix and interpret the various elements and their meanings. True Positive values, for example, are when the model correctly predicted the actual value of a given input. It is essential to understand the values in a confusion matrix to evaluate the performance of a model. By analyzing the values, the model's accuracy, precision, recall, and other relevant metrics, can be assessed. These metrics are then used to make informed decisions regarding the model's effectiveness in identifying and classifying instances accurately.

A screenshot of a computer

Description automatically generatedA picture containing text, screenshot, square, number

Description automatically generated

Image 4.4: Example of a multi-class confusion matrix values

Confusion Matrix Values

|  |  |
| --- | --- |
| TP | True Positive: The **actual value** is **true**, and the **predicted value** is also **true.** |
| TN | True Negative: The **actual value** is **false**, and the **predicted value** is also **false** |
| FP | False Positive: The **actual value** is **false**, and the **predicted value** is **true.** |
| FN | False Negative: The **actual value** is **true**, and the **predicted value** is **false.** |

The image below (Image 4.5) shows the confusion matrix from the test set predictions. The true positives are the values that start from 0,0 and go diagonally to 4,4. To better understand the meaning of each number, refer to the previous image which uses as a background the confusion matrix below. Each number corresponds to a cyberbullying type. As it was mentioned before in the section titled “3.2.2 Target Column Encoding”, the values that correspond to each number are 'religion':0, 'age':1, 'ethnicity':2, 'gender':3, 'not\_cyberbullying':4.” For comparison to the right, we see the confusion matrix of the test results when the model was trained with the BERT layer set to not trainable. The confusion matrices, show that the most wrong predictions were when either the predicted or the actual value was “4: not \_cyberbullying”. The model was tested to an additional 100 sentences and labeled 82 correctly. The sentences written were random with no high difficulty in understanding the context and if it contained any type of cyberbullying. Simple and clear in context sentences and comments were selected to make sure that the model could grasp the essence of what bullying means without misinterpreting some swear words for bullying even if they are not. This test was done to ensure that the model could correspond to data and label it correctly even if it was never in the training or test set. The most mistakes were again made when the actual label was "not\_cyberbullying," and the model labeled it as a type of cyberbullying.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Image 4.5: Confusion matrix of running the model on the test set.

(Left: BERT layer set to trainable. Right: BERT layer set to not trainable)

### 4.4.2 Results

The performance of the model exceeded our expectations, yielding highly satisfying results. Our experiments showed that most predictions made by the model were accurate, even when dealing with more complex and sophisticated sentence structures. However, there were challenges where the model needed help to make precise predictions. This difficulty primarily arose from the nuanced nature of language, wherein what might initially appear as a negative comment does not necessarily indicate an instance of bullying.

To illustrate this point, let us examine the following sentence: "I miss the time when I was young and irresponsible." In this case, the model mistakenly labeled it as "1: age," whereas it should have been labeled as "4: not\_cyberbullying.". Another example is the following sentence: “I don’t understand how living in Europe is worse than living in America”. The previous sentence was classified as “2: ethnicity” while the correct label is “4: not\_cyberbullying”. These misclassifications further highlight the intricacies involved in distinguishing between genuine instances of bullying and harmless expressions that might be misconstrued as such.

The same misclassification was observed upon examining the confusion matrices, emphasizing the need for improvement in accurately identifying and classifying non-cyberbullying content. In the example, the correct label of "4: not\_cyberbullying" was previously mentioned when discussing the confusion matrices, reaffirming the importance of rectifying such mispredictions. While our model demonstrated impressive performance overall, these instances serve as valuable learning opportunities for enhancing accuracy and fine-tuning its understanding of context.

# Conclusion and future work

Cyberbullying can have a severe impact on people's everyday lives. Nowadays, it is hard to identify which comments or posts classify as cyberbullying because of the hidden meanings of the acronyms. It is also difficult to know if the comment or post is directed to someone specifically. It is easy for the human eye to sometimes understand if a certain comment or post should be considered bullying, but it is not easy to inspect all the comments and posts across all social media platforms. The solution to this problem could be to have a neural network identify and classify cyberbullying.

This work's neural network using BERT had a validation accuracy of 95.6%. Even though these are exceptional results there are some problems and many improvements that can be made. It had shown that the most frequent problem was when the comment or post was not considered cyberbullying, but the model classified it as such. A way to improve that and the accuracy is to do better and more accurate data preprocessing. For example, many duplicates were removed during the preprocessing, and the dataset needed to be more balanced. The last class (4: not\_cyberbullying) had ~1000 entries less than the other classes.

Moreover, modifying the neural network's architecture, such as adjusting the layers or altering their sizes, presents another avenue for enhancing overall accuracy. However, it is essential to acknowledge that such modifications may introduce trade-offs, potentially increasing training time alongside improved performance.

Lastly, expanding the dataset by incorporating additional data from various social media platforms can be a valuable step toward capturing the diverse range of comments and posts encountered online. Right now, a lot of acronyms and argots of groups of people are commonly used in social media. Having a dataset from many online sources and precisely labeling it can help the model to be better fitted to our needs. This approach enables the model to learn from a more comprehensive array of contexts and user interactions, improving its ability to detect and accurately classify cyberbullying instances.

# Acronyms Table

|  |  |
| --- | --- |
| AI | Artificial Intelligence |
| NLP | Natural Language Processing |
| BERT | Bidirectional Encoder Representations from Transformers |
| MLM | Masked Language Model |
| NSP | Next Sentence Prediction |
| SQaAD | Stanford Question Answering Dataset |
| SWAG | Situations With Adversarial Generations |
| GLUE | General Language Understanding Evaluation |
| ReLU | Rectified Linear Unit |
| RNNs | Recurrent Neural Networks |
| LR | Linear Regression |
| SGD | Stochastic Gradient Descent |
| RF | Random Forests |
| SVM | Support Vector Machine |
| CSV | Comma Separated Values |
| TF-IDF | Term Frequency-Inverse Document Frequency |
| AdaBoost | Adaptive Boosting |

Bibliography

**[1]:** Grigg, D. (2010). Cyber-Aggression: Definition and Concept of Cyberbullying. Journal of Psychologists and Counsellors in Schools, 20(2), 143-156. <https://doi.org/10.1375/ajgc.20.2.143>

**[2]**: Schneider, S., O'Donnell, L., Stueve, A., & Coulter, R. (2012). Cyberbullying, School Bullying, and Psychological Distress: A Regional Census of High School Students. American Journal of Public Health, 102(1), 171-177. <https://doi.org/10.2105/AJPH.2011.300308>

**[3]**: Muneer, A., & Fati, S. M. (2020). A Comparative Analysis of Machine Learning Techniques for Cyberbullying Detection on Twitter. Future Internet, 12(11). <https://doi.org/10.3390/fi12110187>

**[4]**: Hani, J., Mohamed, N., Ahmed, M., Emad, Z., Amer, E., & Ammar, M. (2019). Social media cyberbullying detection using machine learning. International Journal of Advanced Computer Science and Applications, 10(5).

**[5]**: Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Retrieved March 20, 2020, <https://doi.org/10.48550/arXiv.1810.04805>

**[6]**: Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, & Illia Polosukhin. 2017. Attention Is All You Need. [*https://doi.org/10.48550/arXiv.1706.03762*](https://doi.org/10.48550/arXiv.1706.03762%20)

**[7]**: Dzmitry Bahdanau, Kyunghyun Cho, & Yoshua Bengio. (2016). Neural Machine Translation by Jointly Learning to Align and Translate. <https://doi.org/10.48550/arXiv.1409.0473>

**[8]**:Desai, Aditya, Kalaskar, Shashank, Kumbhar, Omkar, & Dhumal, Rashmi. (2021). Cyber Bullying Detection on Social Media Using Machine Learning. ITM Web Conf., 40, 03038. <https://doi.org/10.1051/itmconf/20214003038>

**[9]**:Malik, J. S., Pang, G., & Hengel, A. V. D. (2022). Deep learning for hate speech detection: a comparative study. [https://doi.org/10.48550/arXiv.2202.09517](https://doi.org/10.48550/arXiv.2202.09517%20)

**[10]**:J. Wang, K. Fu, C.T. Lu, “SOSNet: A Graph Convolutional Network Approach to Fine-Grained Cyberbullying Detection,” Proceedings of the 2020 IEEE International Conference on Big Data (IEEE BigData, 2020), December 10-13, 2020. [*https://doi.org/10.1109/BigData50022.2020.9378065*](https://doi.org/10.1109/BigData50022.2020.9378065)