Text Mining and Natural Language Processing

2024-2025

**Mykhailo Arlamov (535261), Pelinsu Topaloğlu (533792)**

# BuDecT (Bullying Detector on Twitter)

## Introduction

**Cyberbullying** on social media has become a prominent issue with the rise of digital communication and social media platforms. Harmful posts on these platforms might perpetuate **hate** based on **age**, **ethnicity**, **gender**, **religion**, and **other** human attributes. In this project, we frame cyberbullying detection as a **single label multi-class classification** problem. Given the text of a post, the project initially aims to detect if it is abusive, and if so, to assign it to one of the five aforementioned bullying categories. We employ several **word representation techniques** and **compare** their performance across different categories. We are comparing performances of different NLP approaches, starting from traditional NLP and finishing with the transformer.

## Data

The **cyberbullying dataset** contains *47692* entries, and it is made up of two fields:

- *Tweet_text:* the raw tweet as a string
- *Cyberbullying_type:* the label annotated by humans

Each entry is classified into one of the 6 categories under the *cyberbullying_type*:

- *Age*
- *Ethnicity*
- *Gender*
- *Religion*

- *Other_cyberbullying*
- *Not_cyberbullying*

The dataset is split into 3 sets:

- *Train set* with *32193* entries, corresponding to 70% of the total dataset,
- *Validation set* with *6899* entries, corresponding to 15%,
- *Test set* with *6899* entries, corresponding to 15%.

Furthermore, the dataset is mostly balanced with each class having approximately *8000* instances, except for the *other_cyberbullying* class. This distribution is illustrated in Figure 1.

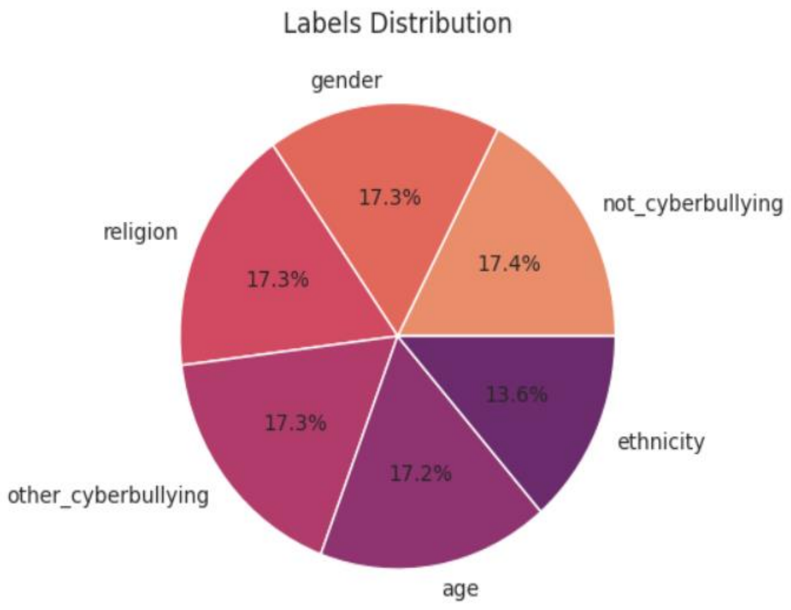


*Figure 1*

After pre-processing, the total number of tokens in the dataset is *506953*, *34432* of which are unique. This is our **vocabulary size**. Additionally, the **average length** of a tweet is *11* tokens, and the **maximum length** is *314* which is represented by the bar chart in Figure 2.

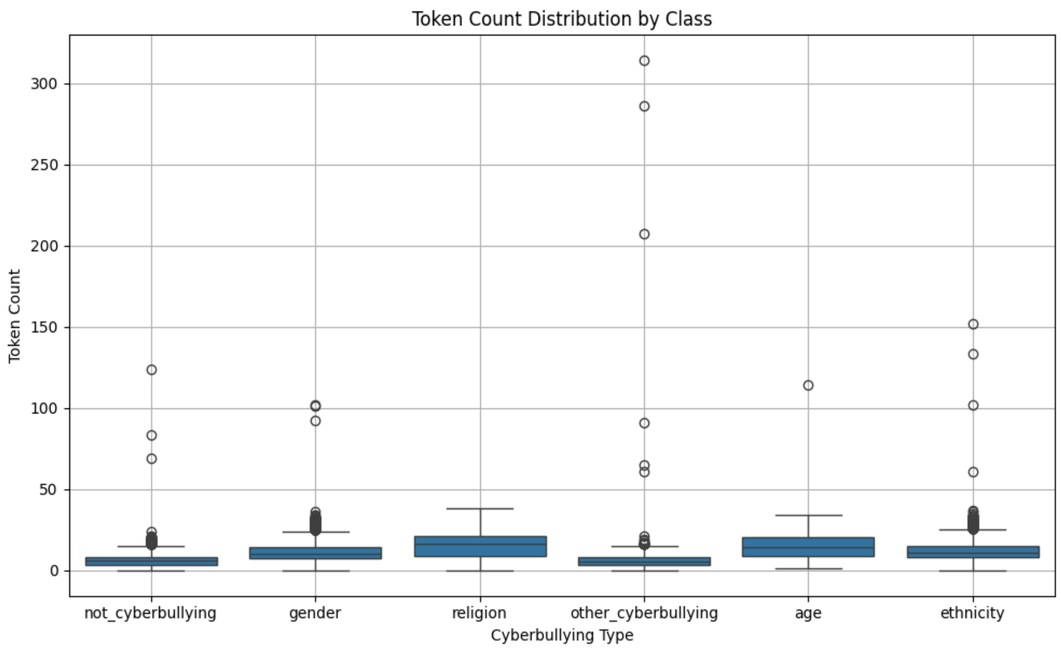


*Figure 2*

Moreover, the top 5 unigrams associated with each category according to the PPMI matrix are represented in *Figure 3*.

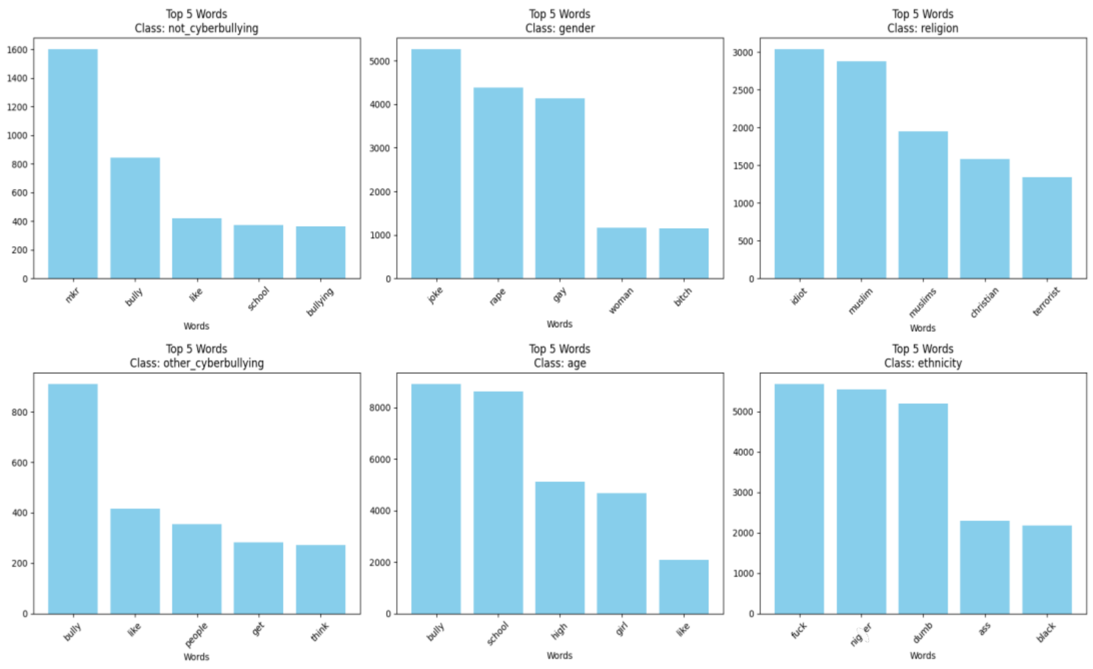


*Figure 3*

The word clouds per category, which allow a closer inspection of unigrams associated with each category, are illustrated in *Figure 4*.

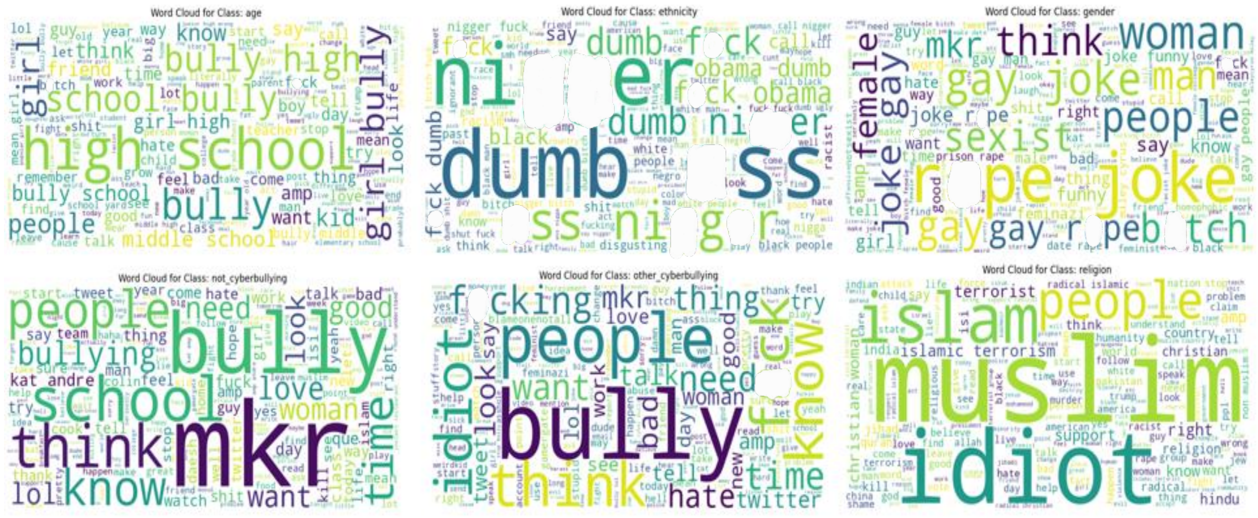


*Figure 4*

Similarly, the top bigrams can be observed in the word cloud in *Figure 5*.

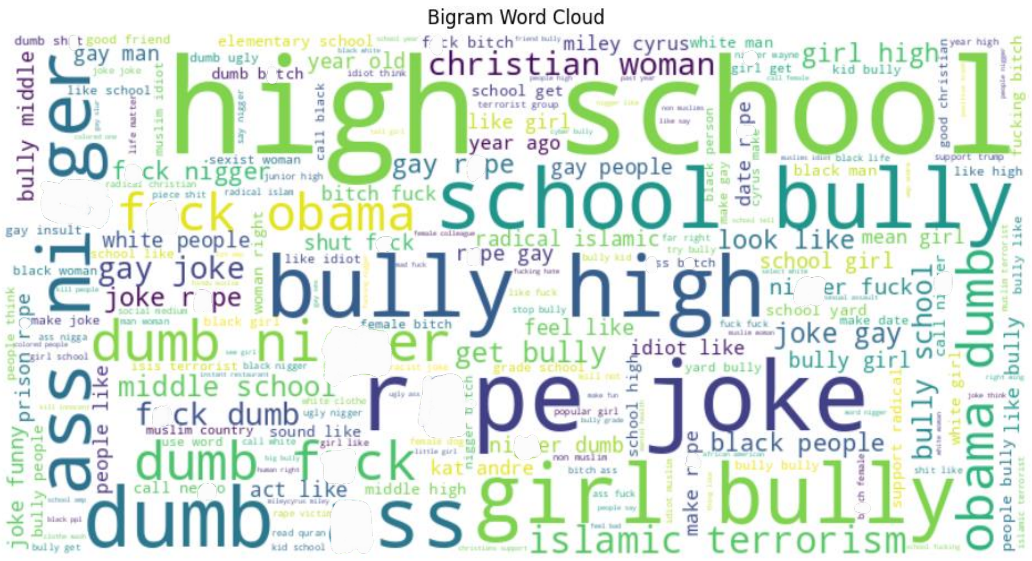


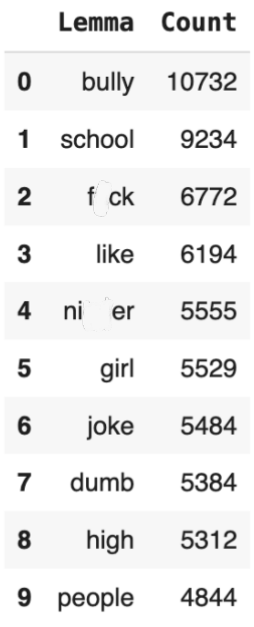
*Figure 5*

## Methodology

Prior to any feature extraction procedure, every tweet is passed through the **preprocessing** section, where *SpaCy* tokenizer is utilized for the following:

- *Lowercasing* `(token.lemma_lower)`
- *Alphabetic* filtering `(token.is_alpha)`
- *Stopword* removal `(token.is_stop)`
- Minimum *length* constraint `(len(token.lemma) >= 3)`

These steps ensure that the final token set contains *meaningful* content words only. This is important since it improves computational efficiency and reduces *noise*. The top 10 tokens are visualized in *Figure 6*.



| | Lemma | Count |
|---|---|---|
| 0 | bully | 10732 |
| 1 | school | 9234 |
| 2 | f[redacted]ck | 6772 |
| 3 | like | 6194 |
| 4 | ni[redacted]er | 5555 |
| 5 | girl | 5529 |
| 6 | joke | 5484 |
| 7 | dumb | 5384 |
| 8 | high | 5312 |
| 9 | people | 4844 |

*Figure 6*

Having completed the pre-processing step, we explore several natural language processing methods and compare their performance.

We start by constructing two **bag of words** representations: Positive Pointwise Mutual Information **(PPMI)** matrix and Term Frequency Inverse Document Frequency **(TF-IDF)** vectors.

A PPMI matrix computes the word and context co-occurrence counts across the whole corpus according to the following formula:

$$PPMI(w,c) \ = max(log\frac{P(w,c)}{P(w)\cdot P(c)}, 0)$$

This method emphasizes the *semantic associations* between words.

TF-IDF vectors, on the other hand, use the following formula to capture which words hold more *importance* for a tweet in the given training corpus:

$$TF - IDF(w,d) \ = \ TF(w,d)\cdot log\frac{N}{DF}$$

For our vocabulary, the words with the strongest associations found by the PPMI method and those found by the TF-IDF representation and cosine similarity are represented in the heatmaps in *Figure 7*.

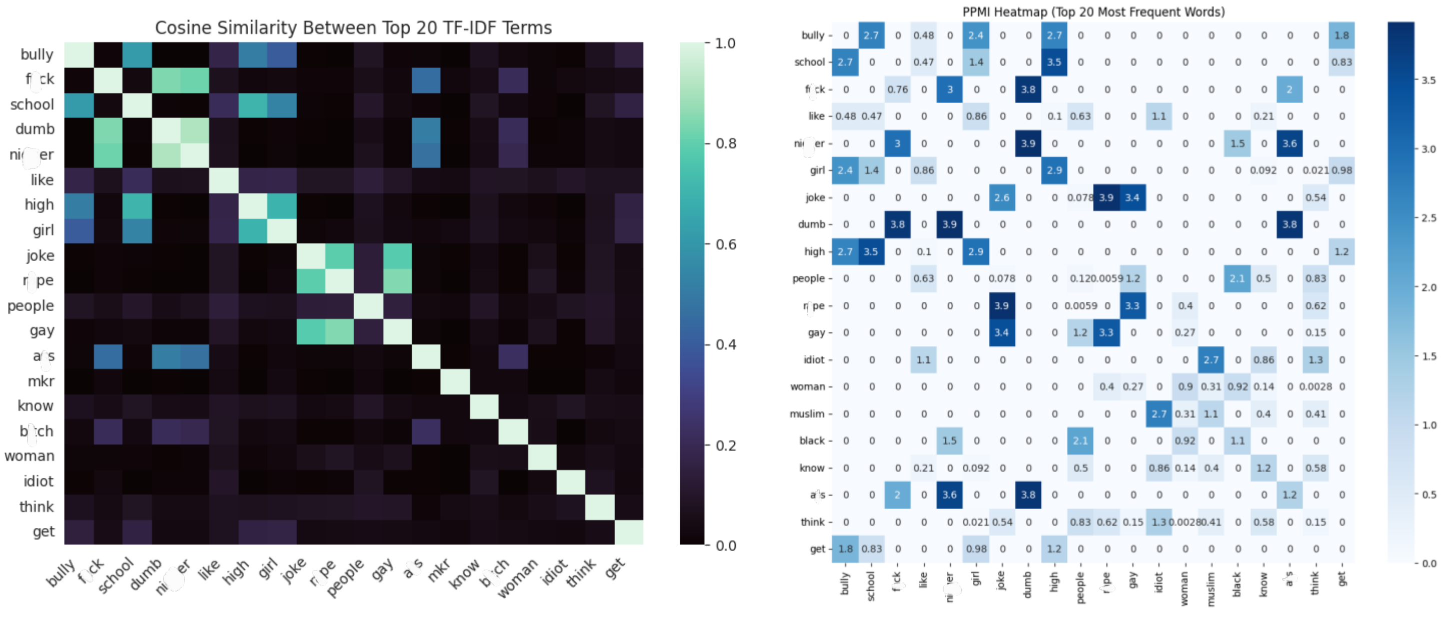


*Figure 7*

The words with the highest TF-IDF scores, as well as the top terms per class, which are likely to be distinctive in classification, are illustrated in *Figure 8*.

As the baseline model, we train a **multinomial logistic regression** classifier on each sparse representation. We compare the performances of the models using *accuracy* and *f1-scores* as metrics.

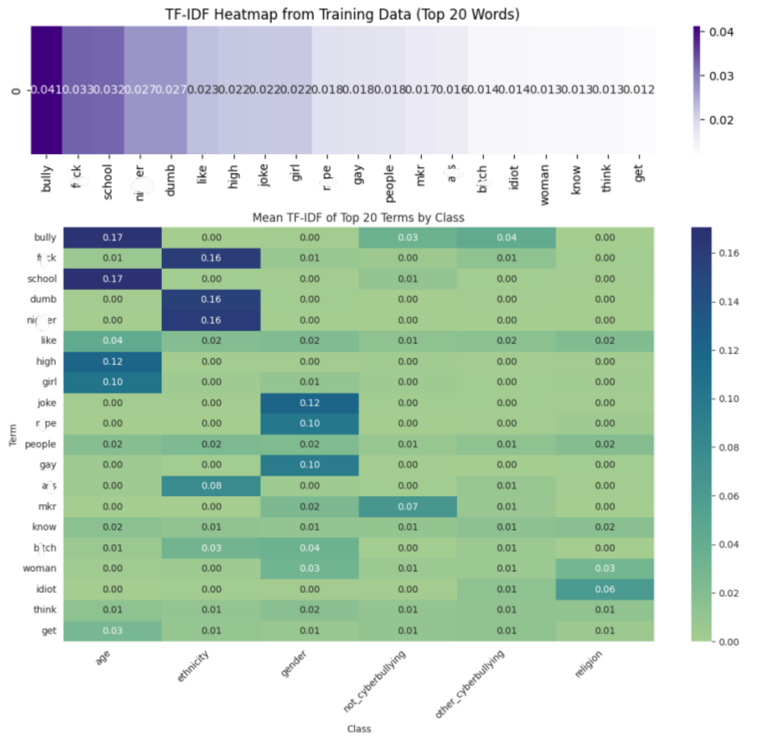


*Figure 8*

Having completed the **sparse word representations**, we move on to dense word embeddings. **Dense word embeddings** capture richer semantic and contextual information since they project words into a vector space where proximity reflects distributional patterns, thereby capturing. Moreover, they are more computationally efficient since they utilize less dimensions. We compare the performance of a pre-trained embedding **GloVe** (Global Vectors for Word Representation) with word level tokenization, and a **custom** dense word embedding with the **sub-word** level **tokenization**, on an **LSTM** (Long Short Term Memory) based model.

Last but not least, we utilize **contextual word embeddings** in our model. In contrast to dense word embeddings, these embeddings do not have a fixed vector. Instead, they have **dynamic meanings** for each word in each context. In this project, we use the pretrained **Distil BERT** model (Distilled Bidirectional Encoder Representations from Transformers) with the classifier head, which attends to the entire sequence from both directions.

In conclusion, we compare 5 different models with different text representation techniques - namely PPMI, TF-IDF, GloVe, custom embedding, and BERT. We analyze the strengths and weaknesses of each model and evaluate their test performances with metrics such as accuracy, recall, precision, and F1 score.

## Result and Analysis

We begin by comparing two sparse bag-of-words representations, namely PPMI and TF-IDF, each paired with a multinomial logistic regression classifier. The corresponding F1 scores, which balance precision and recall, and accuracy **scores per class** are illustrated in *Figure 9*.

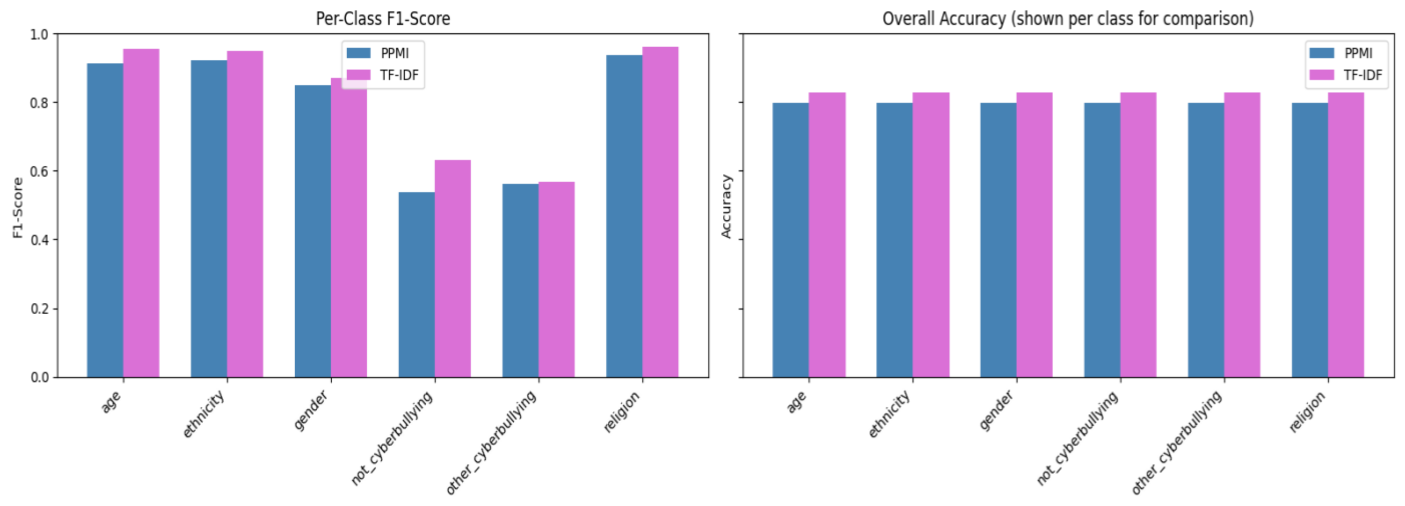


*Figure 9*

We see that the TF-IDF representation consistently outperforms the PPMI across all categories. The richer document level weighting of TF-IDF appears to be more effective in highlighting discriminative words. *Figure 10* presents the **confusion matrices** for both representations.

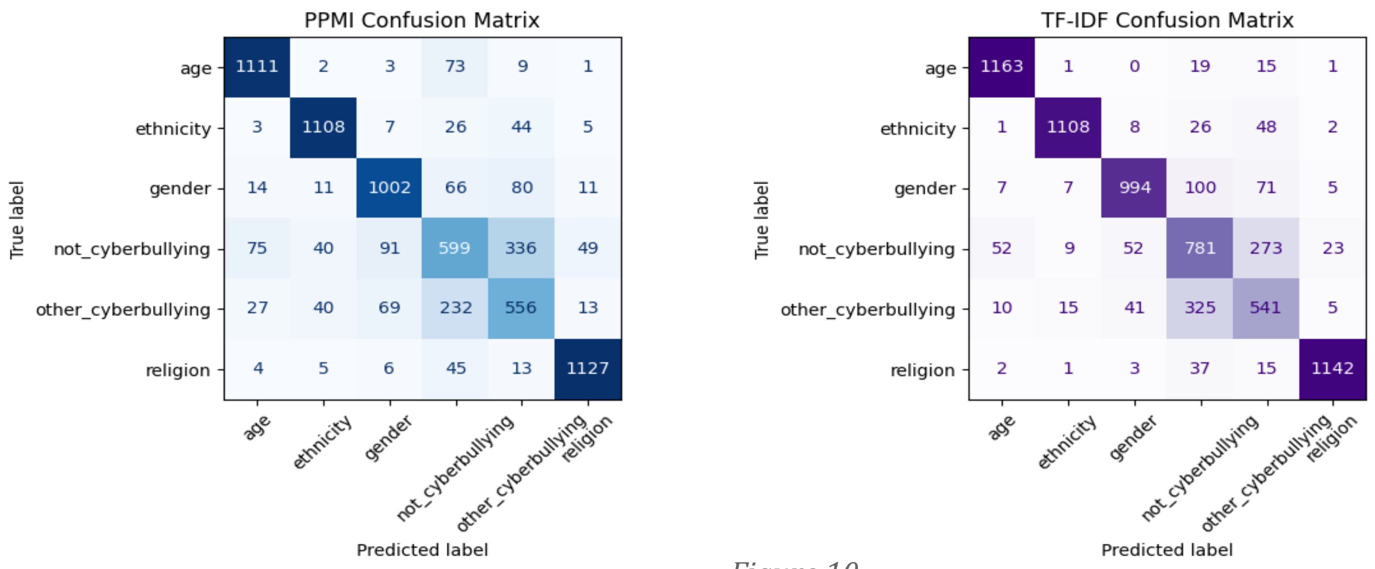


*Figure 10*

Notably, both representations struggle significantly with *not_cyberbullying* and *other_cyberbullying* classes and often confuse them.

Given the TF-IDF representation's superior baseline, we focus on this pipeline for further tuning. We compare several classifiers, and settle on **XGBoost** to proceed to tuning the model with the following **hyper parameters**:

- Number of estimators `(100)`
- Maximum depth `(10)`
- Learning rate `(0.1)`
- L1 regularization `(1)`

The best model found by the search algorithm is then used to plot the **learning curve** given in *Figure 11*.

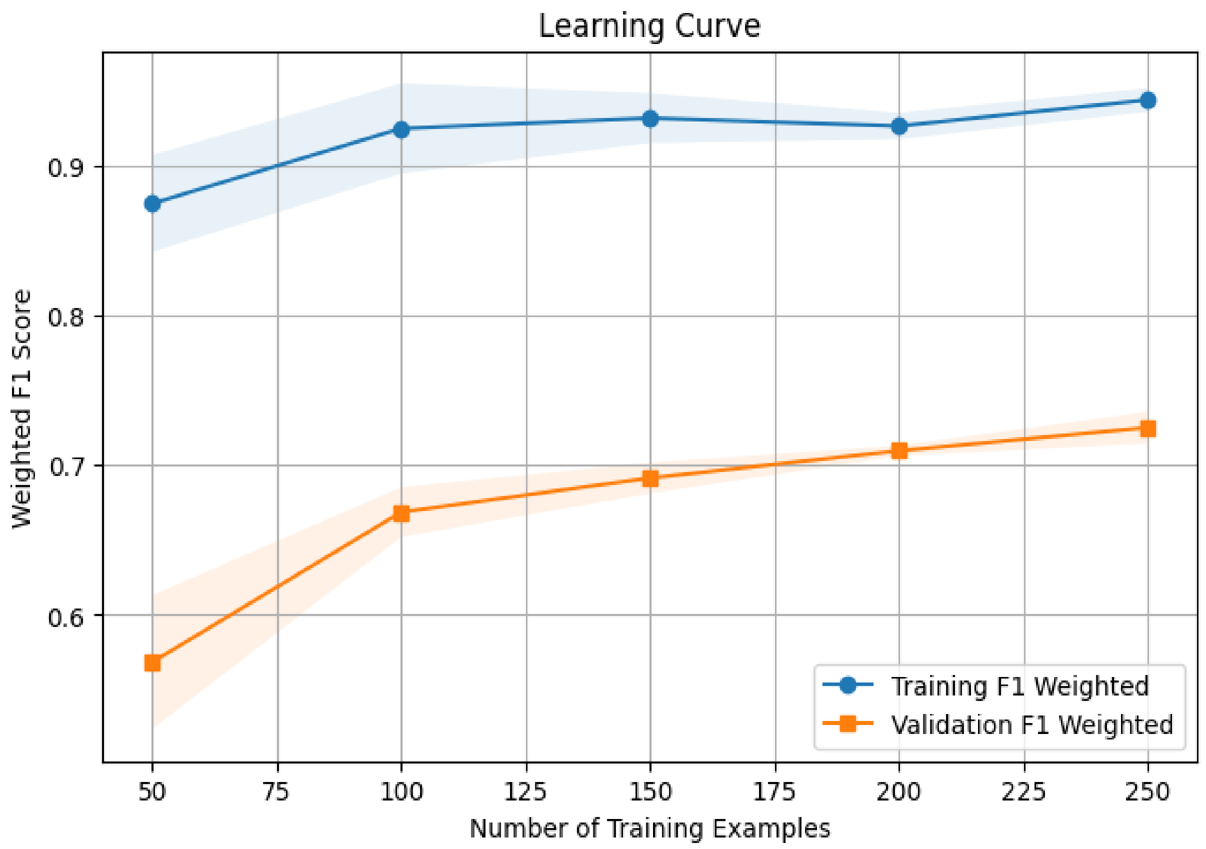


*Figure 11*

We observe that the model has a significant overfitting issue. Therefore, we plot a **validation curve** showing the effect **of L2 Regularization** in *Figure 12*.

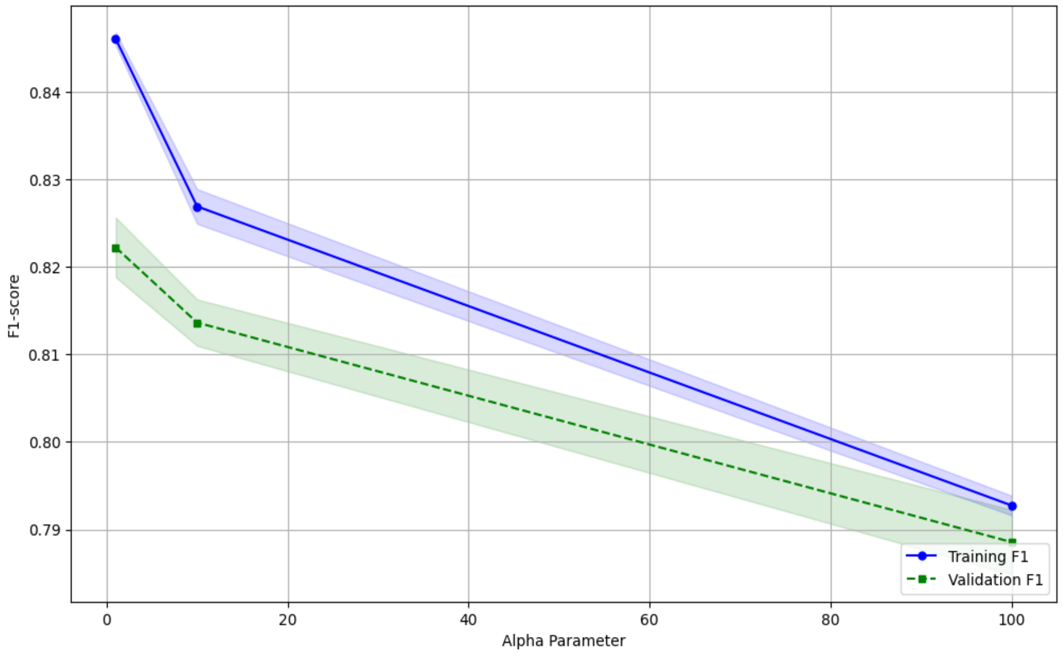
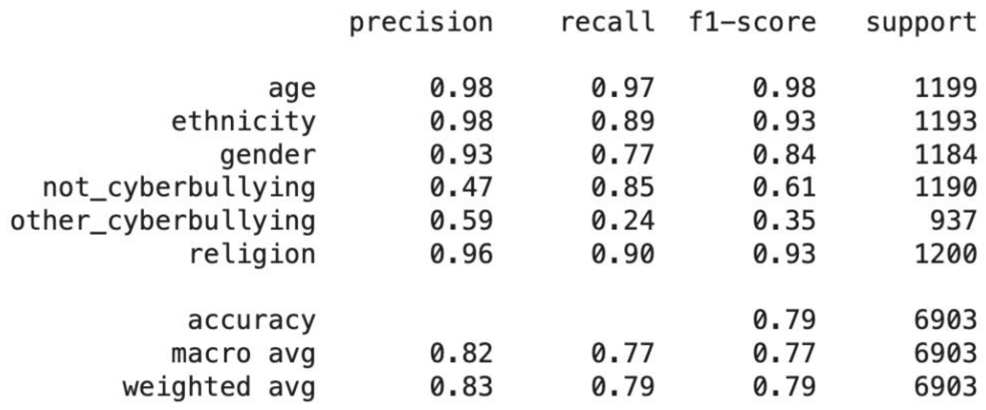


*Figure 12*

In the end, we obtain the **classification report** given in *Figure 13*, with the model achieving 79 % accuracy.



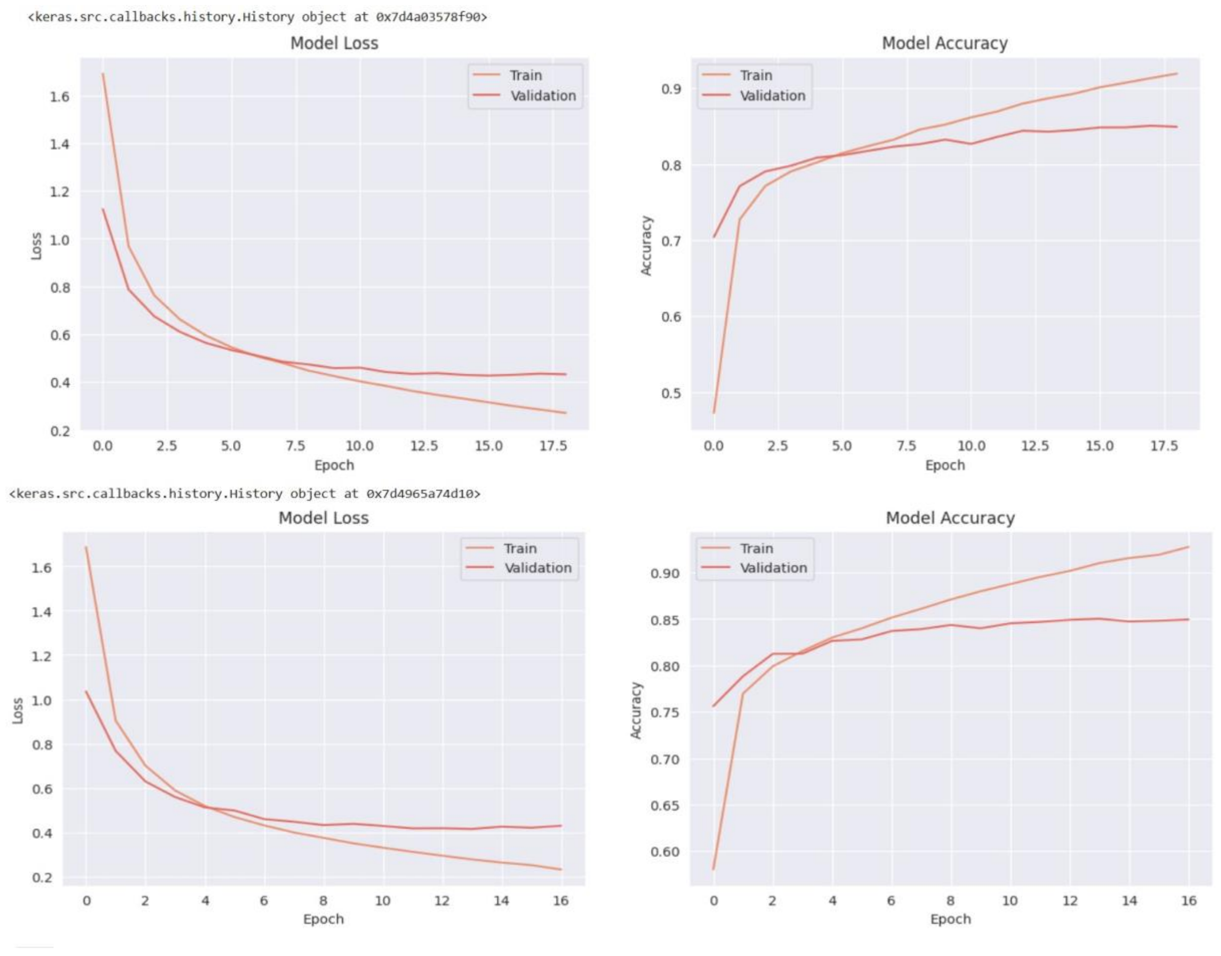
```
                    precision    recall  f1-score   support

               age       0.98      0.97      0.98      1199
         ethnicity       0.98      0.89      0.93      1193
            gender       0.93      0.77      0.84      1184
 not_cyberbullying       0.47      0.85      0.61      1190
other_cyberbullying       0.59      0.24      0.35       937
          religion       0.96      0.90      0.93      1200

          accuracy                           0.79      6903
         macro avg       0.82      0.77      0.77      6903
      weighted avg       0.83      0.79      0.79      6903
```

*Figure 13*

Secondly, we replace sparse features with pre-trained GloVe embeddings fed into a **bidirectional LSTM**. The model **losses** and **accuracies** over numerous **epochs** are represented in *Figure 14*.



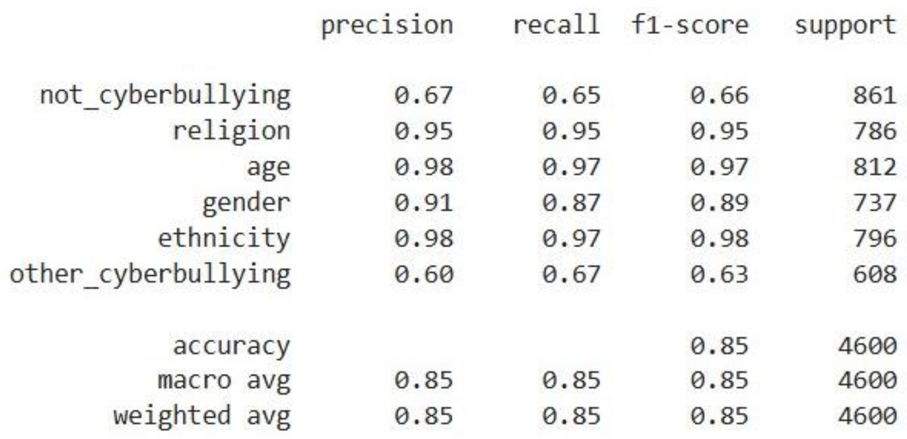
```
<keras.src.callbacks.history.History object at 0x7d4a03578f90>
```

```
<keras.src.callbacks.history.History object at 0x7d4965a74d10>
```

*Figure 14*

We experimented with two LSTM-based networks—one with 32 units and another with 64—using a deliberately minimal architecture because anything more complex began overfitting right away. To combat **overfitting**, we added L2 **regularization** on the dense layer, introduced **dropout**, and lowered the **learning rate**.

Despite these adjustments, the models still show signs of overfitting, though they seem to stabilize, with the largest errors occurring on the *not_cyberbullying* and *other_cyberbullying* categories. Full classification metrics are presented in *Figure 15*.



```
                    precision    recall  f1-score   support

 not_cyberbullying       0.67      0.65      0.66       861
          religion       0.95      0.95      0.95       786
               age       0.98      0.97      0.97       812
            gender       0.91      0.87      0.89       737
         ethnicity       0.98      0.97      0.98       796
other_cyberbullying       0.60      0.67      0.63       608

          accuracy                           0.85      4600
         macro avg       0.85      0.85      0.85      4600
      weighted avg       0.85      0.85      0.85      4600
```

*Figure 15*

In *Figure 16,* we visualize the confusion matrix obtained using GloVe LSTM.

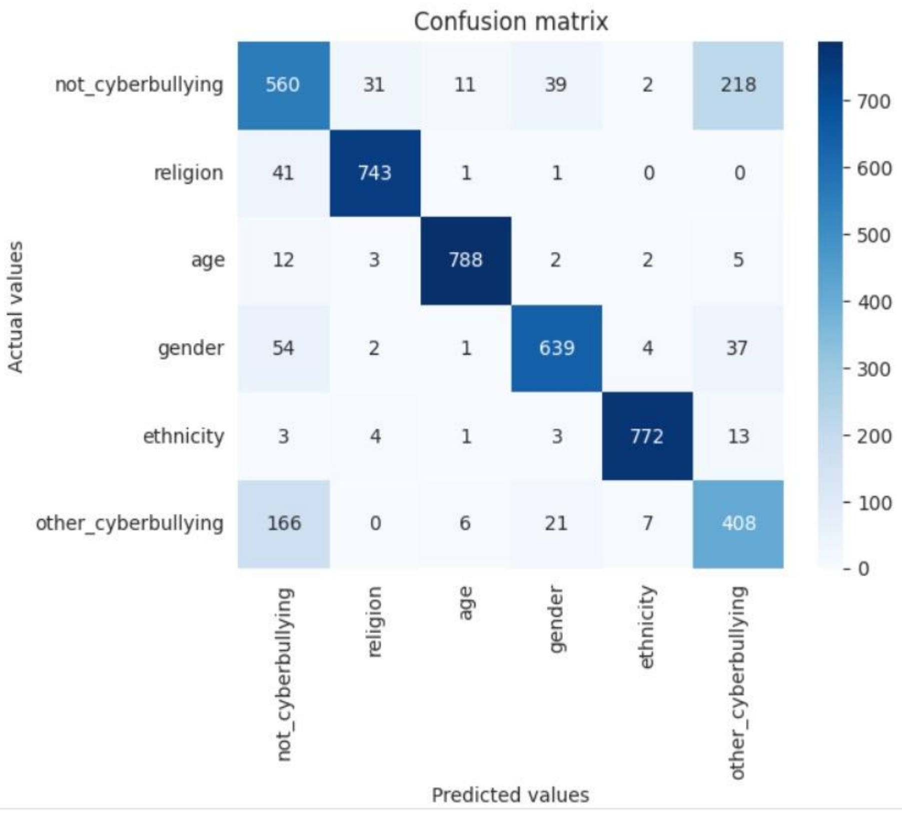


*Figure 16*

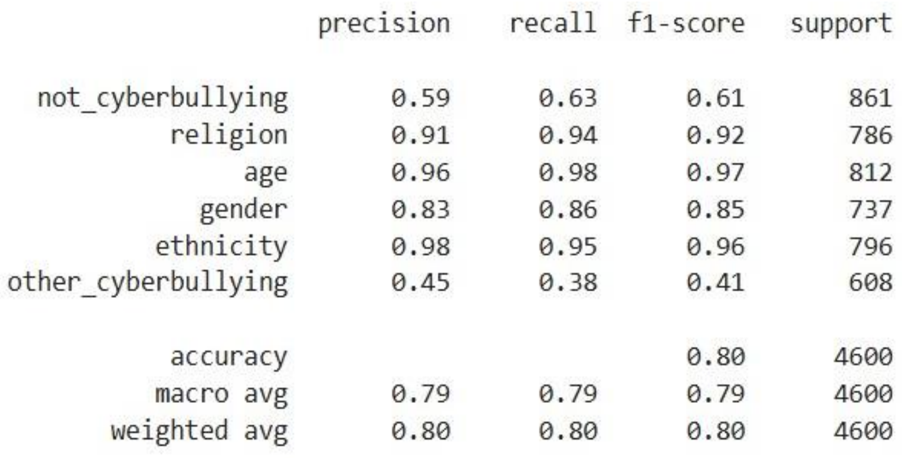
*Table 1* compares the GloVe against the tuned TF-IDF baseline.

| | Precision | | Recall | | F1 Score | |
|---|---|---|---|---|---|---|
| | GloVe | TF-IDF | GloVe | TF-IDF | Glove | TF-IDF |
| Age | 0.98 | 0.98 | 0.97 | 0.97 | 0.97 | 0.98 |
| Ethnicity | 0.98 | 0.98 | 0.97 | 0.89 | 0.98 | 0.93 |
| Gender | 0.91 | 0.93 | 0.87 | 0.77 | 0.89 | 0.84 |
| Not_cyberbullying | 0.67 | 0.47 | 0.65 | 0.85 | 0.66 | 0.61 |
| Other_cyberbullying | 0.60 | 0.59 | 0.67 | 0.24 | 0.63 | 0.35 |
| Religion | 0.95 | 0.96 | 0.95 | 0.90 | 0.95 | 0.93 |
| Weighted Average | 0.85 | 0.83 | 0.85 | 0.79 | 0.85 | 0.79 |

*Table 1*

It can be noted that compared to the traditional NLP approaches, the neural network approach performs significantly better. Specifically, it handles the confusion between the classes *not_cyberbullying* and *other_cyberbullying*, as it obtains an F1 score of 0.63 in the *other_cyberbullying* class, compared to 0.35 from TF-IDF. Even though this is a huge improvement, there is still room left for improvement.

In an attempt to improve the model, we use a custom embedding model with sub-word level tokenization. This model is expected to perform better since sub-word tokens handle out of vocabulary words much more efficiently. The results obtained by this model are summarized in *Figure 17*.



```
                     precision    recall  f1-score   support

   not_cyberbullying       0.59      0.63      0.61       861
            religion       0.91      0.94      0.92       786
                 age       0.96      0.98      0.97       812
              gender       0.83      0.86      0.85       737
           ethnicity       0.98      0.95      0.96       796
 other_cyberbullying       0.45      0.38      0.41       608

            accuracy                           0.80      4600
           macro avg       0.79      0.79      0.79      4600
        weighted avg       0.80      0.80      0.80      4600
```

*Figure 17*

We observe that this model is worse than the previous one, likely because there isn't enough data to effectively train the custom embeddings. It also begins to overfit much sooner than the GloVe based version.

Lastly, we leverage a pretrained **DistilBERT** with a classification head. This model performed the **best** out of all the previous approaches, with as much accuracy as 88% on the test set. The **training loss**, **validation loss**, and the **validation accuracy over time** of this model is plotted in *Figure 18*.

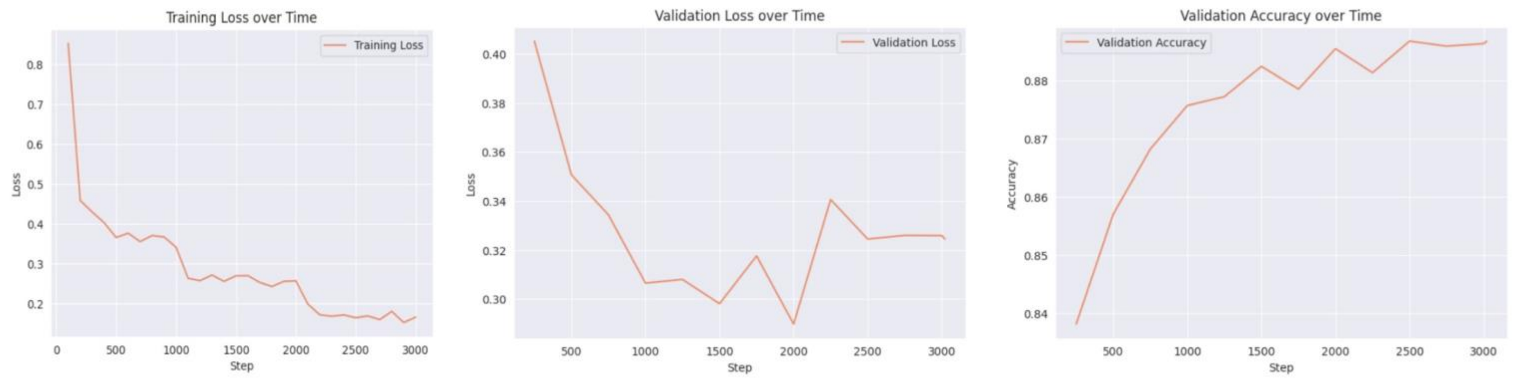
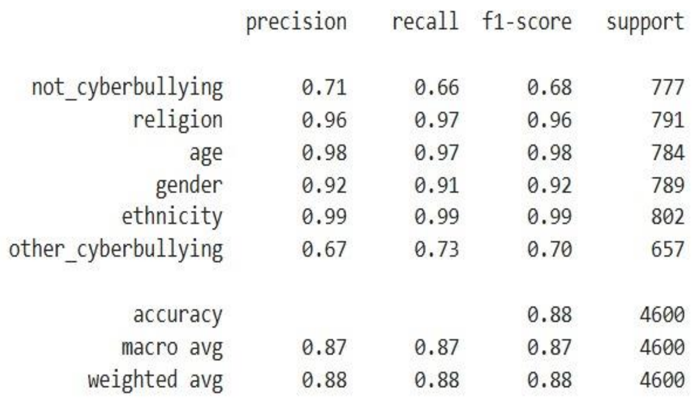


*Figure 18*

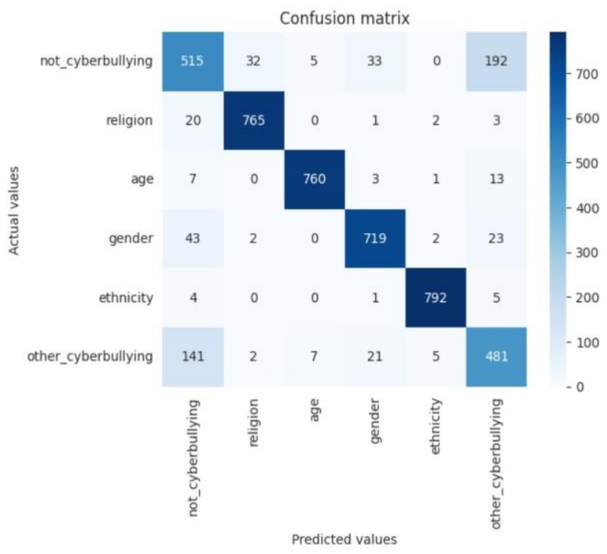
Upon closer inspection of the curves, we notice that after the second epoch, the model goes into overfitting. The corresponding **classification report**, along with the **confusion matrix**, is displayed in *Figure 19*.



```
                    precision    recall  f1-score   support

  not_cyberbullying      0.71      0.66      0.68       777
           religion      0.96      0.97      0.96       791
                age      0.98      0.97      0.98       784
             gender      0.92      0.91      0.92       789
          ethnicity      0.99      0.99      0.99       802
other_cyberbullying      0.67      0.73      0.70       657

           accuracy                          0.88      4600
          macro avg      0.87      0.87      0.87      4600
       weighted avg      0.88      0.88      0.88      4600
```



| | not_cyberbullying | religion | age | gender | ethnicity | other_cyberbullying |
|---|---|---|---|---|---|---|
| not_cyberbullying | 515 | 32 | 5 | 33 | 0 | 192 |
| religion | 20 | 765 | 0 | 1 | 2 | 3 |
| age | 7 | 0 | 760 | 3 | 1 | 13 |
| gender | 43 | 2 | 0 | 719 | 2 | 23 |
| ethnicity | 4 | 0 | 0 | 1 | 792 | 5 |
| other_cyberbullying | 141 | 2 | 7 | 21 | 5 | 481 |

*Figure 19*

Most notably, the distinction between *not_cyberbullying* and *other_cyberbullying* classes has improved significantly. This is quite crucial since it dampens the main factor that undermines the model performance.

Since this issue is not fully solved, we obtain a **random sample** of the classifications by the BERT model and **manually analyze** them.

We observe that the dataset contains mistakes in the labels classified by humans as shown in *Table 2*.

| Tweet Text | True Label | Predicted Label |
|---|---|---|
| RT @anne_theriault: And there goes any respect I had for Emma Stone RT @direhellswan: what in the fuck is this http://t.co/EFZEUTXGT6 | not_cyberbullying | other_cyberbullying |
| Kat and Andre better fuck off this show quickly. Awful people #mkr | gender | not_cyberbullying |
| I'm all for equality, but FemiNazi's need to chill the fuck out, and slapping a woman is not fucking cool. Even if she slaps you first. | other_cyberbullying | gender |
| @ahtweet yup. i went in the water. | other_cyberbullying | not_cyberbullying |

*Table 2*

Furthermore, it can be argued that some posts are ambiguous regarding which category they should belong to. This is illustrated in *Table 3*.

| Tweet Text | True Label | Predicted Label |
|---|---|---|
| Is it a pony?! Oh fuck off. | not_cyberbullying | other_cyberbullying |
| i was referring to u saying josie words doesn't sit right with me so i added onto it. im guessing english isn't ur first language? 'find something else?' u just admitted to being an ableist. chile how does it feel to be one? i wanna know. | ethnicity | not_cyberbullying |
| @JabberwockySR i need a ticket #, the tweet in question, and i can sometimes escalate. not all the time, though. | other_cyberbullying | not_cyberbullying |

*Table 3*

Lastly, it can be seen that the model misclassifies some sentences due to **natural nuances** in human communication. Some swear words contain no malicious **intent**, but the model struggles to differentiate between actual bullying and harmless profanity. This is illustrated in *Table 4*.

| **Tweet Text** | **True Label** | **Predicted Label** |
|---|---|---|
| @KingHendo95: @VonDreaam naw it's my old tape. new cover . my upcoming mixtape is #KingOfTheHill Yeah I'm waiting on that hoe Mayne! | other_cyberbullying | gender |
| Throwback to when vivek offered to show me around SU to shut me up. https://t.co/DoMBsoijp3 | not_cyberbullying | other_cyberbullying |
| @Vodage @GameNinja08 @KaitlynBurnell ah, okay. | not_cyberbullying | other_cyberbullying |

*Table 4*

Overall, we obtain a comprehensive view of our models' strengths and weaknesses by combining quantitative metrics, visual analyses, and manual case studies.

## Conclusion

In conclusion, our project systematically **explores** a variety of **word representation** techniques for a **single label multi-class classification** task on Twitter data. Beginning with rigorous **data preprocessing** - including tokenization, lemmatization, stop word removal, and noise filtering- we implemented **PPMI, TF-IDF**, **pre-trained dense word embeddings**, **custom** dense word embeddings, and **contextualized word embeddings**. Each model was carefully tuned and evaluated on **held-out test splits**. Our findings highlight that more **expressive representations** yield better performance and capture more subtle linguistic features. We then discussed the limitations of the project.

One group member led the exploratory data analysis, implemented the pre-processing pipeline (stop word and noise removal, tokenization, and lemmatization) and developed the sparse feature baselines with a traditional NLP approach.

The other group member employed dense and contextual word embedding models, setting up GloVe, designing and training a BiLSTM classifier, integrating BERT and conducting comparative experiments.

The most difficult step was robust pre-processing of the text, since social media posts contain many abbreviations (ASAP), and noise such as usernames (@Trigger_Check), hashtags (#NeverSayNever), and technical tags (rt). Furthermore, processing sparse representations required immense processing power. This was a serious limitation and had to be worked around. As for the LSTM part, a difficult step was to create an efficient pipeline which would allow reusing the code for similar purposes.

## AI policies

The following prompt were used in the development stage:
"How to check how the label encoder encodes different classes, wrapped in a data frame to view it as a table?"
"Show an example of a sentence and its assigned class"
"Make the data frame show the whole text, because the sentences cannot be displayed fully."
"Is there any function in sk-learn to show all the evaluation metrics in a concise way?"
"Help me display this code in a table: `classification_report(y_test_int, y_pred, target_names=original_class_names, output_dict=True)`"
"Can you put these charts in 2 rows, side by side:

```
def plot_top_words_per_class(df, lemma_col='lemmas', label_col='label',
top_n=5):
      labels = df[label_col].unique()
      for label in labels:
      tokens = [token for tokens in df[df[label_col] == label][lemma_col]

      for token in tokens]
      freq_dist = Counter(tokens).most_common(top_n)        words, counts =
      zip(*freq_dist)
      plt.figure(figsize=(8, 4))
      plt.bar(words, counts)
      plt.title(f"Top {top_n} Words for Class: {label}") plt.xlabel("Words")
      plt.ylabel("Frequency") plt.xticks(rotation=45) plt.tight_layout()
      plt.show() plot_top_words_per_class(df, lemma_col='lemmas',
      label_col='cyberbullying_type', top_n=5)
```
"

Throughout the project, we leveraged AI assistance to streamline our development workflow. We found it to be especially useful in:

- Debugging support: By describing error messages and code snippets, we obtained detailed explanations of why an error might occur and potential ways to solve the problem
- Code refactoring and formatting: AI assistance was helpful in organizing and reformatting the code snippets, especially enhancing style consistency

On the other hand, we have experienced some occasional hallucinations, along with some overly generic suggestions that were not suited for the problem at hand.