**Course of Text Mining and Search - Professor G. Pasi, M. Viviani - Academic year 2024/2025**

# On Hate Speech Detection and Classification

Carlo Arpini 918543 - Emmanuele Lotano 918608 - Chiara Mariani 918354

*Università degli Studi di Milano-Bicocca, CdLM Data Science*

## Abstract

In this study we analyze hate speech on social media through a dataset of tweets, with the aim of creating a classifier able to intercept hate speech. Preprocessing techniques are used to clean the data and text representations such as BoW, GloVe, and sBERT are employed; through sBERT we also gather evidence on how, given contextual meaning to sentences, the different classes of hate speech emerge from our dataset, validating ground truth and showcasing that multi-class classification is in some sense an emergent property of hate speech.

## Contents



## 1 Introduction

As social media use becomes increasingly prevalent across all age groups, it has become an essential medium for day-to-day communication. However, this freedom also means that hate speech has unfortunately become ubiquitous. The relative anonymity of the Internet makes this phenomenon more difficult to stop than traditional bullying, as an example. The COVID-19 pandemic has exacerbated this issue, with increased screen time and decreased face-to-face interactions leading to increased cyberbullying and hate speech.

Hate speech can be defined as abusive or threatening speech or writing that expresses prejudice on the basis of ethnicity, religion, sexual orientation, or similar grounds [1]. Moreover, it can be hard to see who is harming the victim, who is being harmed, and how serious it is, because it is all done online. Some statistics we can look at, such as those on cyberbullying, are alarming: 36.5% of middle and high school students have experienced cyberbullying and 87% have observed it with effects ranging from decreased academic performance to depression and suicidal thoughts.

Current methods to combat hate speech include teaching "Internet street smarts" (i.e. acts of cyberbullying can be prevented through teaching the benefits and risks that Internet can have), looking for warning signs and counseling. Major social media platforms have policy guidelines and passive reporting mechanisms, but they lack active anti-hate speech functions. As a consequence, an estimated 90% of cyberbullying activities go unreported, highlighting the need for more effective solutions.

In this work, we consider a dataset of tweets classified by cyberbullying types such as *age*, *ethnicity*, *gender*, *religion*. We believe that although the dataset itself classifies such tweets as "cyberbullying", it would be more correct to use a more general term of "hate speech", because of their nature, which might not be "aimed" at a specific victim that is being bullied. First, we proceed by making some preprocessing steps to obtain cleaned and usable tweets for our goals. Then, after obtaining three different and more accurate document representations with BoW, GloVo and sBERT, we decide to apply text classification and clustering hoping to gain insights into the patterns and characteristics of hate speech.

## 2 Preprocessing

Our dataset [2] contains 46.017 tweets that are classified based on the hate speech type, which can be: *age*, *ethnicity*, *gender*, *religion*, *other cyberbullying* and *not cyberbullying*.

To effectively perform well in the tasks of our project, we first need to do some preprocessing activities that enable us to obtain the final clean dataset useful for analysis: an example are links, mentions and emoticons made with letters (e.g. :D) because they do not convey any additional meaning to our tweets; as such we can just remove them. Moreover we can do the same by removing the "RT" letters at the start that indicate that the tweet itself was a retweet of someone else's tweet: once again no meaning is lost with this step.

Another action we performed was to remove hashtag symbols. Most of the time this is correct because people insert hashtags between the words of a tweet (e.g. "I love #pizza") and thus we want to preserve the meaning, which would be lost were we to remove all hashtags entirely. In cases where hashtags are instead at the end of the sentence (e.g. "I love pizza #food #vlog"), they still add little noise to the meaning of the sentence itself and it is best to keep them. We decided to remove the # symbol and store each hashtag in a dedicated column - this way even such small noise can be accounted for.

Then, we noticed some html tags; we can simply substitute them with their meaning: "&amp" is replaced with "and", "&lt" is replaced with "<" and "&gt" is replaced with ">".

At this point, the main problem we notice is the one related to the language in which tweets are written. Since we need uniformity of language (starting with uniformity of alphabet) to perform our analysis, we must deal with this problem. We can use the *langid* library in Python [3] for this purpose as a reference; we know although it is not perfect and can fail. As such, we decided to try to correct false positives and false negatives: first we start by checking whether every word within the tweet is a valid word of the english language given a word vocabulary [4]. This will reveal false negatives: if a sentence is not recognized as English but is made up of English words, it is more than likely it's an English sentence. Then we can drop the remaining non-english tweets; some will be lost but most will be true negatives.

Then another thing we can do is check through all sentences classified as English ones for false positives; one way is to see if we have non-latin characters within the sentence; if we do, we assume it's likely a false positive and delete the character (again, we might lose some sentences but we can't work with non-latin characters). Before doing that though, it's best we substitute emojis and currencies into literal words. We remove numbers, the remaining punctuations, white spaces and empty sentences; lastly, we transform our tweets into lower case. By doing all this beforehand our "false positives" check will be easier. Only then, we can tackle the problem of false positives within the langid classification.

At this point, we can see how our dataset has changed with a plot of the distribution of hate speech classes after the preprocessing.

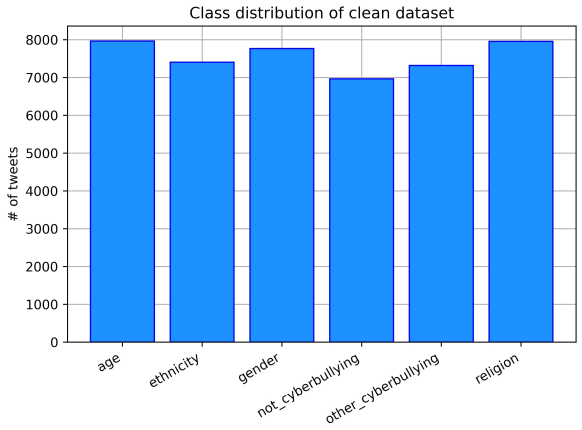


Fig. 1: Tweets distribution after preprocessing

To highlight all "lost" data from the original dataset we can also visualize the difference of tweets count by classes.

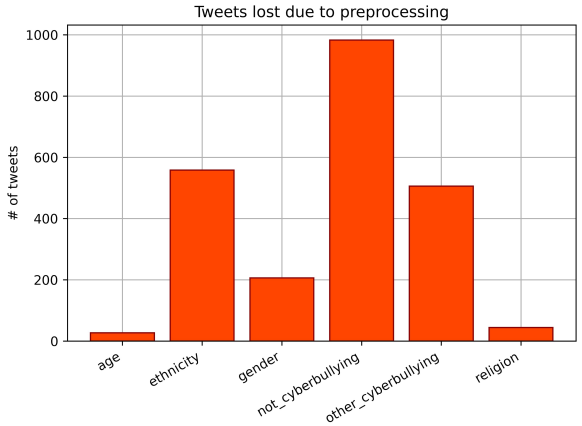


Fig. 2: Tweets loss after preprocessing

We see that unfortunately we have introduced some imbalance. To address if this will be a problem we can compute some measure of how actually imbalanced is the dataset. We chose to compute the IR Index for our dataset's classes, which is just the ratio of the most frequent to the least frequent class.

We consider an IR of 1 as a perfect balance and we obtain a IR of about 1.14, hence we have a slight imbalance that won't affect our analysis: we will not have to account for it with our choice of metrics for any model to be trained on our data.

We can also check out some other interesting plots, such as how long our tweets are.

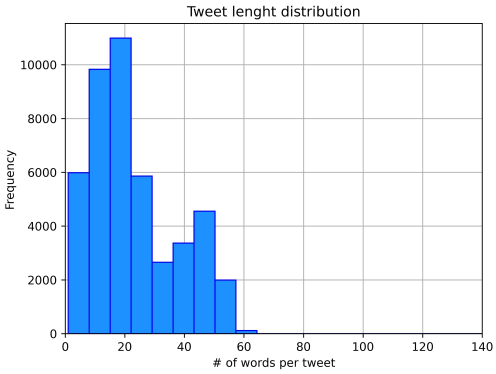


Fig. 3: Tweets length distribution

The distribution is highly skewed, with most tweets having fewer than 20 words, and frequency dropping sharply as word count increases.

As a final descriptive analysis, we look at a simple word cloud for all hate speech categories.

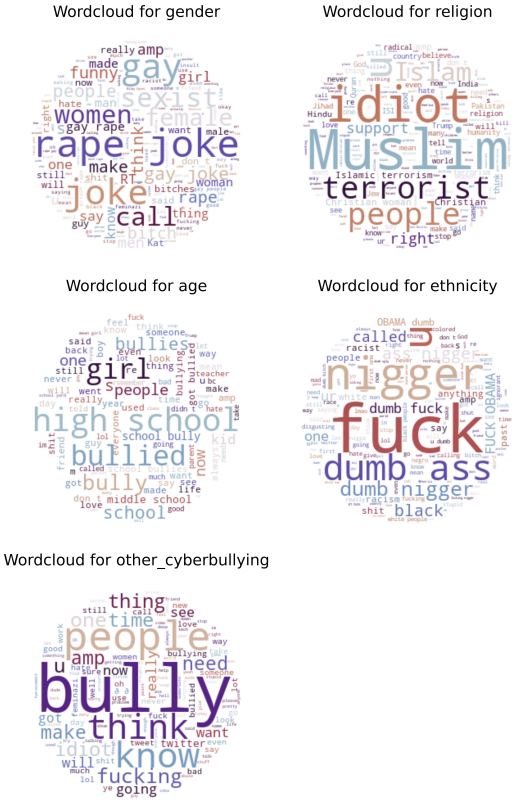


Fig. 4: Word cloud for each class

From the word cloud [Fig.4] it is apparent why we decided to use "hate speech" and not "cyberbullying": many of these insults can be directed not upon a single victim but rather a class of different people.

As final remark, we highlight that we did not remove stopwords. We chose not to remove stopwords, as some of the commonly agreed upon stopword lists contain words that may be helpful context for the cyberbullying detection task, such as the use of "not" or the unusually frequent use of male or female pronouns.

We also did not perform lemmatization for mainly two reasons: the first is that although it might help with some encodings such as BoW, it can definitely hurt others such as sBERT, which benefit from a more "natural" type of language, especially given the distribution of tweet lengths shows most tweets are composed of few words - lemmatization might make interpretation more difficult. The second is that any library we tried performed very poorly, probably also because of the colloquial nature of tweets.

Once the corpus of texts has been properly processed, we can move to text representation.

# 3 Tweets representation

The three techniques for document representation that we chose are Bag of Words (BoW), Global Vectors for Word Representation (GloVe) and sentence-Bidirectional Encoders Representation from Transformers (sBERT):

- BoW model is a simple and intuitive method for text representation. It involves converting a document (tweet in our case) into a vector of word frequencies. Each unique word in the document corpus (tweet collection) is represented as a feature and the value of each feature is the count of its occurrences in the document.

- GloVe is an unsupervised learning algorithm for generating word embeddings. Unlike BoW, GloVe captures semantic relationships between words by analyzing word co-occurrence statistics from a large corpus. It produces dense vector representations where words with similar meanings are located close to each other in the vector space.

- sBERT is a modification of the BERT model, designed specifically for sentence-level embeddings. This approach allows sBERT to capture the contextual and semantic meaning of sentences more effectively than traditional word embeddings.

The BoW representation is implemented using the CountVectorizer of the Sklearn package in Python [5]. We have to decide how many words keep in our bag of words vocabulary: too many and BoW model could overfit, too little and it could not capture correctly patterns. To do so, we plotted the frequency of each word and the results pointed in the direction of stopwords removal; for the aforementioned reasons, this is the only times we effectively removed stopwords, and we ended up limiting vocabulary size to 10000 (approximate value where word frequency had decreased between 4 and 3). For GloVe we use a pre-loaded word embeddings from Github [6] as base to create word embeddings of our dataset. Lastly, for sBERT we once again used a pre-loaded contextual word embeddings [7].

# 4 Text Classification

The first task we perform is classification. Our aim is to assign each tweet of our dataset to a specific class; so, to be more precise, we are considering the Single-Label Multi-Class classification, where we have multiple layers and each item belongs exactly to one class. The classes we take into account are those represented in the *cyberbullying_type* column in our dataset. We switch them to integers in order to enable the classifier to better understand the target class and so we have:

- *not cyberbullying*: 0
- *age*: 1
- *ethnicity*: 2
- *religion*: 3
- *gender*: 4
- *other cyberbullying*: 5

We experimented with three different machine learning classifiers. These include:

1. XGBoost: a powerful and efficient implementation of the gradient boosting framework. It builds an ensemble of decision trees in a sequential manner, where each tree corrects the errors of its predecessor.
2. Multi-Layer Perceptron: a type of artificial neural network composed of multiple layers of nodes.
3. Support Vector Machine: it works by finding the optimal hyperplane that best separates the data points of different classes in the feature space. We choose as kernel the *RBF*, Radial basis Function that allows us to deal with complex and non-linear relationships in data.

We decided to use traditional accuracy (adapted to multi-class classification) as a metric given that our dataset was not imbalanced, as shown in the preprocessing section.

The accuracy values obtained from cross validating our models provide meaningful insights into the performance of different document representations and classifiers in the context of text classification for cyberbullying detection. We can summarize the accuracy values obtained in the following table [Table 1].

| | **XGB** | **MLP** | **SVM** |
|---|---|---|---|
| **BoW** | 0.834 | 0.785 | 0.824 |
| **GloVe** | 0.768 | 0.774 | 0.796 |
| **sBERT** | 0.792 | 0.788 | 0.830 |

Table 1: Accuracy values for each classifier

Among the three representations, BoW demonstrates the highest accuracy when used with the XGBoost model, suggesting that this simple yet effective representation captures sufficient information for the task. However, its performance slightly drops with MLP, which might be due to BoW's sparse and high-dimensional nature.

The GloVe embeddings, designed to capture semantic relationships between words, performed comparably well across all three classifiers, with accuracy scores ranging from 0.77 to 0.80.

With sBERT embeddings we obtain consistent performance across classifiers, with the best accuracy achieved by SVM. Interestingly, sBERT slightly outperformed GloVe in most cases, indicating that sentence-level embeddings might capture context better in a classification task where tweets often consist of short and contextually rich text.

Up to this point, we have focused on the analysis of accuracy to evaluate the performance of our models. To do a more in-depth analysis, we can consider the classification report which is able to give us more precise information about the classification of tweets. In practical terms, it's as if the model just learns that some slurs are associated with a specific

class, which is not surprising, given we are looking at frequency when using BoW.

In figure [Fig.5] we can see the classification report obtained with the XGBoost classifier with the BoW representation, for which we have reached the highest accuracy.

XGBoost Accuracy: 0.834597072826662
XGBoost Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.63 | 0.41 | 0.50 | 1723 |
| 1 | 0.99 | 0.97 | 0.98 | 2012 |
| 2 | 0.98 | 0.97 | 0.98 | 1899 |
| 3 | 0.96 | 0.94 | 0.95 | 1974 |
| 4 | 0.91 | 0.83 | 0.87 | 1918 |
| 5 | 0.56 | 0.83 | 0.67 | 1816 |
| accuracy | | | 0.83 | 11342 |
| macro avg | 0.84 | 0.83 | 0.82 | 11342 |
| weighted avg | 0.85 | 0.83 | 0.83 | 11342 |

Fig. 5: XGBoost classification report with BoW

The thing that stands out most from this report is the fact that the model manages to classify tweets in classes 1, 2, 3 and 4 very well, while tweets in classes 0 and 5 are more difficult to classify. This can be easily interpreted as the tweets in the four core classes belong to well-defined classes such as *age*, *ethnicity*, *religion* and *gender*, while on the other hand classes 0 and 5 could contain a variety of very different tweets.

Specifically, this is what happens when combining BoW representation and XGBoost as classifier, but the behaviour is repeated more or less similarly for all analyzed combinations of document representations and classifiers.

SVM Accuracy: 0.8303650149885382
SVM Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.62 | 0.50 | 0.56 | 1798 |
| 1 | 0.96 | 0.97 | 0.96 | 1977 |
| 2 | 0.97 | 0.93 | 0.95 | 1875 |
| 3 | 0.95 | 0.96 | 0.95 | 1961 |
| 4 | 0.88 | 0.85 | 0.87 | 1926 |
| 5 | 0.59 | 0.74 | 0.66 | 1805 |
| accuracy | | | 0.83 | 11342 |
| macro avg | 0.83 | 0.83 | 0.82 | 11342 |
| weighted avg | 0.83 | 0.83 | 0.83 | 11342 |

Fig. 6: SVM classification report with sBERT

Taking a look at what the best performance of sBERT was [Fig.6], we also can highlight similar considerations: the model, although it encapsulates contextual embeddings, still falls for the easiest trick in the book of simply classifying based on slurs, and as such performs poorly in non well-defined classes. This highlights the needs for an explicit over-representation of those non-well-defined classes in a training dataset, or some other tailored technique to achieve results that actually use contextual meaning - the representation alone is not enough.

# 5 Clustering

The other task we have decided to perform to extract possible insights from our data is clustering. The main goal of clustering is to regroup documents based on similarity or other measures. Documents within a cluster should be similar and documents from different classes should be dissimilar. As output we do not have a certain label associated with each document, but the output is only a membership with respect to a certain class.

In our case, we use clustering to check whether the resulting clusters will match or not the original classification; this is interesting because it highlights both potential other types of hate speech we might be missing but also if we are being biased in our classification choice.

We decide to perform clustering using flat algorithms that start with a random partial partitioning and create a flat set of clusters without any explicit structure that would relate clusters to each other. In particular, we use the $k$-means algorithm, where we have to firstly assess the number of clusters $k$. To decide this value we used the Silhouette score, mainly because it does not rely on ground truths and it does not rely on the structure of the cluster not overlapping with other clusters or having a clear variance structure within its data.

We can consider both the Silhouette score and the fit time in seconds for different values of $k$ from 2 to 30. Since we do not care about fit time because our aim is on an insight on the "true" number of clusters, we chose the value of $k$ corresponding to the maximum of the Silhouette score. This leads usually to an higher value of $k$ and hence a lower value of the distortion metric, another way to assess $k$.

Now we can discuss the results we have obtain considering the three different document representations. To formally evaluate the performances of our models, we compute both the Silhouette score and the Adjusted Rand Index to investigate on the agreement with ground truth. The Silhouette coefficient is computed using the mean intra-cluster distance and the mean nearest-cluster distance for each sample and ranges from -1 to 1. Instead, the Adjusted Rand Index is an external evaluation criterion that needs information about the true cluster

assignments of the data and it is basically a count of how many tweets are classified "correctly" based on our ground truth, accounting for random chance.

## 5.1 BoW

Within the BoW documents representation, the Silhouette method as described above leads us to an optimal value of cluster $k$ equal to 7.

Performing the $k$-means algorithm, we can obtain the plot of frequency of tweets within each cluster.

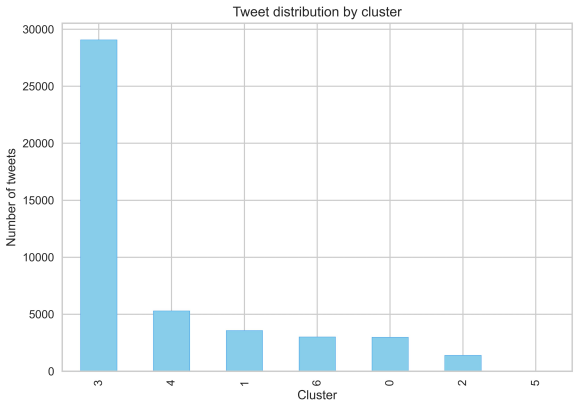


Fig. 7: Tweets distribution among clusters

From this graph [Fig.7], we notice that the number of tweets is not equally distributed among clusters. Moreover, there is a huge difference between, for example, cluster number 3, which contains more than 25.000 tweets and cluster number 2, which contains less than 5.000 tweets.

To formally assess the goodness of our clustering, we can compute the Silhouette score and the Adjusted Rand Index. We obtain a value of 0.02 for Silhouette, which is a value very close to zero suggesting that our data points are at the edges of the cluster or that the clusters are not well separated (most likely option). This can be interpreted as a consistent result with the known limitations of BoW: it does not capture semantic or contextual information treating words as independent of each other. We also obtain a value of 0.14 for Rand Index, which indicates that there is little agreement between the generated clusters and the actual data structure.

Lastly, we can visualize the clusters to see if they are well separated plotting them along the Principal Component Axes.

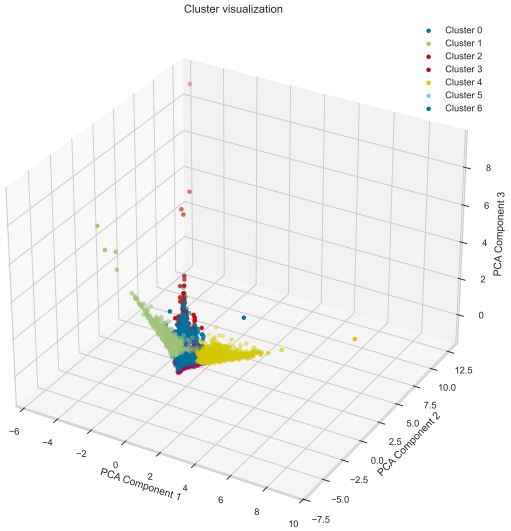


Fig. 8: Cluster visualization using BoW

The clustering results with BoW representation, visualized in figure [Fig.8], show a significant overlap between clusters. This confirms the findings of the quantitative metrics, in particular the low value of the Silhouette score and Rand Index, probably due to the BoW inability to capture semantic relationships and context of the data. This observations underlines the need to use richer representation, such as GloVe or sBERT, to obtain clustering results more consistent with the intrinsic structure of data.

## 5.2 GloVe

Applying the same reasoning also for the documents representation obtained with GloVe, we set $k$ equal to 4. The distribution of tweets among the four clusters is represented in the figure [Fig.9].

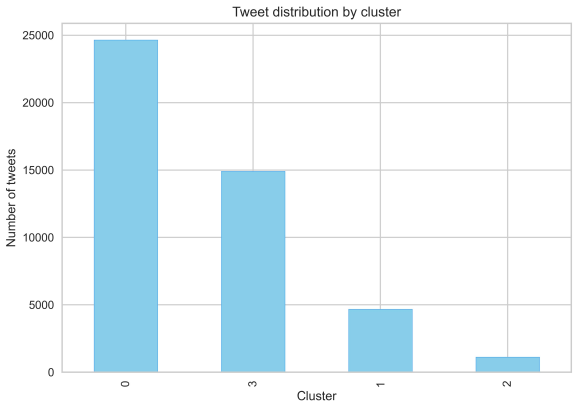


Fig. 9: Tweets distribution among clusters

As was the case before, we notice that the num-

ber of tweets is not equally distributed and we have a huge imbalance between clusters.

The value of the Silhouette score is 0.10, while the Rand Index is 0.07. These results with GloVe representation show a slight improvement of the Silhouette score, 0.10 compared to 0.02 of BoW, indicating a slightly better separation of the clusters. However, the Rand Index is lower than in BoW representation suggesting a reduced coherence with respect to the reference structure.

We can visualize what we have obtained on the Principal Component Axes.

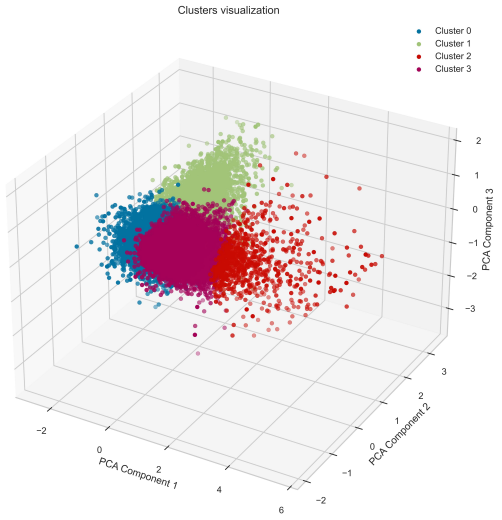


Fig. 10: Cluster visualization using GloVe

The clustering results with GloVe representation, visualized in figure [Fig.10], show a situation in which clusters are less overlapped compared to the visualization obtained with BoW [Fig.8], but obviously we still have some problems related to the reference structure. This might be due to the fact that, while capturing semantic information, GloVe uses static, non-contextualized word representations, which seems insufficient for our complex dataset.

## 5.3 sBERT

Using sBERT and the Silhouette score, the best value for $k$ is found to be 5. As in the other two cases, we can have a look at the plot of how many documents belong to each cluster.

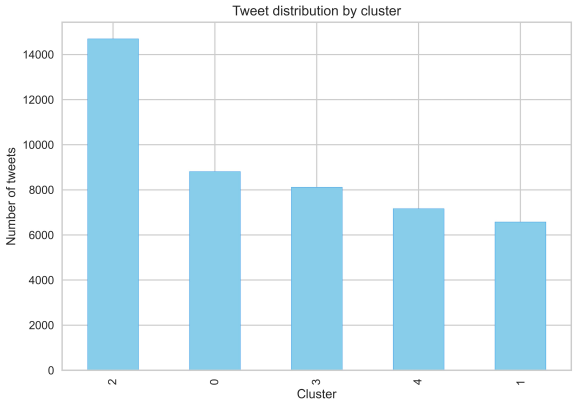


Fig. 11: Tweets distribution among clusters

We can already see from [Fig.11] that the situation has improved compared to previous cases and the tweets are much better distributed among the five clusters. Moreover, the number of tweets per cluster is very similar to the original number of tweet per class we started with, except that for cluster "2".

We can assess our results by computing the evaluation metrics: we obtain a value of 0.06 for the Silhouette score and a value of 0.52 for the Rand Index. The results obtained with sBERT highlight a marked improvement in the Rand Index, a sign that the clusters generated are more consistent with the reference structure than BoW and GloVe.

This improvement is likely attributable to the ability of sBERT to generate contextual representations that capture semantic and syntactic relationships between documents, making the intrinsic structure of the data more apparent to the clustering model. However, the Silhouette score remains low, indicating that clusters are still poorly separated in vector space, but this is not a novelty: it is somewhat to be expected that some hate speech tweet can overlap, just for example by "insulting" two different persons.

We can visualize our results on the Principal Component Axes.

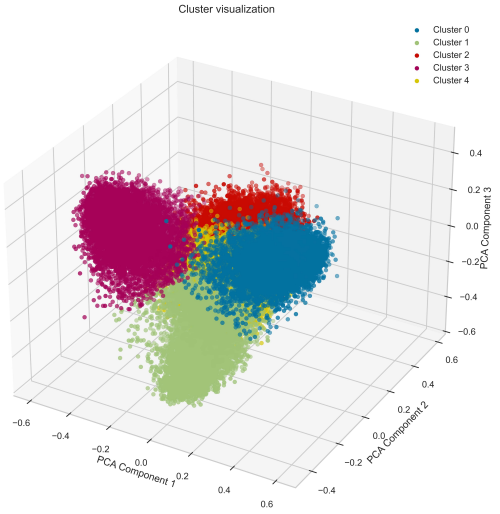


Fig. 12: Cluster visualization using sBERT

This graph [Fig.12] suggests that although sBERT helps to identify proper cluster membership with respect to ground truth, the distances between clusters are not large or well defined, as is expected in the case of natural language.

### 5.3.1 Class Deletion

The overall results show that the choice of document representation significantly affects clustering performance. sBERT manages to improve consistency with respect to ground truth, as evidenced by the Rand Index (0.52), due to its contextual representation.

The number of clusters identified using sBERT is very similar to the actual number of clusters; in fact, the original number of cluster is 6, while the number of cluster identified by $k$-means using sBERT is 5. Moreover, the frequency of samples per class, except that for cluster 2, is close to 8000, which is very similar to the original number of tweets per each class.

If we compute the Imbalanced Ratio IR excluding class 2, we obtain an IR of 1.33 and the mean number of tweets per class is about 7667. Again, this confirms that the distribution in classes, excluding class 2, is not particularly imbalanced and generally speaking close to 8000.

This could be interpreted as the sBERT clustering "finding" all our original classes but the two most general ones: the classes *other_cyberbullying* and *not_cyberbullying* may not be as well defined as the other classes, because they do not revolve around a specific topic but are much more general.

To see if this hypothesis is correct, we can compute which is the best Silhouette score and the corresponding number $k$ of clusters for our dataset excluding one class at a time with the sBERT representation.

The average number of clusters for each best Silhouette score computed removing a cluster is equal to 6.16: very close to our original number of 6 classes. In addition to this we can plot the best Silhouette score obtained by removing classes one by one in order to compare it to the best Silhouette score computed without removing any class, regardless of noisy or not.

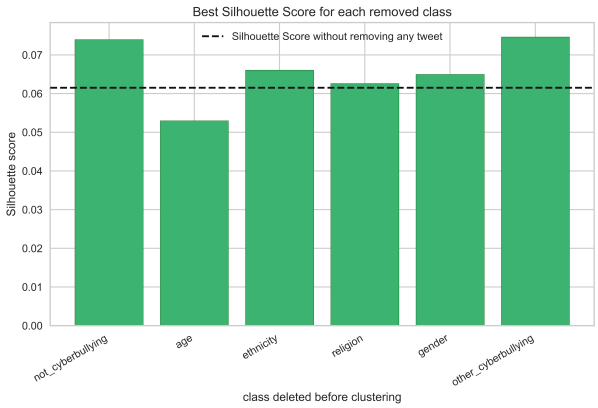


Fig. 13: Improvement of Silhouette score with class removal

This plot [Fig.13] confirms our hypothesis: when removing *not_cyberbullying* and *other_cyberbullying*, we see the Silhouette score improving more than 15% compared to the baseline of no class removal in both cases and essentially staying the same or even worsening when removing the other classes. This is somewhat of a hint that sentence-BERT picked up our original classes and was not able to interpret the most general ones.

### 5.3.2 Topic Modeling

We now want to get useful insights from the sBERT clustering to understand if we can match our original classes in the dataset with ones obtained with the clustering, or if the one obtained in clustering are different than our ground truth.

For this reason, we perform topic modeling using BERTopic, which is the native topic modeling library for BERT models. An example of the output we obtain is represented in the figure [Fig.14] where we have the percentage of associated tweets matched to that topic and a list of the first 10 words

that characterized it, all for the most frequent 15 topics.

number of tweets to which a topic was associated successfully: 4180 out of 6577
Topics for cluster 1

| | Percentage of tweets associated with topic | Representation |
|---|---|---|
| 1 | 8.588517 | [disgusting, black, people, word, is, its, and, niggers, to, are] |
| 2 | 7.535885 | [white, dumb, fuck, you, im, black, niggers, racist, say, nigger] |
| 3 | 7.177033 | [niggers, dumb, fuck, as, are, fucks, some, these, hate, they] |
| 4 | 4.808612 | [colored, racism, ppl, people, color, in, to, but, is, not] |
| 5 | 3.947368 | [negro, called, white, black, people, african, it, be, we, by] |
| 6 | 3.349282 | [her, she, shes, dumb, bitch, girl, said, fuck, nigger, that] |
| 7 | 2.822967 | [obama, ass, president, nigger, dumb, fuck, smh, that, for, this] |
| 8 | 2.607656 | [twitter, tweet, tweets, tweeting, shit, niggers, fuck, dumb, like, you] |
| 9 | 2.511962 | [mad, obama, ass, lol, nigger, fuck, dumb, lmao, bitch, awwww] |
| 10 | 2.344498 | [shut, up, nigger, fuck, you, dumb, the, talk, shit, your] |
| 11 | 2.081340 | [past, anything, black, history, we, people, to, do, the, about] |
| 12 | 1.961722 | [she, obama, mad, her, ass, rt, nigger, fuck, dumb, bitch] |
| 13 | 1.794258 | [incident, pick, racism, to, of, choose, the, on, colored, was] |
| 14 | 1.722488 | [blm, past, anything, lives, black, about, the, in, to, children] |
| 15 | 1.555024 | [he, past, anything, his, him, done, hes, kanye, has, racist] |

Fig. 14: Output of topic modeling for cluster 1

The interesting thing we have is that the representation does reflect the original classes that the dataset has: while single topics are unclear, main concepts across topics of the same clusters reflect the original distinction

For example, considering cluster 1 as in figure [Fig.14], due to the prevalence of racial slurs across different topics' representations, it is clear that cluster 1 is effectively associated with the *ethnicity* class from the original classification. Moreover, essentially no racial slurs are to be found in other clusters.

Lastly, if we take a look at cluster 2 [Fig.15], we see no such structure of specific hate speech emerging from different topics' representations. This confirms our interpretation that the sentence-BERT clustering actually merged the two classes *non cyberbullying* and *other cyberbullying*, probably because they were too generic.

number of tweets to which a topic was associated successfully: 8785 out of 14697
Topics for cluster 2

| | Percentage of tweets associated with topic | Representation |
|---|---|---|
| 1 | 6.226522 | [mkr, teams, show, round, end, contestants, rigged, judges, competition, scoring] |
| 2 | 4.359704 | [twitter, tweet, tweets, retweet, hashtag, tweeting, delete, clients, tweetdeck, client] |
| 3 | 3.130336 | [kat, kats, mkr, face, she, awful, smile, cunt, her, smirk] |
| 4 | 2.540801 | [feminazi, feminazis, called, term, use, joyless, yeller, word, response, actual] |
| 5 | 2.435970 | [andre, kat, mkr, and, hope, strategic, are, katandandre, scoring, they] |
| 6 | 2.253842 | [cook, cooking, cooked, bacon, food, breakfast, meatballs, kitchen, mkr, dishes] |
| 7 | 2.014798 | [christian, woman, right, she, good, god, jesus, her, chickfila, radical] |
| 8 | 1.741605 | [black, white, negro, called, past, anything, racism, colored, negros, americans] |
| 9 | 1.684690 | [trump, barr, he, vote, bully, republicans, yard, school, his, him] |
| 10 | 1.513944 | [play, game, games, blizzard, shadow, wow, pvp, sims, heroes, garrison] |
| 11 | 1.422880 | [song, album, music, songs, musical, band, type, black, albums, artists] |
| 12 | 1.274000 | [college, school, break, tomorrow, th, go, accepted, back, attend, haha] |
| 13 | 1.172453 | [instant, restaurants, restaurant, round, another, kitchen, rounds, mkr, secret, hq] |
| 14 | 1.104155 | [hate, mistakes, people, care, fucking, everyone, off, matter, anymore, pissed] |
| 15 | 1.001707 | [katie, nikki, mkr, nikkis, kim, marriages, mickey, kardashian, stretch, model] |

Fig. 15: Output of topic modeling for cluster 2

To get a complete picture of what we have obtained, we can analyze the word cloud for each specific cluster, highlighting the words that characterize it.


Fig. 16: Word cloud for each cluster

This plot [Fig.16] can be treated as further confirmation of what we said about sBERT's ability to capture the original structure of our dataset.

To be more precise, based on the words that characterized each cluster, we can associate cluster 0 to *age* class, cluster 1 to *ethnicity* class, cluster 3 to *religion* class and cluster 4 to *gender* class of our dataset. The word cloud related to cluster 2 is more difficult to interpret since there are no specific words characterizing it and as such is the combination of the two other original classes.

# 6 Conclusions

This study on classification and clustering on our tweets dataset highlights one important double-sided aspect of hate speech on social media.

Our classification first of all highlights how even a simpler statistical model is able to filter "most" hate speech, by effectively associating different slurs to different hate speech types. This is also what more complex models that are supposed to encapsulate context actually fall back into: a sort of "lazy"

solution, that, when applied in the real world, has many downsides, as we already see today with social media. In fact, in today's world, this result in outright censorship of any and all social media posts that contain such a word; and this leads to the birth of new words, even cumbersome ones, that replace the censored ones, in an endless game of cat and mouse. A rising example of this in 2025 is the censorship of the world "die" or "dead" on platforms such as TikTok, which leads to words such as "unalive" or "unalived" being born, with the exact same meaning.

On the other hand, clustering and in particular sBERT's topic modeling shows contextual embeddings do encapsulate enough meaning to be able to rediscover classes which appear to align with the type of hate speech we already know; in other words, contextual embeddings truly encapsulate colloquial meaning and emergent knowledge behavior, which would never be possible with statistical models.

This last aspect uncovers a path to true moderation on social media: in fact, as much as GloVe was insufficient in encapsulating emergent knowledge and sBERT is, we can extrapolate that some more capable model, with access to more data/trained on dataset more fit to represent the real world, could very well be able to interpret context and meaning in such a way to moderate fairly for everyone, not just with an outright censorship.

As a last remark, better models and more sophisticated techniques already do exist; so it is likely it's just a matter of costs and scalability until such a path referenced above is going to be the main industry standard.

# References


1. Wang, J., Fu, K. & Lu, C.-T. "Sosnet: A graph convolutional network approach to fine-grained cyberbullying detection", 1699–1708 (2020).
2. Kaggle. "Cyberbullying Classification". `https://www.kaggle.com/datasets/andrewmvd/cyberbullying-classification`.
3. Lui, M. & Baldwin, T. langid.py: An Off-the-shelf Language Identification Tool. `https://github.com/saffsd/langid.py` (2012).
4. dwyl. english-words. `https://github.com/dwyl/english-words`.
5. Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **12,** 2825–2830 (2011).
6. NLP, S. GloVe. `https://github.com/stanfordnlp/GloVe?tab=readme-ov-file`.
7. Lab, U. K. P. sentence-transformers. `https://github.com/UKPLab/sentence-transformers/blob/master/README.md`.