

university of groningen / faculty of arts

F*ck Whom?

Target Identification for Offensive Dutch Tweets

Robin van der Noord

Bachelor thesis
Information Science
Robin van der Noord
s3745686
June 9, 2021

ABSTRACT

Because social media platforms have to deal with moderating lots of posts every day, systems are needed to (partly) automate this process. They need to decide whether a post should be removed because it could be considered for example libel or hate speech. In order to help these platforms with such decisions, we sought to answer the question 'What type of classification system can best be used to identify the type of target in offensive Dutch Tweets?'. To answer this question, we compared three different types of models: Support-Vector Machine (SVM), Bidirectional Long Short-Term Memory (BiLSTM) and two types of BERT models: BERTje and RobBERT. We annotated a corpus of Dutch Tweets with explicitness and target. In order to overcome a big class imbalance, we filtered out non-offensive Tweets until there was a ratio of 1:2 of offensive to non-offensive Tweets. The data was pre-split in a training, test and development (dev) set of approximately 70%, 20% and 10% respectively. Due to duplicates being removed, these percentages slightly changed for the final dataset. The models were trained on the training part of this dataset. While developing the models, they were evaluated on the dev part of this corpus.

The final versions of each model were compared with each other using unseen data in the test-set with the macro-averaged F1 score as measure. Multiple different features were tried out for SVM and it resulted in an F1-score of 0.4658. For BiLSTM we experimented with architecture and hyperparameters, which resulted in a score of 0.5434. For BERTje and RobBERT we experimented with epochs and hyperparameters, leading to scores of 0.6204 and 0.6154 respectively. Thus, a BERT-based model can best be used to identify the type of target in offensive Dutch Tweets.

All code for this thesis can be found on https://github.com/robinvandernoord/thesis.

CONTENTS

Abstract i
Preface iii
1 INTRODUCTION 1
2 BACKGROUND 2
3 DATA 5
3.1 Collection and Annotation 5
3.2 Processing 7
3.3 Insights 9
4 METHODS 11
4.1 SVM 12
4.2 BiLSTM 13
4.3 BERT 13
5 RESULTS AND DISCUSSION 15
6 CONCLUSION 18
A EXTENDED ANALYSIS 22
A.1 Data 22
A.2 SVM Feature Examples 24
A.3 Results 24

PREFACE

Please be aware this thesis contains offensive language as examples, in both Dutch and English.

After months of annotating, experimenting and writing, this thesis is now finished. I wrote this as a bachelor thesis for the BSc Information Science at the University of Groningen (Rijksuniversiteit Groningen). I would like to thank my supervisor Tommaso Caselli for his constant support, as well as for providing us with a big corpus with predefined train/test/dev splits to work with. Furthermore, I want to thank the fellow students from my thesis group, Victor Zwart, Ward Ruitenbeek and Zhenja Gnezdilov, for making the annotation process much more enjoyable. Staying motivated during a pandemic is not easy, but being in this together surely helped!

Robin van der Noord
Groningen, June 2021

Ceterum censeo Carthaginem esse delendam.

1 INTRODUCTION

In this decade, more than ever before, we talk a lot on social media. Some of this happens in private or group chats, but a big part of what people say happens publicly on platforms such as Reddit or Twitter. These sites are confronted with the heavy task of moderating the user uploaded content in multiple ways: they have to check copyright claims, adhere to laws of the countries they operate in and remove content that is deemed illegal or a violation of their terms of service.
It is estimated that about 10,000 racial slurs are used on Twitter every day (Bartlett et al., 2014). Moderating that much content can not be done by hand, so systems are needed that can detect and automatically flag offensive text. After detecting an offensive Tweet, it is also useful to find out the target of the offense. Is the Tweet a harsh criticism of a politician or can it be considered libel ('*smaad*' in Dutch)? Is something hate speech ('*groepsbelediging*') or can it be considered freedom of speech ('*vrijheid van meningsuiting*')?
These issues are of course complex and require the attention of legally qualified people. However, classifying whether the target of an offensive Tweet is a person, group or something else can aid platforms in moderating their content quickly.

Most research published about this topic has been for languages other than Dutch (e.g. for English by Zampieri et al. (2019a) or for Danish by Sigurbergsson and Derczynski (2019)). We thought it would be interesting to apply the knowledge of previous work to a new language, and compare whether the same classification models achieve high scores or whether a different model works better for Dutch. Thus, our research question for this thesis was:
What type of classification system can best be used to identify the type of target in offensive Dutch Tweets?

The thesis is structured as follows: Chapter 2 contains an overview of related work in this field. In chapter 3 we explain the data structure, annotation and processing that was required to answer our research question. In chapter 4 we describe the different methods used and in chapter 5 we describe the results gained from the different models. Finally, in chapter 6 we reflect on the research. Appendix A contains graphs and tables that are informative, but were left out of the main text in order to avoid cluttering.

2 BACKGROUND

Waseem et al. (2017) observed that previous works in the field of offensive/abusive language all use different labels for similar concepts. For example, some work would call sexist remarks 'insults' (e.g. Van Hee et al. (2015)), whereas some others would call these 'hate speech' (e.g. Nobata et al. (2016)). This ambiguity causes confusion during annotation, so they propose a new standardized typology with two main categorizing factors:
1. Is the abuse directed or generalized?
Directed means the abuse is targeted towards a person, a small group of people, an organisation or some other entity. One example of directed abuse (in terms used by previous works) is cyberbullying. Alternatively, the abuse can be generalized: directed towards a group of people with a common ethnicity, gender, sexual orientation or some other categorization. To model that the abuse is directed, the authors advise using modelling features that identify the target, like the uses of proper nouns, pronouns or specific syntax. A complication they foresee is with modelling generalized abuse, since different indicative words can be used for each distinct group. For instance, different words are used to target different ethnicities.
2. Is it explicit or implicit?
Explicit language contains unambiguous key words that can be considered offensive; It is obvious that it is meant to offend someone. Implicit language on the other hand does not contain such key words. The offense is hidden behind words that are innocent by themselves. For example, this can be achieved by using sarcasm (for an insult) or so-called 'dog whistles' (for hate speech): normal words that have a hidden and hostile meaning most people are not aware of (Albertson, 2015). The absence of indicative words make this type of offense harder to detect, both for human annotators and machine learning models. The paper suggests that the best method to improve automatic identification of implicit abuse is having a very high quality of annotation, possibly with a (partial) lexicon.

Whereas previous studies had usually focused on very specific subtasks (e.g. hate speech or cyberbullying), Zampieri et al. (2019a) aim to target several different kinds of offensive content. They introduce a three-step annotation schema:

- Task A. Is the Tweet offensive? This includes both explicit and implicit offenses.
- Task B. (if the Tweet is offensive:) Is the offense targeted or untargeted? An untargeted offense is when a Tweet contains a (general) profanity, not directed towards someone or something in particular. A targeted offense is either an insult or a threat.
- Task C. (if the Tweet is targeted:) Who is the offense targeted towards?
  - Individual: can be either targeted by name or unnamed (cyberbullying)
  - Group: considered based on ethnicity, gender or some other common characteristic (hate speech)
  - Other: if the offense is clearly targeted but not at one of the two above (e.g. organisation or concept)

While the authors do mention the typology of Waseem et al., they decided to create this new schema, without the differentiation between explicit and implicit, but with more specific targets. We will focus on this third task, since it is most similar to

what we aim to do in this research.

After creating a dataset with these guidelines, Zampieri et al. experimented with three different types of models:
- Support vector machine (SVM) with word unigrams,
- Bidirectional long short-term memory (BiLSTM) with FastText word embeddings, adapted from an existing model for sentiment analysis,
- Convolutional neural network (CNN).
For the third task (Task C, Target Identification), these models all scored roughly the same, with the neural network-based models (0.47 F1 Macro) slightly better than SVM (0.45).

The dataset they created for this paper (called OLID) was also used in Task 6 of the 2019 Workshop on Semantic Evaluation (SemEval). In this shared task (also called OffensEval, Zampieri et al. (2019b)), dozens of teams created models as well to tackle the same three subtasks on the same dataset. For subtask C there were 66 participants. A lot of these teams managed to achieve higher scores than the CNN and BiLSTM models from the original paper.
The number one on Task C (Nikolov and Radivchev, 2019) used a BERT-Base (uncased) model to score an F1 score of 0.6597. This shows the potential of BERT on our domain. While they also experimented with many other models (SVM, CNN, Logistic Regression i.a.), these struggled with either overfitting or underachieving.

Whereas OLID is an English-only dataset and OffensEval only focused on English language offenses, Sigurbergsson and Derczynski (2019) set out to create models that would work with both English as well as Danish.
They worked on the same subtasks with their own dataset, which contains Danish texts from Twitter, Facebook and Reddit (r/DANMAG and r/Denmark). They performed classification using Logistic Regression and three variations of BiLSTM on the OLID and their own dataset.
On Task C, they outperformed the models from the original OLID paper with every one of their models. Their highest scoring model was (learned) BiLSTM (with 10 epochs) with a macro F1 score of 0.557.
On the Danish dataset, the same model with 100 epochs scored 0.629 as the best. The models with FastText word embeddings and additional features scored worse for this task.

A year before SemEval 2019, there was a similar shared task at GermEval 2018 (Wiegand et al., 2018). This (pilot) task had two subtasks:
1. Coarse-grained (binary) classification, where the Tweet is classified offensive or not,
2. Fine-grained classification, where the Tweet gets one of four labels:
- OTHER: the Tweet is not offensive,
- PROFANITY: profane words are used, but the Tweet is not meant to hurt or insult someone,
- INSULT: the Tweet is meant to offend someone,
- ABUSE: this is a stronger form of insult, and the Tweet is meant to degrade someone. Someone can be degraded for being the member of a (minority) group, or other dehumanizing language is used.
This shared task does not contain a subtask for the target of the offenses, and Zampieri et al. (2019a) mention there was no prior work that explored this, while it might be important in many scenarios.

Table 1: SemEval 2020 Task 12 Results. Source: Zampieri et al. (2020)

Team	Paper	Macro F1	BERT type
Galileo	Wang et al. (2020)	0.7145	ALBERT
LT@Helsinki	Pàmies et al. (2020)	0.6700	BERT
PRHLT-UPV	De la Pena Sarracén and Rosso (2020)	0.6692	BERT

At SemEval 2020, there was another edition of OffensEval (Task 12, Zampieri et al. (2020)). Rosenthal et al. (2020) used semi-supervised learning to create a bigger version of the OLID dataset (called SOLID for *Semi-Supervised Offensive Language Identification Datatset*) that was used for this task. For all three subtasks their BERT model achieved the best results on this dataset. Furthermore, the top three teams with the highest macro-averaged F1 scores for subtask C all used a BERT-based model (See Table 1 for results).

For Dutch, there has not yet been a lot of research in this field. Tulkens et al. (2016) created and used a word dictionary to detect racism in Dutch social media. Their best model (SVM) scored an F1-score of 0.46. While this research has a much more specific goal (racist texts rather than offensive ones), it might be interesting to use (part of) the dictionary to improve our own models.

This research builds upon previous works by combining the task of target detection with a new language domain (Dutch) and state-of-the-art machine learning models. Based on this previous work, our hypothesis is that a BERT-based model will reach the best accuracy and F1-scores, followed by the BiLSTM and the SVM model.

3 DATA

3.1 COLLECTION AND ANNOTATION

Since our thesis supervisor provided us with a big corpus of Dutch Tweets, we did not have to do data collection ourselves. Additionally, we were provided with annotation guidelines that fit our goals well.
How this data was collected can be found on the DALC GitHub page. The dataset we received consists of Tweets from multiple different sources. It contains the following fields: id, text (Tweet contents), username, user description and source. This last field is used to indicate from which corpus the Tweet was collected.
Part of this dataset had already been annotated for abusiveness. Since abusive Tweets are always offensive, we did not have to annotate these Tweets ourselves. Another part of the corpus was previously annotated 'not abusive'. Since these Tweets could still contain offensive content, we had to re-annotate these with our offensiveness guidelines. The final part of the corpus consisted of not yet annotated data. Due to time constraints, we were only able to annotate approximately half of these Tweets.
The Tweets have been annotated by everyone in our research group (four bachelor students). After annotating each Tweet individually, the annotations were merged. For the cases where there was no majority agreement about a label, we manually discussed and agreed on the final (gold) label. As can be seen in Table 2, there was an overall moderate agreement for explicitness (Fleiss' Kappa of 0.41). The pair-wise agreements ranged from slight to moderate agreement. For target, there was overall fair agreement (0.36, Table 3) and pair-wise agreements ranging from fair agreement to moderate agreement. The fact that these kappa scores are not very high can be explained due to the fact that these annotations are very subjective, and every student had their own views on what could be considered offensive. Even after combining our annotations and discussing the difficult cases, there still was a big disagreement for some words and cases.

We used a two-step annotation schema, that can be described as a combination of the typology by Waseem et al. (2017) and the OLID schema by Zampieri et al.

Table 2: Pair-wise Cohen's Kappa scores for annotator pairs and the group Fleiss' Kappa score (Explicitness)

	annotator 1	annotator 2	annotator 3	annotator 4	Fleiss' Kappa
annotator 1	-	0.4142	0.5387	0.3983	
annotator 2	0.4142	-	0.3473	0.3111	
annotator 3	0.5387	0.3473	-	0.455	
annotator 4	0.3983	0.3111	0.455	-	
					0.4123

Table 3: Pair-wise Cohen's Kappa scores for annotator pairs and the group Fleiss' Kappa score (Target)

	annotator 1	annotator 2	annotator 3	annotator 4	Fleiss' Kappa
annotator 1	-	0.3494	0.4688	0.3632	
annotator 2	0.3494	-	0.2867	0.2266	
annotator 3	0.4688	0.2867	-	0.4427	
annotator 4	0.3632	0.2266	0.4427	-	
					0.3631

(2019a). The first step is used to indicate whether the Tweet is potentially offensive, as with the OLID schema. However, we also annotated whether the offense was explicit or implicit. If the text contained a word that clearly indicated the Tweet is meant to be offensive, such as a profanity, slur or other offensive terms, we annotated it with EXPLICIT. Examples:
1) *Liberalen zijn idioten [Liberals are idiots]*
2) *Godverdomme [Goddamnit]*

If the Tweet has the intention to offend, but this is not indicated by one or a few key words, we marked it IMPLICIT. Example:
3) *Minder minder Marokkanen [Less less Moroccans]*

If the Tweet was considered not offensive, it received the annotation NOT for explicitness. Note: a Tweet where the author offends themself is also considered offensive. Example:
4) *Ik ben een idioot [I am an idiot]*

Whereas Zampieri et al. (2019a) have a separate step for annotating whether the offense is targeted or untargeted and who the target is, we annotated this in one label: TARGET (INDIVIDUAL | GROUP | OTHER | NOT). If a potentially offensive Tweet was targeted at one person or a few specific people (for example three politicians), we marked it INDIVIDUAL. Example:
5) *Wilders is een idioot [Wilders is an idiot]*

A target was marked as GROUP if the offense was directed towards a (social) group: a group of people with a common ethnicity, gender, sexual orientation or some other categorization. When the target could not be exactly determined, but was clearly targeted towards an unknown 'they', we marked it with GROUP as well. Examples:
6) *Zij zijn idioten [They are idiots]*
7) *Conservatieven zijn corrupt [Conservatives are corrupt]*

When the offense was targeted towards something else than an individual or group (e.g. an organisation, institution or concept), the Tweet received the annotation label OTHER. Examples:
8) *Wat een kutweer [The weather sucks]*
9) *Het katholicisme is onzin [Catholicism is nonsense]*

Notice that when the target is a faith (such as Catholicism or Islam), the target label is OTHER and when the target is followers of this faith (Catholics or Muslims), the target label is GROUP. The same goes for political parties (See example 6) versus members of these parties (Example 7) and political ideas (*socialism*) versus supporters of these ideas (*socialists*).
6) *De VVD is corrupt [The VVD is corrupt]*
7) *VVD-stemmers zijn dom [VVD voters are stupid]*

If a Tweet was not marked as offensive in step 1, the target was not filled in.

After noticing that Excel was not optimal for annotating, we decided to create a small Python annotation tool[1] in order to somewhat speed up the annotation process. The tool relieved some of the manual labor by keeping track of duplicates and Tweets that were extremely similar, as there were a lot of Tweets where the textual content was exactly the same, but a different URL or mention was included. The tool shows your progress, the Tweet you are currently annotating and the possible

[1] GitHub: Thesis Annotation Tool

Table 4: Amount of Tweets in Pre-Final and Final dataset

	Pre-Final				Final			
	Train	**Test**	**Dev**	**Total**	**Train**	**Test**	**Dev**	**Total**
Offensive	2588	1036	361	3985 (28%)	2588	1036	361	3985 (35%)
Not Offensive	7666	2327	186	10179 (72%)	5176	2072	186	7434 (65%)
Total	10254 (72%)	3363 (24%)	547 (4%)	*14164*	7764 (68%)	3108 (27%)	547 (5%)	*11419*

labels, which you can select by typing the first letter. This made the annotation slightly easier and faster.

3.2 PROCESSING

We combined the aggregated datasets together with the abusive dataset, filling the 'abusive' column when known or with UNKNOWN otherwise. Before annotating, the data was pre-split in a training, development (dev) and test set of approximately 70%, 10% and 20% respectively. Due to removing duplicates, these percentages slightly changed for the final dataset. This resulted in a pre-final dataset with all annotated Tweets (14164 in total): a train set of 10254, a development set of 547 Tweets and a test set of 3363 Tweets. As can be seen in Table 4, this pre-final dataset consisted of 72% non-offensive Tweets. In order to overcome this class imbalance, we removed non-offensive Tweets from the train and test set until these were approximately 66% of the sets. We did not do this for the development set, since this was already a small set, and had almost the right ratio of 1:2 offensive to non-offensive. For train, we removed 2490 Tweets and for test, we removed 255. Removing these Tweets resulted in our final dataset, with a split of 68%, 5% and 27% for train, development and test respectively. These sets were carefully decided before annotation, making sure that there were no authors that exist in both sets, to prevent data leakage (Kaufman et al., 2012).

To use the dataset with our models, we extracted the text and target fields for Tweets with an EXPLICITNESS label of IMPLICIT or EXPLICIT. The target was converted to a numeric label (0 to 4), because some of the models required this. We trained the models on the training data and validated results during development on the dev set. These results were compared using the accuracy (which was the same as micro average F1 score - F1 is the harmonic mean of the precision and recall), since this is the metric most models calculate by default. Furthermore, we used scikit-learn's Classification Report[2] to check the metrics for each individual class, as well as the macro F1 score, which has a higher penalty when some classes are doing poorly in comparison to the rest. Lastly, we also printed out a confusion matrix[3] in order to see where the models go wrong. The best models were compared with each other using the unseen data in the test set. Due to the randomness of neural networks, we trained and evaluated the BiLSTM and BERT models five times and reported the average of these scores. The final models were compared using the Macro F1 score. We tried out the following pre-processing steps (steps marked with * were kept, the others were discarded):

- Changes on the raw Tweet text:
  - * Lowercasing

[2] sklearn.metrics.classification_report
[3] sklearn.metrics.confusion_matrix

  - Removing all non-ASCII characters
  - Removing punctuation (.,!?-|:;())
  - Removing mentions by replacing '@someone' with '[USER]'
  - * Removing mentions by removing the @ symbol, so that '@someone' becomes 'someone'
  - * Removing urls by replacing them with '[URL]'
  - * Removing the # from hashtags
  - Replacing emoji characters with their (English) name, using 'unicodedata'
  - * Replacing numbers with [NUMBER]
- Tokenization:
  - by simply splitting on spaces,
  - * using the nltk tokenizer (word_tokenize),
  - using the nltk TweetTokenizer.
- Changes on the tokens:
  - Stemming using the nltk Snowball Stemmer for Dutch

3.3 INSIGHTS

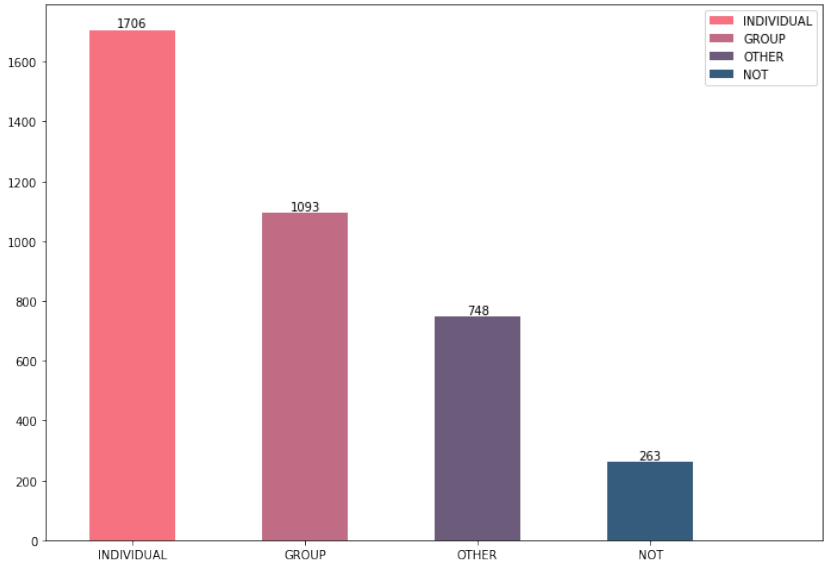


Figure 1: Class Distribution of Target in the final dataset

In Figures 1, 5 (Appendix A) and 6 (Appendix A), the distribution of different classes in the full dataset can be seen for the Target, Explicitness and Abusiveness fields respectively. In Table 5 we show the distribution of Tweets over the different sets.
In Figure 7 (Appendix A) we compare the abusiveness label with explicitness. Since abusive Tweets are always offensive, it makes sense that there is a lot of overlap between the two. On the other hand, non-abusive Tweets can still be offensive and this can also be seen in the heatmap.
In Figure 2 we show the distribution of targets with respect to the explicitness. As one can see, there are generally no considerable differences in the target depending on the explicitness. The exception to this is NOT, where explicit is much more common than implicit. Untargeted profanities (such as Example 2) were assigned the labels EXPLICIT and NOT, while IMPLICIT and NOT was only given when the Tweet was potentially implicitly offensive, but we were not sure to whom. This latter case was not very common (only 16 occurrences), while swearing without a specific target happened much more often (247 cases).

Table 5: Distribution of Tweets

(a) Offensiveness

	Offensive	Not offensive	Total
Train	2588	5176	7764
Dev.	186	361	547
Test	1036	2072	3108
Total	3810	7609	11419

(b) Target[4]

	IND.	GRO.	OTH.	NOT	Total
Train	1144	723	535	186	2588
Dev.	91	49	37	9	186
Test	471	321	176	68	1036
Total	1706	1093	748	263	3810

[4] IND: Individual, GRO: Group, OTH: Other, NOT: No Target

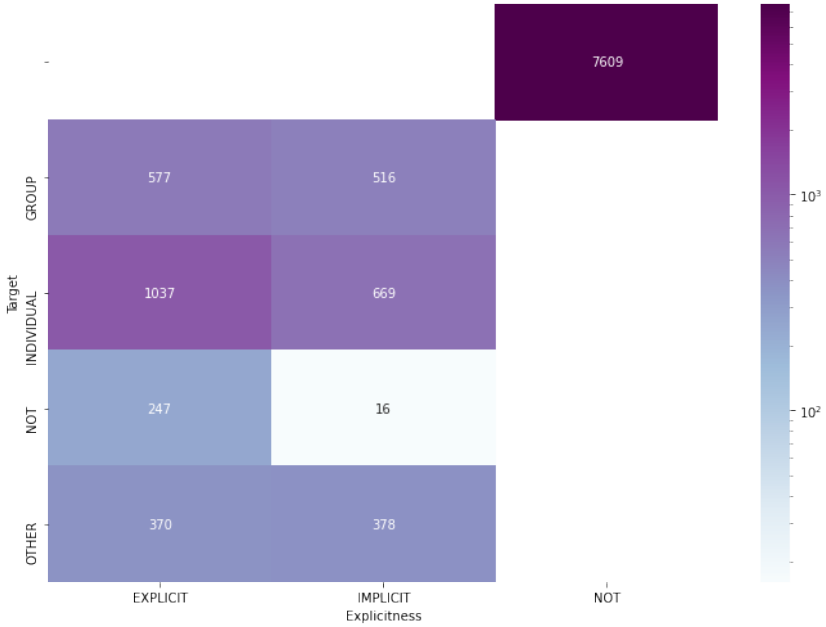


Figure 2: Explicitness vs Target

4 METHODS

To answer our research question, we have compared some of the models used by Sigurbergsson and Derczynski (2019) and by the participants of SemEval 2019 Task 6 (Zampieri et al., 2019b) and 2020 Task 12 (Zampieri et al., 2020). The SVM, BiLSTM and CNN of Zampieri et al. (2019a) all achieved roughly the same F1 Macro score (0.45-0.47) for the target detection subtask. Sigurbergsson and Derczynski (2019) also reached the highest Macro F1 scores using a (Learned) BiLSTM model, both for English and Danish. Because BiLSTM scored highest in both of these papers, and due to not having unlimited time, we have decided to use this classifier and to leave out CNN. SVM was included as well, since this classifier is not as complicated to setup to train with. Additionally, a baseline 'dummy' classifier was used to compare the other models with. This dummy always predicts the most frequent class from the training data: Individual.

In addition to these previously used models, we thought it would be interesting to include the state-of-the-art system BERT (Bidirectional Encoder Representations from Transformers, Devlin et al. (2018)), which achieved high scores on SemEval 2020 Task 12. While BERT has support for Dutch through its multilingual model, we chose the Dutch-specific BERTje (de Vries et al., 2019) and RobBERT (Delobelle et al., 2020). Since BERT is said to mark the beginning of a new era in natural language[1], we were curious to see its result in relation to 'older' models.
To summarize, we have used the following classification models:

- Linear SVM (Support-Vector Machine: Hearst et al. (1998); using scikit learn: Pedregosa et al. (2011))
- BiLSTM (Bidirectional Long Short-Term Memory: Hochreiter and Schmidhuber (1997), Graves and Schmidhuber (2005))
- BERTje (de Vries et al., 2019)
- RobBERT (Delobelle et al., 2020)

The experiments were performed on Google Colaboratory with a GPU runtime (NVIDIA Tesla T4, P100 or V100 GPU), since this is much faster for neural network-based models (BiLSTM, BERT) than using a regular CPU runtime. All code for this thesis can be found on https://github.com/robinvandernoord/thesis.

[1] @lmthang on Twitter

4.1 SVM

In addition to the pre-processing steps as mentioned in Section 3.2, SVM required us to do feature extraction on the Tweets. The following features were tried:

- Bag of Words (BOW): a dictionary containing the words in the Tweet as keys, and True as the values,
- Counted Bag of Words: the same keys as BOW, but with how often the word exists in the Tweet as value,
- Bag of Words and Bigrams: in addition to BOW, this feature set also had a key for each bigram in the Tweet,
- Bag of Hateful Words: Using the Dutch hateful dictionary by Tulkens et al. (2016) called HADES, we created a bag of words feature set that contained just the matched hateful words in the Tweet as keys, and True as the values. This dictionary contains mostly racist terms. The keys were prefixed with 'hateful__' in order to be able to use them in combination with other features, without conflicting with those keys.
- Counted Bag of Hateful Words: the same keys as the Bag of Hateful Words, but with how often the word exists in the Tweet as value.

An example for each of these features can be found in Section A.2 (Appendix A). Furthermore, using the high_information_words function by Nitin et al. (2016), we implemented a 'high-info' feature selection on top of the aforementioned features. This filtered out any features that would otherwise not add anything to the system.

We decided to use the scikit-learn implementation Linear Support Vector Classification (LinearSVC, Pedregosa et al. (2011)), since this is their fastest implementation and yields adequate results. As can be seen in Figure 3, scikit-learn offers multiple so-called kernels that calculate the boundaries between the different classes. LinearSVC however only supports the linear kernel. We also experimented with the hyperparameters C (regularization parameter), class weight and maximum iterations. However, this had either a slight negative effect (in the case of C) or no effect at all (class_weight and max_iter).

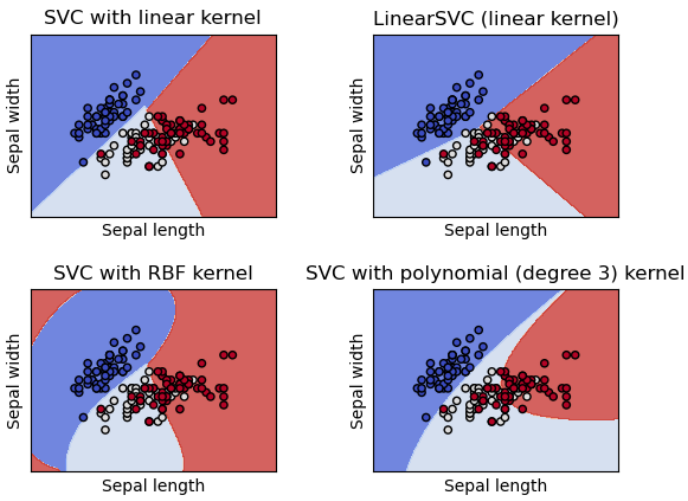


Figure 3: SVM kernels, Source: Pedregosa et al. (2011)

4.2 BILSTM

In order to help the BiLSTM model understand the meaning of human (Dutch) language, we had to initialize it with word embeddings[2]. We chose the word2vec embeddings from Coosto[3]. While FastText also has a Dutch word embeddings model, it is trained on Wikipedia texts, whereas the Coosto model is trained on multiple online sources, including Social Media messages. This better fitted our own corpus.

We used the tensorflow-based library 'keras' to build our BiLSTM model[4], because it is, as they claim, "Simple. Flexible. Powerful." [5].
The architecture of the final model is as follows (see also Figure 4):

- Embedding - the first layer is used to convert the tokens into the Coosto embeddings.
- Bidirectional LSTM - the actual BiLSTM layer.
- Dropout (with a rate of 0.25),
- Dense,
- Dropout (with a rate of 0.2) - these three layers are used to help the model learn better and prevent overfitting.
- Dense with Softmax activation - the final layers outputs the actual classification.

While we experimented with different architectures (without the middle Dense layer or with an additional BiLSTM layer), these were either faster while yielding worse results, or slower with similar results. Additionaly we experimented with the rates of the Dropout layers, the amount of units for each layer and the amount of epochs. The advantage of Bidirectional LSTM over a regular LSTM model is that it can see more context (the words that follow and precede a word in a sentence), which makes it better at prediction. An advantage of (Bi)LSTM over SVM is that word embeddings carry much more information than the features that SVM can use, which can help the system learn better. Features on the other hand are more flexible, whereas they are not limited by words that might be missing from the word embeddings model ('out-of-vocabulary' words), which could still contain relevant information. Some examples of this are proper nouns, new words, slang or intentionally misspelled words: *Halalsema* was a common implicit insult towards Femke Halsema (INDIVIDUAL) in our dataset.

4.3 BERT

We made our concluding model using the Simple Transformers library (Rajapakse, 2019)[6], which is based on HuggingFace's Transformers library (Wolf et al., 2019)[7]. Simpletransformers has built-in support for (multi-class) text classification and made it very easy to start working with a BERT-based system. It also took care of all pre-processing for this model, including tokenization. We experimented with the amount of epochs, learning rate and batch size. The BERTje model is listed on HuggingFace as 'GroNLP/bert-base-dutch-cased'.
In addition to BERTje, we also experimented with an additional Dutch language

[2] Manning (2017) explains this well
[3] GitHub: Coosto Dutch Word2Vec Model
[4] Keras + word2vec tutorial
[5] Keras: BiLSTM
[6] Simple Transformers
[7] Hugging Face

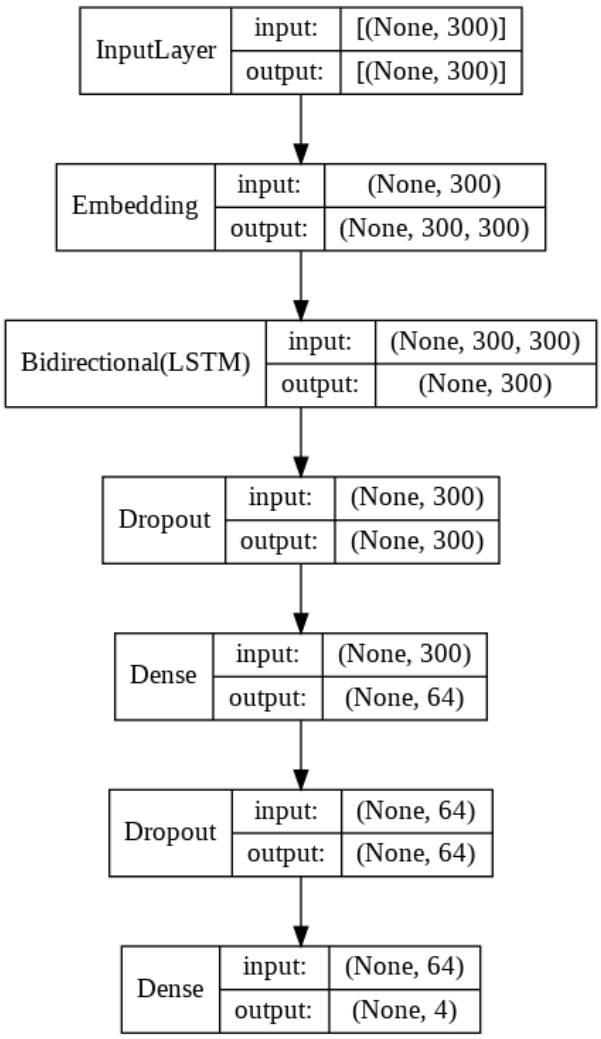


Figure 4: BiLSTM architecture, made using keras.utils.plot_model

model: RobBERT (Delobelle et al. (2020), 'pdelobelle/robbert-v2-dutch-base'). This model is based on the RoBERTa architecture (Robustly Optimized BERT Approach, Liu et al. (2019)). Since they claim it is even more powerful than the original BERT model, we wanted to compare it with BERTje on our domain. RobBERT required us to do a little additional preprocessing: we needed to remove newlines from the Tweet texts, since otherwise it would crash.

BERT is a relatively new, state of the art transformer model (Vaswani et al., 2017) and proved to be successful in generating features for classification (Devlin et al., 2018).
An advantage of a BERT-based model over BiLSTM with word embeddings, is that BERT models have contextual word embeddings: the meaning of a word depends on its context. This is due to to the fact they are (pre-)trained differently, namely by removing (masking) words or parts of words from the text and letting BERT guess which word was removed. This makes the model more aware of the different meanings a word can have, and this helps its understanding during classification.
Another advantage of BERT(je) is that it uses a vocabulary of Word Pieces (Wu et al., 2016) instead of words/tokens. This means it has better support for out-of-vocabulary words.

5 RESULTS AND DISCUSSION

For our final SVM model, we used a combination of the 'Bag of Words and Bigrams' and 'Bag of Hateful Words' features. The high-info feature selection did not have a (positive) effect on the scores, and was not used for the final model. As mentioned in section 4.1, the default hyperparameters gave the best results.
For BiLSTM we decided to use eight epochs for the final model, since this consistently gave the best results on the development set. The architecture is described in section 4.2 and can be seen in figure 4.
For BERTje, 3 epochs with a batch size of 4 and a learning rate of 6e-5 (6×10^{-5}) worked best and for RobBERT the best configuration was found to be 4 epochs with a batch size of 4 and the simpletransformers default learning rate of 4e-5 (4×10^{-5}).

For each of the different models we calculated a 'baseline' score: the accuracy and F1 score of the model with minimal or without configuration. These scores were used to compare the effect of our improvements on the models.

As one can see in Table 6, the Dummy classifier (which only predicted the most common label: INDIVIDUAL) did very poorly in measures of macro-averaged F1 score. This is of course due to it only predicting one class, thus having a precision and recall (and resulting F1 score) of 0 for the other three classes. It is also interesting to see the baseline BiLSTM scores worse than the baseline SVM, but for the improved versions it was the other way around. A possible explanation for this effect can be found in the word embeddings. The input for the improved BiLSTM model was much cleaner than for the baseline, which meant it could match more words with the Coosto vocabulary. Consequently, the improved model perhaps had a better understanding of the meaning of the Tweets, which could have lead to improved scores.
Furthermore, the baseline for BERTje already almost matched the improved BiLSTM F1 score and the RobBERT baseline even exceeded it. As mentioned before, BERT-based models have an even better understanding of the meaning of sentences due to the contextual word embeddings. Whereas BERT has a way of handling 'out-of-vocabulary' words, the BiLSTM model is limited by the word embeddings model. This means some words and emoji that could be important to the meaning of the Tweet, were skipped by the BiLSTM. This could be a possible reason for the difference in scores.

Table 6: Results: dev/test accuracy and test F1-score for each baseline and improved model

model	configuration	dev accuracy	test accuracy	test F1
Dummy	most frequent class	0.4892	0.4546	0.1563
SVM	baseline	0.5645	0.5685	0.4658
SVM	improved	0.6344	0.6197	0.5108
BiLSTM	baseline	0.5376	0.5819	0.4319
BiLSTM	improved	0.6398	0.634	0.5434
BERTje	baseline	0.6523	0.6782	0.5234
BERTje	improved	0.7205	0.6942	0.6204
RobBERT	baseline	0.6666	0.7054	0.6011
RobBERT	improved	0.7114	0.699	0.6154

Table 7: true positives per class for each model[1]

	INDIVIDUAL	GROUP	OTHER	NOT	Total
Dummy	471	0	0	0	471
SVM baseline	256	166	46	21	489
SVM improved	381	188	47	26	642
BiLSTM baseline	2046	744	163	61	3014
BiLSTM improved	1853	920	349	162	3284
BERTje baseline	2071	1081	318	43	3513
BERTje improved	1894	1064	423	215	3596
RobBERT baseline	2004	1165	357	128	3654
RobBERT improved	1926	1109	404	182	3621

In section A.3 of Appendix A one can find the confusion matrices of each of the models mentioned in Table 6. As a summary, Table 7 contains the true positive cases per class for each model[1].
The improved SVM model (Fig. 10, appx. A) scored better for each of the classes than the baseline (Fig. 9, appx. A).
BiLSTM (Fig. 11, appx. A and Fig. 12, appx. A) lost a few of the INDIVIDUAL true positives, but improved on the other three classes. This could mean that this model went from often simply guessing the most frequent class to better understanding the different classes.
BERTje (Fig. 13, appx. A and Fig. 14, appx. A) and RobBERT (Fig. 15, appx. A and Fig. 16, appx. A) both decreased the true positives for INDIVIDUAL and GROUP in the improved models, while increasing them for the OTHER and NOT classes. This effect can also be seen in the models' scores: improved BERTje has only a slightly higher accuracy than the baseline, but the F1 score is much higher.
For RobBERT, the accuracy was even lower for the improved model, whereas the F1 score was slightly higher. It is also interesting to see that the RobBERT baseline already scored such high scores, that it was hard to improve them beyond these initial scores. While BERTje started off with a much lower F1 score, it managed to surpass RobBERT for the final model.

When looking at the full classification report summary in Table 8 and the full Tables 9 to 17 in Appendix A, there appears to be a connection between how often a class occurs in the data (as can be seen in Figure 1) and how well the models could predict it. In general, the models did well with INDIVIDUAL, which was the most common class, and GROUP (second most common). They tended to struggle with OTHER and NOT. Only BERTje and RobBERT reached an F1-score of at least 0.5 and 0.4 for OTHER and NOT respectively. Moreover, even for these models the F1-score for INDIVIDUAL and GROUP was still much higher.

[1] Note: The neural-network based models (BiLSTM, BERTje and RobBERT) were trained and evaluated five times and their true postives were totalled. The dummy classifier and SVM only contain the results of one run.

Table 8: F1 scores summary for each (improved) model

	Dummy	SVM	BiLSTM	BERTje	RobBERT	support
INDIVIDUAL	0.625	0.7362	0.7587	0.7975	0.7921	471
GROUP	0	0.6277	0.6213	0.6887	0.7157	321
OTHER	0	0.3208	0.4203	0.5296	0.5125	176
NOT	0	0.3586	0.3735	0.4658	0.4414	68
Micro-Average	0.4546	0.6197	0.634	0.6942	0.699	1036
Macro-Average	0.284	0.5108	0.543	0.6204	0.6154	1036

Based on previous work, notably SemEval 2020, it was expected that the BERT-based models would score the highest. As can be seen in the results, these models delivered on this expectation.

6 CONCLUSION

Because social media platforms have to deal with moderating lots of content every day, systems are needed to (partly) automate this process. In order to help these platforms, we sought to answer the question '*What type of classification system can best be used to identify the type of target in offensive Dutch Tweets?*'.
In order to answer this question, we trained multiple machine-learning models on a newly annotated dataset. We annotated this dataset with explicitness and target, together with three other bachelor students. We labeled the Tweets with EXPLICIT, IMPLICIT or NOT for explicitness, and labeled offensive Tweets with a target of INDIVIDUAL, GROUP, OTHER or NOT (no target).
The models we trained were a Support Vector Machine (SVM), Bidirectional Long Short-Term Memory (BiLSTM) and two BERT-based models: BERTje and RobBERT.

The final ranking of models, ordered by F1 score (descending) is:

1. BERTje (improved)
2. RobBERT (improved)
3. RobBERT (baseline)
4. BiLSTM (improved)
5. BERTje (baseline)
6. SVM (improved)
7. SVM (baseline)
8. BiLSTM (baseline)
9. Dummy (most frequent class)

So to answer the research question: the highest accuracy and F1-scores can be achieved by using a BERT-based classification system (BERTje or RobBERT). This result is in line with our hypothesis.
RobBERT is a good choice when you do not have as much time to spend on optimizing the model, since it already gave very results for the baseline. If you need optimal scores, BERTje would be better. However, these Transformer models are very computationally expensive and require a good GPU to run efficiently.
If you do not have access to this, or do not want to wait for the training to complete, LinearSVC (SVM) might be better, since this system is much less computationally demanding while also giving decent results after pre-processing and good feature selection. In order to improve the SVM model, the hateful words feature could be improved with a more extensive word list, since the one we used was focused on racist terms. Other offensive terms could be included to improve this model further.

Future research could also include the explicitness label, since this could be a possible indicator of the target. Perhaps implicit offenses are more often targeted towards for example a group. However, this would require either looking at the annotated explicitness, which would make it hard to use in a production environment with unannotated data, or an additional system to classify the explicitness.
Additionally, the user description could be included since this may give additional context to the Tweet. Some Tweets that might be ambiguous by themselves could possibly be classified more easily with this context. However, one would have to be careful to prevent bias: a Tweet should not be classified as offensive towards a

group just because of the users description.
Lastly, a semi-supervised machine learning approach (such as used for SOLID, Rosenthal et al. (2020)) could be taken in order to extend the dataset without all of the manual labor. This would mean more training data, which could hopefully improve all or some of the models.

BIBLIOGRAPHY

Bethany L Albertson. 2015. Dog-whistle politics: Multivocal communication and religious appeals. *Political Behavior*, 37(1):3–26.

Jamie Bartlett, Jeremy Reffin, Noelle Rumball, and Sarah Williamson. 2014. Anti-social media. *Demos*, (2014):1–51.

Pieter Delobelle, Thomas Winters, and Bettina Berendt. 2020. RobBERT: a Dutch RoBERTa-based language model. *arXiv preprint arXiv:2001.06286*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM networks. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 4, pages 2047–2052. IEEE.

Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Shachar Kaufman, Saharon Rosset, Claudia Perlich, and Ori Stitelman. 2012. Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(4):1–21.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Christopher Manning. 2017. Representations for language: From word embeddings to sentence meanings. *Simons Institute for the Theory of Computing, UC Berkeley*.

Alex Nikolov and Victor Radivchev. 2019. Nikolov-radivchev at SemEval-2019 task 6: Offensive tweet classification with BERT and ensembles. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 691–695, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Hardeniya Nitin, Perkins Jacob, Chopra Deepti, Joshi Nisheeth, and Mathur Iti. 2016. *Natural Language Processing: Python and NLTK*, pages 380–384. Packt Publishing.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th international conference on world wide web*, pages 145–153.

Marc Pàmies, Emily Öhman, Kaisla Kajava, and Jörg Tiedemann. 2020. LT@ Helsinki at SemEval-2020 task 12: Multilingual or language-specific BERT? *arXiv preprint arXiv:2008.00805*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Gretel Liz De la Pena Sarracén and Paolo Rosso. 2020. PRHLT-UPV at SemEval-2020 task 12: BERT for multilingual offensive language detection. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1605–1614.

T. C. Rajapakse. 2019. Simple transformers. `https://github.com/ThilinaRajapakse/simpletransformers`.

Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Marcos Zampieri, and Preslav Nakov. 2020. A large-scale semi-supervised dataset for offensive language identification. *arXiv preprint arXiv:2004.14454*.

Gudbjartur Ingi Sigurbergsson and Leon Derczynski. 2019. Offensive language and hate speech detection for Danish. *arXiv preprint arXiv:1908.04531*.

Stéphan Tulkens, Lisa Hilte, Elise Lodewyckx, Ben Verhoeven, and Walter Daelemans. 2016. A dictionary-based approach to racism detection in Dutch social media.

Cynthia Van Hee, Ben Verhoeven, Els Lefever, Guy De Pauw, Walter Daelemans, and Véronique Hoste. 2015. Guidelines for the fine-grained analysis of cyberbullying. Technical report, version 1.0. Technical Report LT3 15-01, LT3, Language and Translation ....

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. BERTje: A Dutch BERT model. *arXiv preprint arXiv:1912.09582*.

Shuohuan Wang, Jiaxiang Liu, Xuan Ouyang, and Yu Sun. 2020. Galileo at SemEval-2020 task 12: Multi-lingual learning for offensive language identification using pre-trained language models. *arXiv preprint arXiv:2010.03542*.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding abuse: A typology of abusive language detection subtasks. *arXiv preprint arXiv:1705.09899*.

Michael Wiegand, Melanie Siegel, and Josef Ruppenhofer. 2018. Overview of the GermEval 2018 shared task on the identification of offensive language.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019a. Predicting the type and target of offensive posts in social media. *arXiv preprint arXiv:1902.09666*.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019b. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval).

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. Semeval-2020 task 12: Multilingual offensive language identification in social media (OffensEval 2020). *arXiv preprint arXiv:2006.07235*.

A EXTENDED ANALYSIS

A.1 DATA

This section contains more graphs that describe the data, but are less relevant for the research in this thesis.

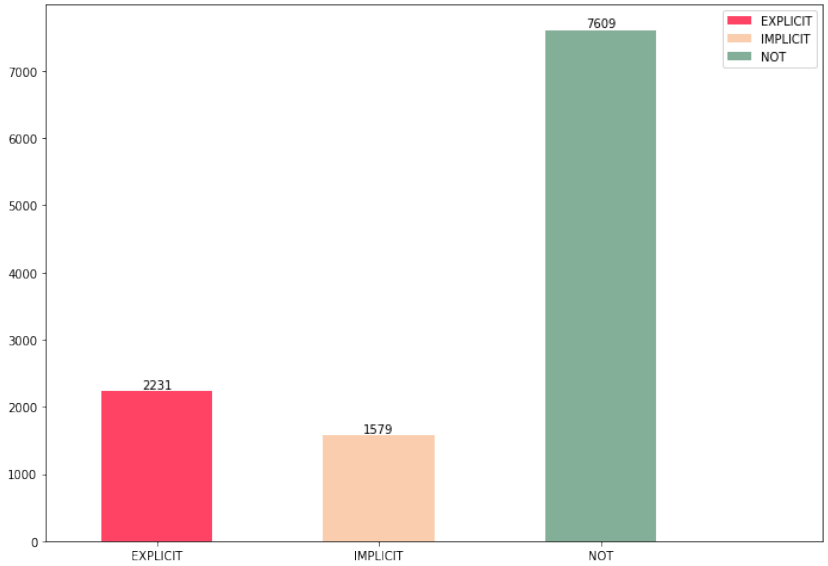


Figure 5: Class Distribution of Explicitness in the final dataset

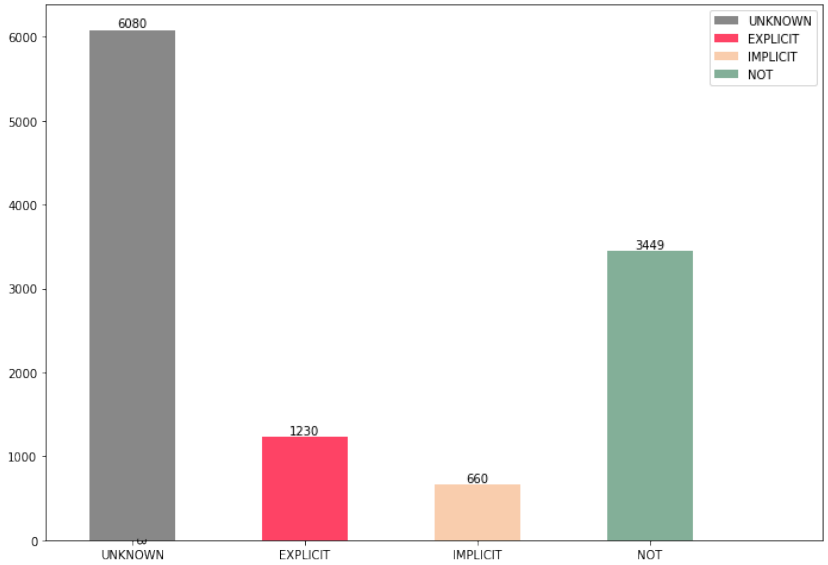


Figure 6: Class Distribution of Abusiveness in the final dataset

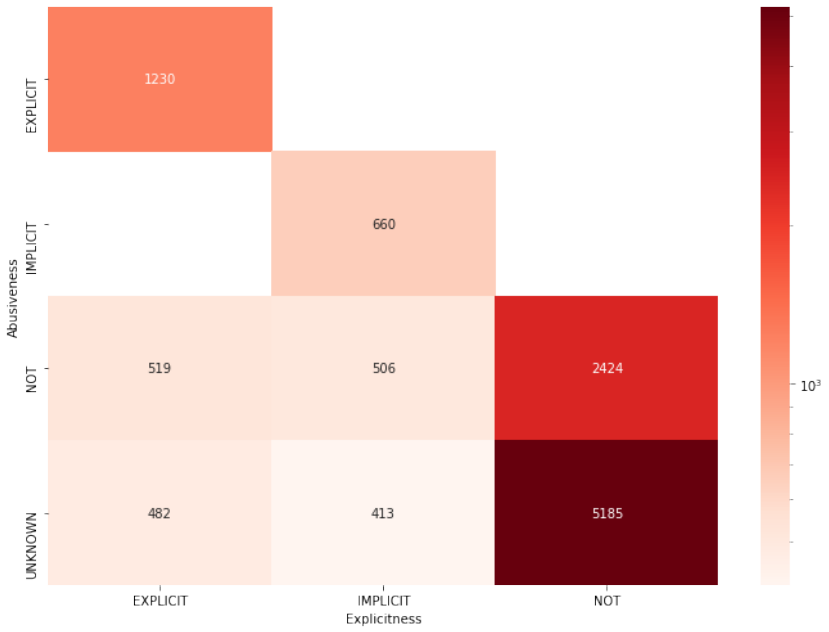


Figure 7: Explicitness vs Abusiveness

A.2 SVM FEATURE EXAMPLES

This section contains an example for each of the SVM feature functions.

- Bag of Words

```
> bag_of_words(word_tokenize('ondankbaar tuig... hup hup weg ermee.'))
{'.': True,
 '...': True,
 'ermee': True,
 'hup': True,
 'ondankbaar': True,
 'tuig': True,
 'weg': True}
```

- Counted Bag of Words

```
> counter(word_tokenize('ondankbaar tuig... hup hup weg ermee.'))
{'.': 1,
 '...': 1,
 'ermee': 1,
 'hup': 2,
 'ondankbaar': 1,
 'tuig': 1,
 'weg': 1}
```

- Bag of Words and Bigrams

```
> bag_of_bigrams_words(word_tokenize('ondankbaar tuig... hup hup weg ermee.'))
{('...', 'hup'): True,
 ('ermee', '.'): True,
 ('hup', 'hup'): True,
 ('hup', 'weg'): True,
 ('ondankbaar', 'tuig'): True,
 ('tuig', '...'): True,
 ('weg', 'ermee'): True,
 '.': True,
 '...': True,
 'ermee': True,
 'hup': True,
 'ondankbaar': True,
 'tuig': True,
 'weg': True}
```

- Bag of Hateful Words

```
> bag_of_hateful_words(word_tokenize('ondankbaar tuig... hup hup weg ermee.'))
{'hateful__tuig': True}
```

- Counted Bag of Hateful Words

```
> hateful_words_counter(word_tokenize('ondankbaar tuig... hup hup weg ermee.'))
{'hateful__tuig': 1}
```

A.3 RESULTS

This section contains the confusion matrix and classification report (sklearn.metrics.classification_report) of each model. These have been placed in the appendix to prevent cluttering the main text.

Table 9: Results: Dummy

	precision	recall	f1-score	support
INDIVIDUAL	0.4546	1	0.6251	471
GROUP	0	0	0	321
OTHER	0	0	0	176
NOT	0	0	0	68
accuracy			0.4546	1036
macro avg	0.1137	0.25	0.1563	1036
weighted avg	0.2067	0.4546	0.2842	1036

Table 10: Results: SVM baseline

	precision	recall	f1-score	support
INDIVIDUAL	0.6414	0.7558	0.694	471
GROUP	0.5907	0.5171	0.5515	321
OTHER	0.3309	0.2614	0.2921	176
NOT	0.3443	0.3088	0.3256	68
accuracy			0.5685	1036
macro avg	0.4768	0.4608	0.4658	1036
weighted avg	0.5535	0.5685	0.5574	1036

Table 11: Results: SVM improved

	precision	recall	f1-score	support
INDIVIDUAL	0.6755	0.8089	0.7362	471
GROUP	0.6763	0.5857	0.6277	321
OTHER	0.4017	0.267	0.3208	176
NOT	0.3377	0.3824	0.3586	68
accuracy			0.6197	1036
macro avg	0.5228	0.511	0.5108	1036
weighted avg	0.6071	0.6197	0.6073	1036

Table 12: Results: BiLSTM baseline (average over 5 runs)

	precision	recall	f1-score	support
INDIVIDUAL	0.5977	0.8688	0.7077	471
GROUP	0.6375	0.4636	0.5327	321
OTHER	0.3662	0.1852	0.2342	176
NOT	0.4694	0.1794	0.2532	68
accuracy			0.5819	1036
macro avg	0.5177	0.4243	0.4319	1036
weighted avg	0.5623	0.5819	0.5432	1036

Table 13: Results: BiLSTM improved (average over 5 runs)

	precision	recall	f1-score	support
INDIVIDUAL	0.7381	0.7868	0.7587	471
GROUP	0.6893	0.5732	0.6213	321
OTHER	0.4592	0.3966	0.4203	176
NOT	0.3104	0.4765	0.3735	68
accuracy			0.634	1036
macro avg	0.5493	0.5583	0.5434	1036
weighted avg	0.6475	0.634	0.6333	1036

Table 14: Results: BERTje baseline (average over 5 runs)

	precision	recall	f1-score	support
INDIVIDUAL	0.7148	0.8794	0.7879	471
GROUP	0.6952	0.6735	0.6824	321
OTHER	0.5026	0.3614	0.4197	176
NOT	0.5449	0.1265	0.2035	68
accuracy			0.6782	1036
macro avg	0.6144	0.5102	0.5234	1036
weighted avg	0.6615	0.6782	0.6543	1036

Table 15: Results: BERTje improved (average over 5 runs)

	precision	recall	f1-score	support
INDIVIDUAL	0.7911	0.8042	0.7975	471
GROUP	0.717	0.6629	0.6887	321
OTHER	0.5911	0.4807	0.5296	176
NOT	0.3689	0.6323	0.4658	68
accuracy			0.6942	1036
macro avg	0.617	0.645	0.6204	1036
weighted avg	0.7064	0.6942	0.6965	1036

Table 16: Results: RobBERT baseline (average over 5 runs)

	precision	recall	f1-score	support
INDIVIDUAL	0.7527	0.8509	0.7987	471
GROUP	0.7381	0.7259	0.7317	321
OTHER	0.5428	0.4057	0.4636	176
NOT	0.4752	0.3765	0.4104	68
accuracy			0.7054	1036
macro avg	0.6272	0.5897	0.6011	1036
weighted avg	0.6943	0.7054	0.6955	1036

Table 17: Results: RobBERT improved (average over 5 runs)

	precision	recall	f1-score	support
INDIVIDUAL	0.7682	0.8178	0.7921	471
GROUP	0.7425	0.691	0.7157	321
OTHER	0.5848	0.4591	0.5125	176
NOT	0.376	0.5353	0.4414	68
accuracy			0.699	1036
macro avg	0.6179	0.6258	0.6154	1036
weighted avg	0.7033	0.699	0.6979	1036

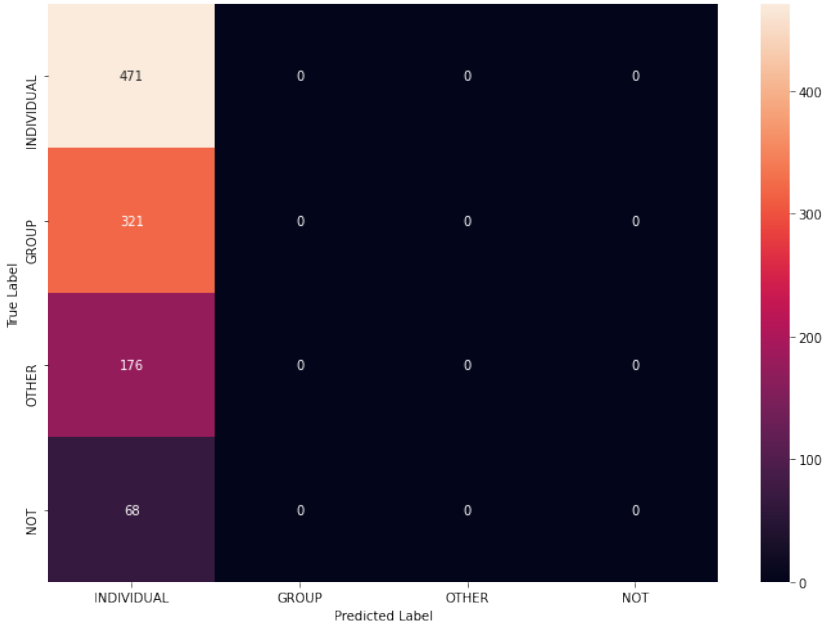


Figure 8: Confusion Matrix Dummy Classifier

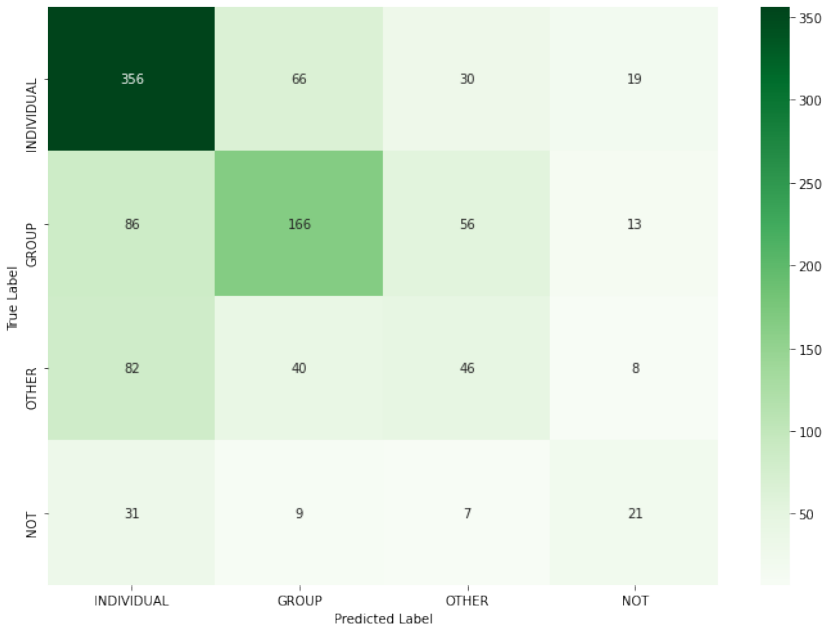


Figure 9: Confusion Matrix SVM baseline

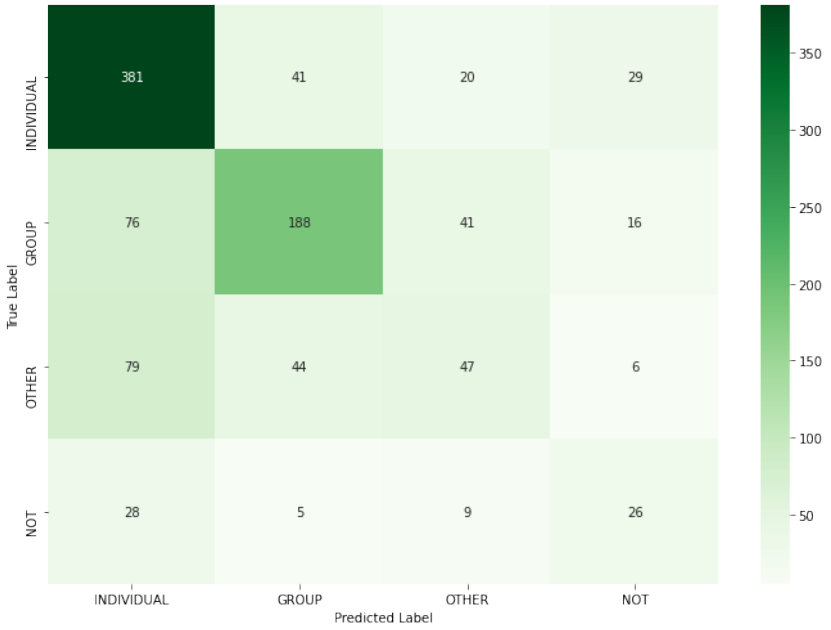


Figure 10: Confusion Matrix SVM improved

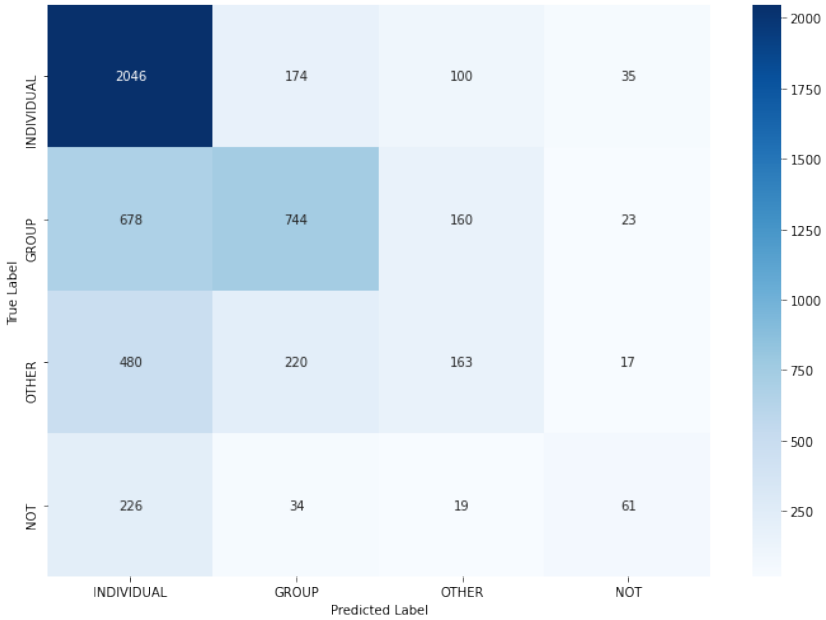


Figure 11: Confusion Matrix BiLSTM baseline (5 runs totalled)

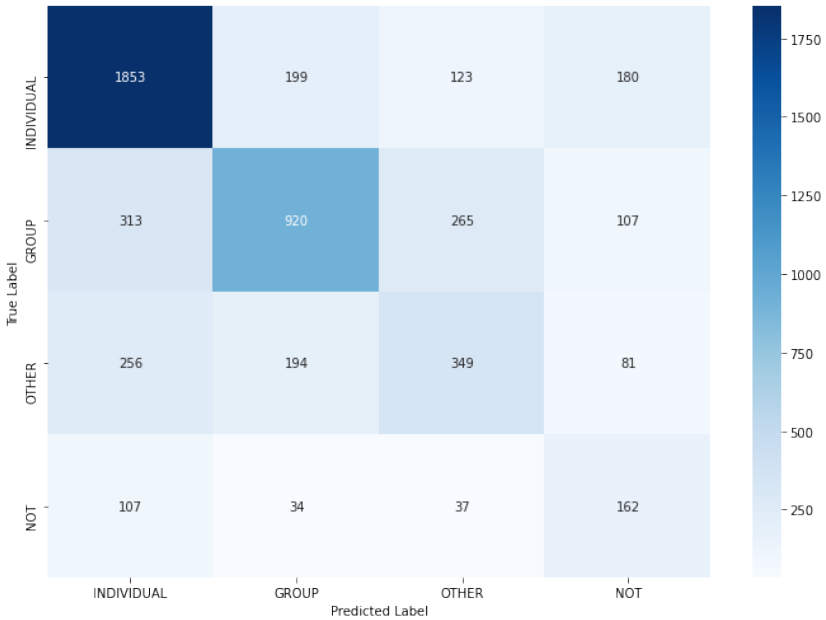


Figure 12: Confusion Matrix BiLSTM improved (5 runs totalled)

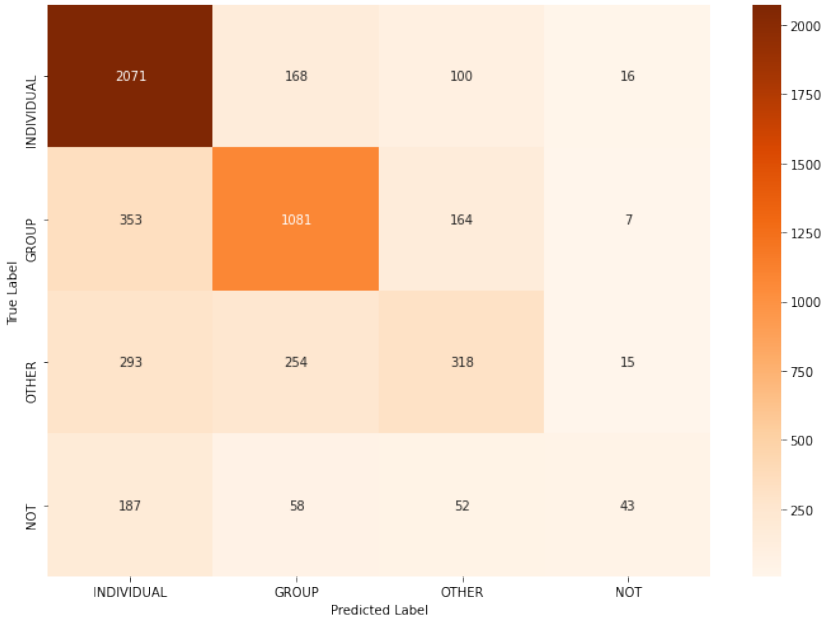


Figure 13: Confusion Matrix BERTje baseline (5 runs totalled)

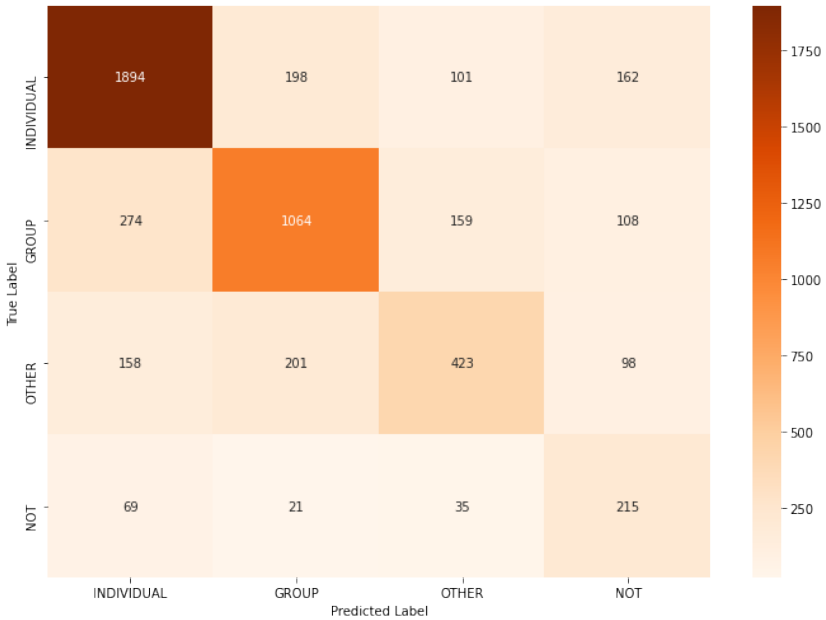


Figure 14: Confusion Matrix BERTje improved (5 runs totalled)

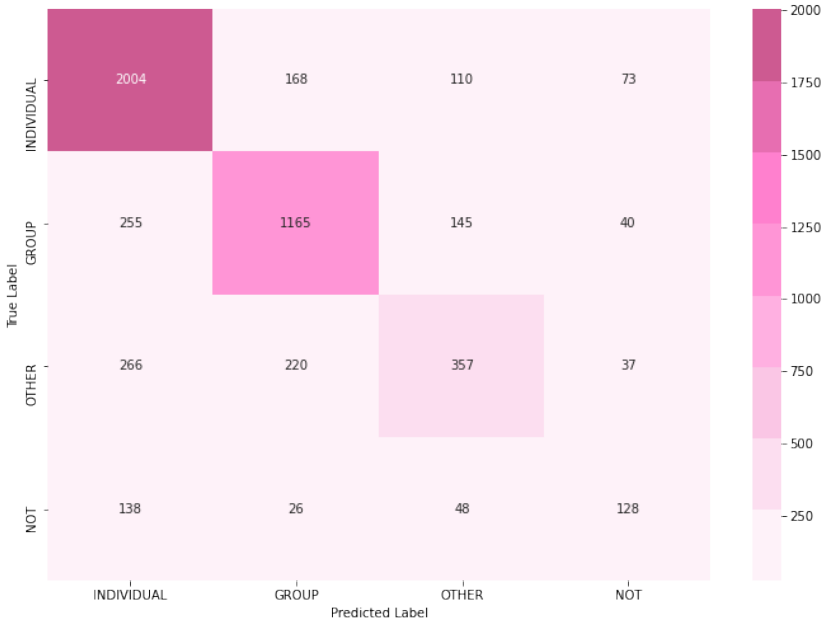


Figure 15: Confusion Matrix RobBERT baseline (5 runs totalled)

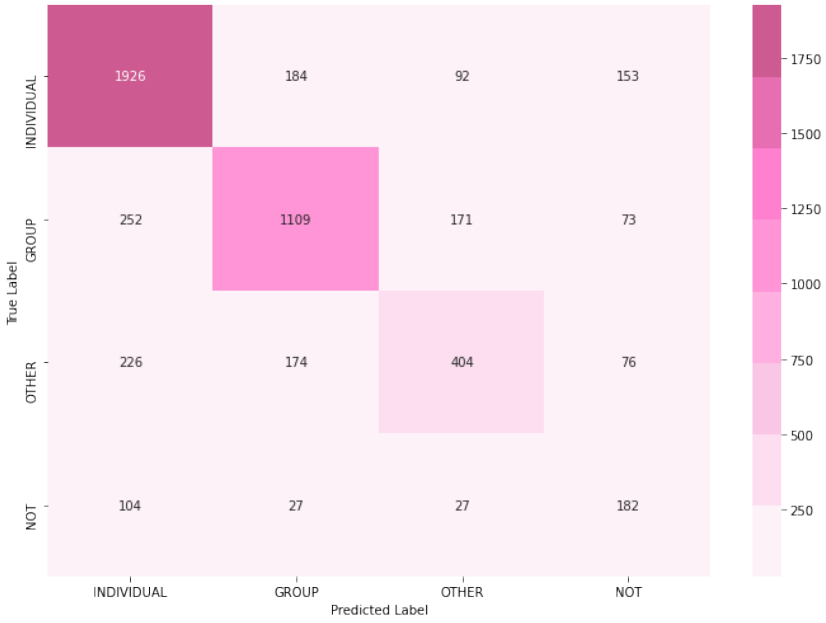


Figure 16: Confusion Matrix RobBERT improved (5 runs totalled)