FRI Natural language processing course 2021
Project report

University of Ljubljana
Faculty of Computer and Information Science

Cross-lingual offensive language identification

Maj Šavli, Blaž Rupnik and Leon Premk

Abstract

The abstract goes here.

Keywords

Keyword1, Keyword2, Keyword3 ...

Advisors: Slavko Žitnik

Introduction

Since the outburst of social media, freedom of speech has allowed anyone to share their opinion on internet. While that allows people to make a change, it can also have negative consequences. Offensive language or hate speech has become a constant on online forums [1]. Best definition of online hate speech we can use are hateful messages (posts on social platforms, comments on news articles) directed against an individual or a group of individuals based on their identity. Because of these messages the group can be viewed as undesirable which warrants hostility towards them.

That's why automatic offensive language detection is highly required task. Some solutions for hate speech detection already exist but most are in english. In this project we tackled offensive language classification using traditional machine learning approaches and state-of-the-art models such as mBERT and XLM-R on english datasets ([2, 3, 4, 5, 6]) and later transfer our models to Slovenian language. We plan to use different ensemble methods to further improve classification models.

Related work

With rapid growth of information on internet, automatic tools for detecting hate speech are in huge demand. Earlier implementations of offensive language detection were based on basic machine learning classifiers such as naive bayes and SVM. By increasing hardware capabilities in recent years deep learning methods became became the new state-of-the-art outperforming previous methods by large margin.

Pitenis et al. [7] used deep learning methods to detect offensive language in Greek Tweeter posts. In another work Rizwan et al. [8] proposed their Convolutional Neural Network n-gram to detect hate speech on dataset containing Roman Urdu tweets. Ranasinghe et al. [9] used different state-of-the-art natural language processing methods such as BERT and XLM to detect offensive language in Bengali, Hindi and Spanish social media posts. In OffensEval 2020 [10] competitors were detecting offensive language, categorizing it based on offense type and identifying toward whom offense was targeted. Datasets were in English, Arabic, Danish, Greek and Turkish language. Most teams used pre-trained Transformers such as BERT [11] and it's variations like RoBERTa [12], or AL-BERT [13]. Other Transformers, most notably GPT-2 [14], were also used for classification. Word embeddings were mostly done by BERT or RoBERTa and BERT's multilingual variant mBERT [11].

Methods

Data

English dataset consist of five datasets from different sources. They mainly consist of social media posts on Twitter [5], Reddit and Gab [3] aswell as Wikipedia posts [2], news articles [4] and forum Stormfront posts [6]. Lots of acquired data is politicaly oriented, which can be benefitial since majority of Slovenian offensive language has political base.

One part of the Slovene dataset was acquired from scraping a slovenian news platform, 24UR. We extracted user comments of various articles. To achieve better variance in data, we made sure the scraped comments belonged to articles of various themes. There were a lot of emoticons present in the comments. Since they could negatively impact on the learning of the algorithms, we removed them.

Another Slovene dataset was acquired from [15], which is a Slovenian Twitter hate speech. It contains roughly 120000 labeled tweets. Each text label falls to one of the four classes: appropriate, inapproriate, offensive or violent. We merged classes in such a way, that it became a binary problem (offensive/non offensive), labeling inappropriate and violent both as offensive language as well.

Our Slovene datasets contain very mild offensive language compared to the english ones. Thats why we will also make another Slovene dataset with more violent hate speech to match the chosen english datasets and hopefully get better results classifying offensive speech.

Data preprocessing

In many cases, especially in text classification or translation text prepocessing can improve the accuracy of algorithms and models. Before input text, presented in natural form, can be passed to machine learning algorithms, it needs to be cleaned of unnecessary words. This way, the machine learning model can focus and learn on the words, which hold the useful information.

The first step was removing all the extra whitespaces and tabs, since they don't hold any information. Second step was changing all characters to lowercase characters, which helps the whole process of text processing and it can be beneficial in other ways, such as parsing. We want every word to be normalized and in original form and not in form of contractions, so the third step was to extend all the recognized contractions. The fourth step was to remove all special characters, since such characters add no value to text understanding and are considered as noise. Removing numbers can sometimes be useful, sometimes not. It depends on what problem you are solving. Since we deal with classification of hateful and offensive language we decided that numbers do not play a major role and can be removed as well. The last and a very import step was stemming. Stemming is process of reducing every word to its root. This is done by removing unneccessary characers, most often a suffix. The stemmers, which we used, are two of the most known stemming models, Porter and Snowball. The models, however, have flaws and do not stem every word correctly and removing or changing information from text.

Classification using traditional approaches

For classifying offensive language on english datasets we also tried using a few traditional approaches, meaning neural network are not involved, so statistical models like logistic regression, support vector machine and random forest which constructs multitude of decision trees that are used for classification. In two subsections we first describe how we constructed features from specific dataset and then we give some results for classification of offensive language.

Extracting features

We seperated extracting features into three different steps. First one was defining sentiment score for each sample in the data. For this we used a list of known english hate words, which we retrieved from Hatebase repository [16]. So for each sample we calculated what percentage of words are hate words from this hate words list.

Second step was extracting bigrams from the whole dataset and adding each bigram as a seperate column and defining binary value, meaning value is yes if sample has correlated bigram else no. Bigram is a sequence of two adjacent words. Since number of columns would be huge if used all bigrams we decided to set lower limit to 0.5% which means that we ignore terms that appear in less than 0.5% samples. Before performing this we also stemmed the data since many charged words can have different derivations.

Last set of features we produced are so called tf-idf features. Their purpose is to reflect how important a word is in the whole dataset. We extracted 50 terms with the biggest tf-idf weight and again used them as a seperate columns where values tell us if given sample contains correlated term. Weight of a term is determined by term frequency (tf) that tells us how many times a term occurs in one particular sample and inverse document frequency (idf) which tells us in how many samples the terms appears. When we performed this on our Twitter dataset we found out that big number of the 50 terms are slurs.

Classification

Before starting classification we created one feature dataset that consists of all three feature subsets described in the previous chapter. We split the data into test and training set with test set consisting of 20% of the data. In table 1 we can see accuracy results for three different models in classifying offensive tweets in the given Twitter dataset. As the last one we also added the accuracy for dummy classifier, meaning it always picks the class that is the most frequent.

Table 1. Classification accuracy using three different traditional models

Model	CA
Logistic Regression	0.929
Support Vector Machine	0.926
Random Forest	0.933
Dummy	0.777

Classification with BERT

When using BERT for classification we are using multi-language pre-trained model, that is trained for 104 languages. This allows simple transfer between languages. Output of BERT model is fed into two dense layers - one hidden with 128 parameters and output layer having same number of parameters as our classes. Firstly we split our data on two parts - offensive and not offensive, so we are dealing with binary classification. Later we will expand offensive class on subclasses. Firstly we used BERT to classify english tweets only. BERT scored CA 0.904. When training our model to predict Slovenian data we tried three different training approaches. Firstly we trained our model on English data and tested on Slovenian data, then we trained on both English and Slovenian data and tested on Slovenian data and lastly we trained only on Slovenian data and tested on Slovenian data. Results in terms of classification accuracy are shown in table 2. Majority class presents 68% of all data, meaning models have to score higher than that to be considered useful.

Table 2. Classification accuracy using three different approaches

Training data	CA
ENG	0.636
SLO	0.706
ENG + SLO	0.739

Discussion

Traditional approaches

As seen in results we got high accuracy on all three classifiers in comparison to the baseline one where we simply select the class with highest frequency. In continuation we plan to add other perfomance mesurements like F_1 score and AUC because accuracy is not ideal when working with imbalanced data. We will also add ensemble methods like weighted-average voting for combining different models and hopefully generating better one.

BERT

It's interesting to see that when we are training data on English data classification results are worse. This is due to different types of offensive language. Training on both English and Slovenian dataset brings around 3% improvement. Our current english data consists mostly of personal attacks and vulgar speech, while Slovenian texts mostly cover political hate speech, which brings to low correlation between classes. This is why we have to expand our English training data and divide it on different subclasses. In the future we will also use ensemble voting to combine BERT with XLM, possibly improving our classification accuracy.

References

[1] Konrad Rudnicki and Stefan Steiger. Online hate speech - introduction into motivational causes, effects and regulatory contexts. 08 2020.

[2] Ellery Wulczyn, Nithum Thain, and Lucas Dixon. Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th international conference on world wide web*, pages 1391–1399, 2017.

[3] Jing Qian, Anna Bethke, Yinyin Liu, Elizabeth Belding, and William Yang Wang. A benchmark dataset for learning to intervene in online hate speech. *arXiv preprint arXiv:1909.04251*, 2019.

[4] Lei Gao and Ruihong Huang. Detecting online hate speech using context aware models. *arXiv preprint arXiv:1710.07395*, 2017.

[5] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11, 2017.

[6] Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. Hate speech dataset from a white supremacy forum. *arXiv preprint arXiv:1809.04444*, 2018.

[7] Zeses Pitenis, Marcos Zampieri, and Tharindu Ranasinghe. Offensive language identification in greek. *arXiv preprint arXiv:2003.07459*, 2020.

[8] Hammad Rizwan, Muhammad Haroon Shakeel, and Asim Karim. Hate-speech and offensive language detection in roman urdu. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2512–2522, 2020.

[9] Tharindu Ranasinghe and Marcos Zampieri. Multilingual offensive language identification with cross-lingual embeddings. *arXiv preprint arXiv:2010.05324*, 2020.

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

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

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

[13] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[14] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[15] Petra Kralj Novak, Igor Mozetič, and Nikola Ljubešić. Slovenian twitter hate speech dataset IMSyPP-sl, 2021. Slovenian language resource repository CLARIN.SI.

[16] Hatebase. https://hatebase.org/.