Hate Speech Classification in Bulgarian

Radoslav Ralev
Technical University of Munich
Department of Informatics
80333 Munich, Germany
radoslav.ralev@tum.de

Jürgen Pfeffer
Technical University of Munich
School of Social Sciences and Technology
80333 Munich, Germany
juergen.pfeffer@tum.de

Abstract

In recent years, we have seen a surge in the propagation of online hate speech on social media platforms. According to a multitude of sources such as the European Council, hate speech can lead to acts of violence and conflict on a broader scale. That has led to increased awareness by governments, companies, and the scientific community, and although the field is relatively new, there have been considerable advancements in the field as a result of the collective effort. Despite the increasingly better results, most of the research focuses on the more popular languages (i.e., English, German, or Arabic), whereas less popular languages such as Bulgarian and other Balkan languages have been neglected. We have aggregated a real-world dataset from Bulgarian online forums and manually annotated 108,142 sentences. About 1.74% of which can be described with the categories racism, sexism, rudeness, and profanity. We then developed and evaluated various classifiers on the dataset and found that a support vector machine with a linear kernel trained on character-level TF-IDF features is the best model. Our work can be seen as another piece in the puzzle to building a strong foundation for future work on hate speech classification in Bulgarian.

Keywords: hate speech, natural language processing, classification, Bulgarian

1 Introduction

The term "hate speech" means public speech that expresses hate or encourages violence toward a person or group based on race, religion, sex, or sexual orientation[1]. Hate speech is not something new. We can find evidence of it throughout history ranging from Ancient Greece, through Rome and the middle ages up to modern times. It is no surprise that during times when the most prominent thinkers were freely expressing their hateful opinions and discrimination against minorities was part of both the law and religion, hate speech was omnipresent. However, identifying hate speech is a complex problem. Who decides what hate speech is? Aristotle would probably not consider his writings hateful, but two thousand years later, we might.

Today, social media platforms can enable people with discriminatory views to express their opinions more openly and under anonymity. Furthermore, there have been multiple occasions in which there is a connection between online hate speech and increased violent hate-based activities. Two very prominent examples of increased hate speech online following real-world events are a) hate speech towards immigrants and Muslims following the Manchester and London attacks after the UK left the EU. (Travis, 2017); b) an uptick in racist and xenophobic harassment incidents following the Presidential election in the US. (Okeowo, 2017). By the year 2020 hate crime had already achieved global recognition. In total, 118 countries and international organizations have laws on hate speech[2].

The connection between hate speech and hate crime has also already been studied more thoroughly in academia (Müller and Schwarz, 2021). In general, studying human behavior at scale by utilizing social media data has been the focus of researchers' attention for 15 years (Lazer et al., 2009). While much research has been devoted to big platforms like Facebook and Twitter and focuses on a small number of languages, more recently, research on smaller and more specialized communities (Mooseder et al., 2022) and less popular languages (Nurce et al., 2021; Shekhar et al., 2020; Ljubešić et al., 2018) has become increas-

[1] www.dictionary.cambridge.org/us/dictionary/english/hate-speech

[2] www.futurefreespeech.com/global-handbook-on-hate-speech-laws

ingly visible. We follow this branch of research and focus our attention on content in a language underrepresented in scientific research, namely Bulgarian.

Bulgarian is a language spoken by approximately 8 million people around the globe, however, it plays an important historical role as the first Slavic language to have an official alphabet. It was created and developed in the 9th century AD by the Saints Cyril and Methodius and their disciples. It was the first Slavic language into which the Bible was translated.

Although not as damaging as the examples mentioned above, Bulgaria suffers from an extremely high incidence of hate speech towards representatives of ethnic, religious, or sexual minorities (Lozanova et al., 2017; Ivanova, 2018). Figure 1—showing the sentiment distribution of 1,475 comments in Bulgarian following the Syrian refugee wave—illustrates the gravity of the problem. Article 162, paragraph 1 of the Bulgarian penal code penalizes the more extreme forms of hate speech, hence one can conclude that Bulgaria currently suffers from two problems in terms of hate speech prevention. First, the detection of hate speech and encouragement towards violent acts. Second, is the enforcement of the law. This paper aims to address the first of these two points by collecting, filtering, and manually annotating real-world data, and by implementing, evaluating, and comparing various supervised learning models.

The contributions of this article are:

- We have manually annotated 108,142 Bulgarian sentences and made this dataset publicly available. [3]
- 1,878 of these sentences can be described as being hate speech, namely in the categories racism, sexism, rudeness, and profanity.
- We have tested multiple classifiers and approaches on the dataset and compared their performances.
- The best model in terms of F1 score is a support vector machine with a linear kernel trained on character-level TF-IDF features. The model achieved a macro F1 score of 0.73.

[3] http://www.pfeffer.at/data/bulgarian/

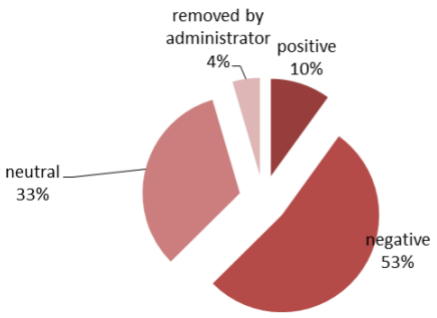

Figure 1: Sentiment distribution of 1,475 comments in Bulgarian, following the Syrian refugee wave (Lozanova et al., 2017).

2 Related work

The rising visibility of hate speech on the online social platform has resulted in a continuously growing rate of published research into different areas of hate speech (Tontodimamma et al., 2021). Due to the enormous volume of data that needs to be checked, more focus has been put on automatic detection algorithms.

Detecting hate speech has become an essential topic in the natural language processing community (Mohiyaddeen and Siddiqi, 2021). As a result, a wide range of approaches to text classification was applied, and new datasets were created (Waseem and Hovy, 2016). The issue is that some more minor, less represented languages go under the radar. There have been efforts for language-agnostic text classification (Feng et al., 2020), however, these languages remain mainly ignored by the scientific community. Bulgarian, for example, is one such language.

Some efforts (Dinkov et al., 2019) have been made toward detecting toxicity in news articles in Bulgarian, but the datasets tend to be too small. In recent years there have been numerous advances in natural language processing conducted by Bulgarian researchers on various topics. In (Koeva et al., 2020) the authors present a new corpus of national legislative documents. In (Zhikov et al., 2012) a multi-class multi-label classifier for social news is presented. In (Marinova, 2019) the author compares the performance of classifiers trained on features generated by a variety of state-of-the-art pre-trained embeddings models for tasks such as Named Entity Recognition and Classification (NERC) and Part-of-Speech (POS) Tagging. In (Kapukaranov and Nakov, 2015) a movie review dataset in Bulgarian, sentiment lexicon, and a first-

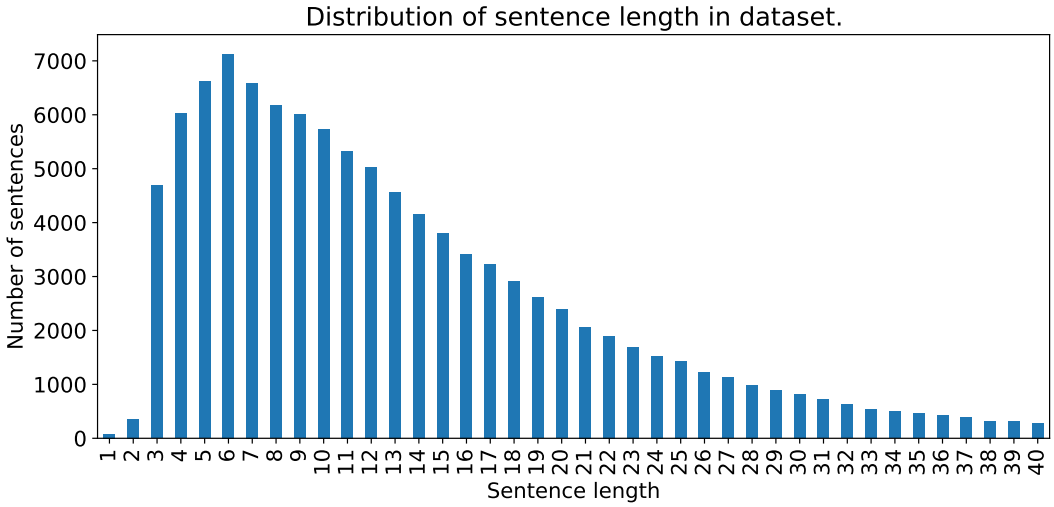

Figure 2: The sentence lengths in our dataset follow a long tail distribution. (Tail has been cut at 40 for better readability.

of-its-kind fine-grained sentiment classifier are presented. Word normalization methods such as stemming (Nakov, 1998) and lemmatization (Iliev et al., 2015) have also been explored, enabling more advanced natural language processing pipelines, sentiment analysis, and others.

To address the problem of hate speech. an automatic detection algorithm has to be created. Usually, this is done by training a machine learning model in a supervised manner for which huge amounts of annotated data are required. Some authors (Waseem and Hovy, 2016) also incorporate social network data features in the model training, however, we have abstained from this and focused explicitly on natural language.

3 Data

The data required for our purpose was natural language written informally in Bulgarian and, if possible, written as part of a dialogue or a comment on a subject.

The biggest portion of data was directly provided by the Bulgarian forum BG-Mamma[4] which is the biggest Bulgarian forum and its main user base is mostly comprised of current or future parents. Except the data provided by them we also scraped other forums such as BG-Jargon[5] and BG-Nacionalisti[6] (BG-Nationalists). BG-Jargon is a website that collects Bulgarian slang words and phrases and includes example sentences of how each word is used in everyday life. We have scraped exactly those sentences. BG-Nationalists is an extremist right-wing political forum. About 80% of the data is from BG-Mamma.

The sentences consist on average of 14.3 words (median 11, standard deviation 12.2). All of the websites above contain mostly informal communication. This is further confirmed by the distribution of the sentence length as seen in Figure 2 with moth sentences being short but also having a very long tail. While we can find one ”sentence” with 685 words, 75% of sentences are 18 words or less. Due to the nature of the main source of the data (BG-Mamma) we were expecting predominantly non-hateful sentences.

3.1 Labeling

Text classification is almost always performed in a supervised way. For this reason, a labeled dataset is required.

At first, we approached this by manually labeling entire ”comments” or ”opinions” which are multi-sentence posts, however, after a few thousand samples we noticed that within multi-sentence hateful posts, hate usually occurs within only one sentence, hence we decided to do sentence classification instead. We split the initial ”comments” dataset into a sentence dataset which greatly increased the sample count. The final result was a total of 108,142 manually annotated sentences, Unfortunately, even before the split, the data was severely imbalanced.

[4]www.bg-mamma.com
[5]https://www.bgjargon.com/
[6]https://bg-nacionalisti.org/

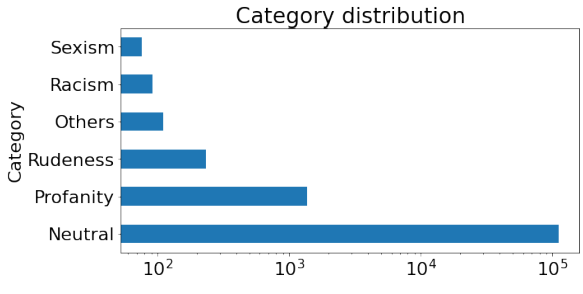

Figure 3: Distribution of each sentence class in the dataset. The x-axis is in a logarithmic scale.

The split made the imbalance even greater as you can see in Figure 3. The major disproportion in the dataset forced us to unify all hateful categories into one and perform a simple binary classification. This led to a dataset with 106,264 non-hateful and 1,878 hateful sentences.

3.2 Data preprocessing

Text data is one of the most disorganized and unstructured data types possible. That makes data preprocessing one of the deciding factors for the final quality of a model.

Cleaning the BG-Mamma dataset required the most time out of all the text gathered. Originally the text was in a BBCode-format[7]. BBCode tags and other format-specific syntax were removed to clean the text. An algorithm to eliminate posts appearing once as a standalone comment and a second time when they were being mentioned was also developed. Aside from this, HTML code, URLs, punctuation, stopwords, and all numbers were removed. The text was also made lowercase.

Lemmatization in linguistics is the act of grouping together different word forms so that a text processing algorithm can recognize them as a single word. In itself, lemmatization is complex because it has to identify the word on a part-of-speech level. For this project, lemmagen3[8] was used.

Stemming is usually considered a more naive version of lemmatization. That is due to the fact that stemming does not consider the context of the word, but only its morphology. Stemming removes or stems the last few characters of a word, often leading to incorrect meanings. In (Nakov, 1998) the author argues that stemming and lemmatization have achieved a similar performance in experiments. The stemmer described in that paper was also used in this project. Along with the software package[9], different rule sets are provided. All of them are included in the evaluation. Both the lemmatizer and the stemmer were evaluated and compared.

Vectorization. As already mentioned, text data is one of the most unstructured data types. One of the worst qualities is that it is of variable length. To offset that, the so-called vectorization is performed. Vectorization is the process of transforming unstructured text into a fixed-size numerical representation (usually a vector) that is easier to understand by a machine (Schütze et al., 2008). There are various ways to do this from simple bag-of-words or bag-of-characters methods and the famous TF-IDF to neural network embeddings (Bengio et al., 2000) using pre-trained models such as BERT (Devlin et al., 2018) or Word2Vec (Mikolov et al., 2013). We have primarily focused on TF-IDF, however, embeddings we have also evaluated some pre-trained embeddings such as the stacked embeddings from FlairNLP (Akbik et al., 2018, 2019), FastText (Joulin et al., 2016) and others.

Data imbalance As previously mentioned, the dataset is significantly imbalanced (Günnemann and Pfeffer, 2017). There are various approaches to offset this. For this, we focused on imbalanced-learn's and scikit-learn's implementations (Lemaître et al., 2017; Pedregosa et al., 2011). One can address this issue by oversampling the minority class, undersampling the majority class, or a mix of both. Two of the most basic approaches to handling imbalanced data consist of either replicating the minority class samples until the class distribution becomes uniform or providing class weights for each class to the classifier which will correct the loss function correspondingly.

A more advanced technique for oversampling is called "Synthetic Minority Over-sampling Technique" or just SMOTE (Chawla et al., 2002). The algorithm works by finding the nearest neighbor of a sample point from the minority class in feature space. Then it chooses a random point between them, which is then added to the dataset. This algorithm's effectiveness has been thoroughly evaluated and usually achieves a performance boost, although some authors suggest that the commonly accepted method for synthetic instance creation may not be the best one (Bajer et al., 2019).

[7] https://en.wikipedia.org/wiki/BBCode
[8] https://github.com/vpodpecan/lemmagen3/
[9] https://pypi.org/project/bulstem/

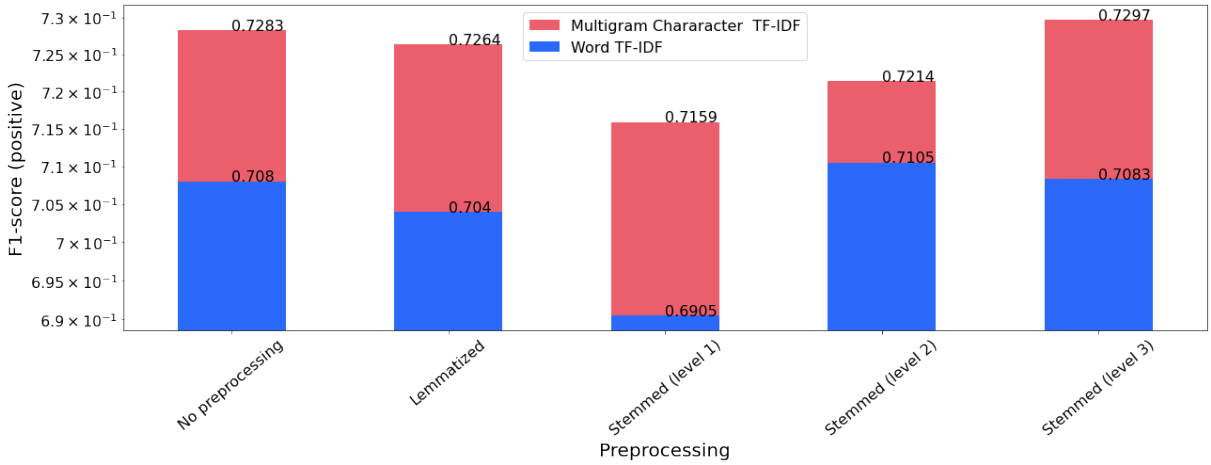

Figure 4: Comparison between the performance of preprocessors with two different vectorizers. (log-scale)

4 Models

For the classification, we trained classical machine learning models such as the logistic regression, support vector machines (Platt et al., 1999), decision trees (Breiman et al., 2017), random forests (Breiman, 2001) and naive bayes classifiers (Schütze et al., 2008).

We also evaluated the performance of several neural network architectures. The most basic of which is a shallow neural network with Keras' (Chollet et al., 2015) embedding layer and TensorFlow's (Abadi et al., 2015) TextVectorization layer. Another architecture we evaluated is the one discussed in (Zhang et al., 2015). We used another architecture that was based on pre-trained Word2Vec embeddings which was fine-tuned on the corpus, and its weight matrix was used to set the weights of a Keras embedding layer. After that, the architecture proposed above for the Character-Level-CNN was used again. Lastly, we also used the stacked embeddings model for text classification provided by the FlairNLP framework in a similar fashion as in (Marinova, 2019).

5 Analysis

The disparity in the distribution of the categories made us rethink how we should observe the classifiers' performance. A dummy classifier predicting only one class has 98% accuracy. For this reason, the primary metric we used is macro F1 which is an arithmetic mean of the F1-Score for both classes and also the positive F1 score. The imbalance is ignored by using the mean of the two scores, and the two classes are equally weighted.

However, other metrics can also be chosen, such as balanced accuracy, if the classifier is to be used in more practical settings (e.g., in the industry). *Balanced accuracy* is defined as the average of recall for all categories in the classification. It is used as a substitute for accuracy for imbalanced datasets. It is also much easier to interpret than F-Score.

For the evaluation, the dataset was split into a training and testing set (75% train, 25% test). A 75-25 proportion instead of 70-30 was used because it allowed for a better distribution of the main five categories in both datasets. Furthermore, although binary classification was performed, due to the data imbalance, the data was still split following a stratified approach for all five categories to achieve a similar distribution in both datasets. That was done to offset any additional bias towards one of the categories.

5.1 Comparing preprocessing techniques

Stemming vs. Lemmatization Before evaluating the performance differences in vectorizers, we wanted to see which preprocessing technique was the best. To do that, we prepared five pipelines: one without any preprocessing, one with lemmatization enabled, and three with stemming enabled, each with a different, more punishing, rule-set. At first, we used only the basic word TF-IDF vectorizer but later, after finding the best vectorizer (see following subsections), we decided to re-do this evaluation. The classifier used is scikit-learn's support vector machine implementation with a linear kernel (also called LinearSVC), but similar outcomes were observed with other classifiers.

Figure 4 depicts the results. The results show that stemming outperforms lemmatization. Espe-

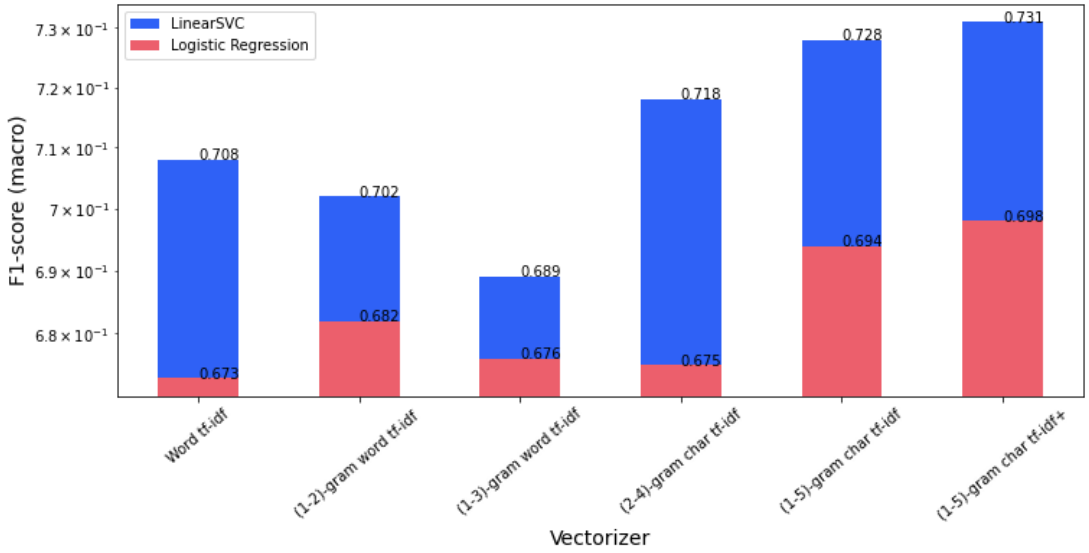

Figure 5: Performance comparison of vectorizers

Classifier	(macro) f1-score	precision	recall
SimpleNN	0.686	0.4246	0.347
Word2Vec	0.6523	0.2675	0.3821
Word2Vec+CNN	0.6850	0.3232	**0.4615**
FlairNLP	**0.7172**	**0.5193**	0.3860
Character-Level CNN	0.6359	0.2828	0.2808

Table 1: Performance comparison between all neural networks.

cially in a setting where words are used as features, level two stemming performs best. However, omitting to preprocess helps boost classifier performance when using character-level features, and despite the better performance, at first sight, we believe that omitting the stemming adds more robustness to the model when using a character-level vectorization. This is because the Bulgarian language is very rich in prefixes and suffixes and stemming at such a high level might disrupt the meaning of a word. Hence we have decided to stick to no preprocessing and unless otherwise specified, everything will be evaluated on a dataset without lemmatization or stemming in the following subsections.

Vectorizers In total, six vectorizers were evaluated on two different classifiers—a logistic regression and a linear support vector machine (LinearSVC). Three word-level and three character-level vectorizers were chosen. The classifiers' macro F1-Score performances are visualized in Figure 5. From the Figure, it can be seen that for both classifiers, the character-level preprocessing tends to outperform the word-level vectorization. Hence, unless otherwise specified, from this point onward, everything will be evaluated on data with character-level uni- to pentagrams.

Another key consideration is that some character n-grams that are too often seen in the dataset can be ignored due to the enormous class imbalance. This parameter is called maximum document frequency. The rightmost bar on the figure ("(1-5)-gram char tf-idf+") is the same as the one before it but includes a maximum document frequency of 40% as well. As it can be seen, although it does offer an increase in performance, it is more or less negligible.

5.2 Models

Neural networks For the more basic network, TensorFlow's Text Vectorization layer was used, again at a character level (but this time without TF-IDF enabled due to an immense increase in training times), followed by an embedding layer with an output dimension of 64. After that, the output of the embedding layer goes through a 1D max. pooling layer to reduce the dimensions and is consequently fed into a sequence of three dense layers, each with 64 neurons and a ReLU activation. We have not

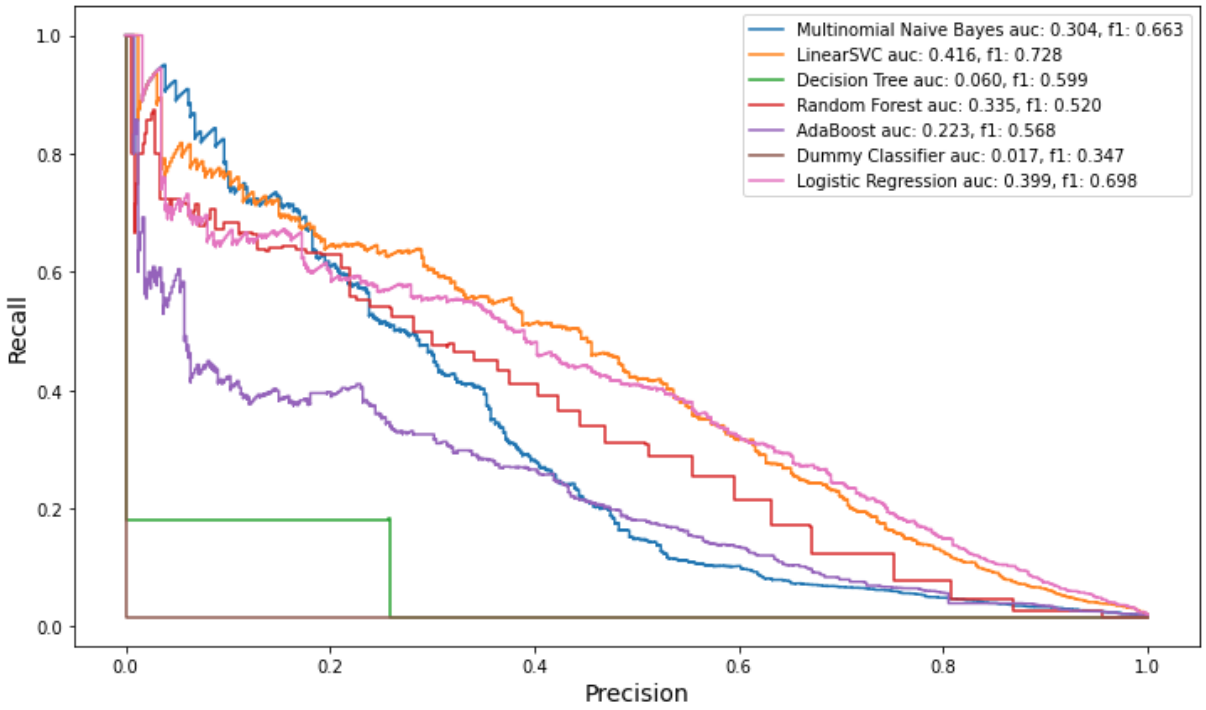

Figure 6: Precision-Recall-Curves for all classical machine learning classifiers without oversampling.

included any of the more simple neural networks with LSTM/Convolutional layers in this evaluation as they did not increase the performance of the model significantly enough.

The other networks are the FlairNLP stacked embeddings network, the Character-Level CNN from (Zhang et al., 2015), and the Word2Vec embeddings network (once with convolutional layers and once without). The results are summarized in Table 1.

All in all, the FlairNLP stacked embeddings model achieved the best performance. It is also the slowest model to train and uses the most pre-trained embeddings (four pre-trained models in total). The Word2Vec model with the CNN architecture and the simple neural network come in close second and third. An interesting note is that the Word2Vec+CNN model achieved the best recall score. A surprise was the performance of the Character-level CNN. It is the second-largest model on the list with a total of 96M parameters but it performed worse even than the simple neural network.

Classical machine learning models Firstly, a naive dummy classifier was created to benchmark the other models. The dummy classifier predicts using a uniform strategy, so each class has equal probability. After that, seven classifiers with default parameters were trained once on the dataset preprocessed as discussed in previous sections and once on the same dataset but additionally with SMOTE oversampling enabled. Surprisingly, most of the classifiers either underperformed or showed no significant improvement on the oversampled dataset and were thus omitted for the sake of brevity.

Classifier	Balanced Accuracy
SimpleNN	0.6695
Word2Vec+CNN	**0.7224**
LinearSVC	0.4202
Logistic Regression	0.5350

Table 2: Performance comparison between selected models in terms of balanced accuracy.

The results are shown in Figure 6. The classifiers are compared based on their precision-recall curves, as well as the overall area under the curve (auPRC) and f1-score. The overarching winner in both setups was the linear support vector classifier with logistic regression coming in a close second. In the end, the LinearSVC managed to achieve an f1-score of **0.728**.

5.3 Balanced Accuracy

As previously mentioned, although F1 is the standard metric for comparing classifiers, in a more practical setting, better metrics can be found. The

main reason for this is that F1 is not as easy to interpret as other metrics may be. An excellent example of a suitable metric for a scenario like that is balanced accuracy.

Much to our surprise, some of the worst classifiers in terms of F1 are, in fact, the best ones in terms of balanced accuracy. In Table 2, we can see a selection of the previously evaluated models. As it becomes clear from the table, although the classical machine learning models are indeed the overall winners in terms of F1-score, they fall behind in terms of balanced accuracy.

6 Discussion

Hate speech has always been a problem in society. The internet revolution reinforced the problem by providing instant connectivity across social media and anonymity. There also exists mounting evidence of a connection between hate speech online and hate crime. All of this has led to increased attention towards hate speech not only from the general public, but also from governments and private organizations.

Because of its online nature, and hence the amount of data that is being constantly generated, hate speech lends itself very well to automatic detection by an artificial intelligence model. To do this, however, large and robust datasets are required, and although they do exist, most of them are focused on languages with a strong internet presence such as English. As a result, many of the not so well represented languages—such as Bulgarian—are mostly ignored.

Multiple reports have shown that hate speech is an even greater problem in Bulgaria than in other countries. For this reason, the scientific community in Bulgaria should follow in the footsteps of such communities in other countries and focus on the issue. A first step in doing that is to create datasets that can be used for training purposes of future research. We believe that by sharing our dataset consisting of 108,142 manually annotated sentences, we can contribute making that first step.

A further contribution of this paper is the evaluation of a variety of machine learning methods for the task of text classification in Bulgarian in an imbalanced setting, including some state-of-the-art approaches.

7 Future Work

Despite the continuing efforts of the scientific community, there are some fundamental issues with solving hate speech classification. For example, in (Arango et al., 2019), the authors argue that researchers have become overly optimistic about the results of their classifiers. That is because most research papers focus only on datasets coming from one source. That causes the models (usually deep neural networks) to overfit and are rarely able to generalize well on new datasets. Therefore, the creation of multiple datasets is compulsory for the development of a robust predictive model.

Another issue is annotator bias. In (Waseem, 2016), Waseem discusses how much the influence of annotators on the performance of classifiers and suggests that systems trained on data labeled by experts perform better than those labeled by amateurs. That leads us to another fundamental issue with hate speech classification: who defines what hate speech is? To mitigate any annotator bias future datasets should not only be annotated by experts but also, if possible, by different people.

Another important point that could be addressed in the context of imbalanced text classification is data augmentation (DA). Data augmentation is the process of creating artificial data to improve the performance of a classifier. One can argue that some aspects of data augmentation are already included by incorporating SMOTE into the preprocessing pipeline; however, SMOTE works on the feature space of the vectorized textual data. What might greatly impact the classifier's performance would be to augment the data at the textual level. There are multiple ways of performing this, from using a thesaurus to substitute words on a synonym level to using model embeddings (for example, Word2Vec) to sample neighboring words. This approach has been shown to increase the performance of hate speech classifiers. (Rizos et al., 2019; Bayer et al., 2021)

A further unexplored method to improve the classifiers' performance is employing additional feature engineering methods such as named entity recognition and part-of-speech taggers. These methods would enrich the feature space and result in a better classifier.

Lastly, if the context in which a model is to be used is social media, a further feature engineering idea would be to take user metadata such as age, location, or gender into account. Furthermore,

a "hate score" could be calculated for each user based on her or his past posts or her/his connections' past posts by utilizing social network analysis techniques.

Acknowledgments

The authors are grateful to Marina Kuzmanova from BG-Mamma for providing the biggest portion of the dataset. The labeled dataset of this article can be found here: http://www.pfeffer.at/data/bulgarian/.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649.

Aymé Arango, Jorge Pérez, and Barbara Poblete. 2019. Hate speech detection is not as easy as you may think: A closer look at model validation. In *Proceedings of the 42nd international acm sigir conference on research and development in information retrieval*, pages 45–54.

Dražen Bajer, Bruno Zonć, Mario Dudjak, and Goran Martinović. 2019. Performance analysis of smote-based oversampling techniques when dealing with data imbalance. In *2019 International Conference on Systems, Signals and Image Processing (IWSSIP)*, pages 265–271. IEEE.

Markus Bayer, Marc-André Kaufhold, and Christian Reuter. 2021. A survey on data augmentation for text classification. *arXiv preprint arXiv:2107.03158*.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2000. A neural probabilistic language model. *Advances in Neural Information Processing Systems*, 13.

Leo Breiman. 2001. Random forests. *Machine learning*, 45(1):5–32.

Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. 2017. *Classification and regression trees*. Routledge.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

François Chollet et al. 2015. Keras. https://keras.io.

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

Yoan Dinkov, Ivan Koychev, and Preslav Nakov. 2019. Detecting toxicity in news articles: Application to bulgarian. *arXiv preprint arXiv:1908.09785*.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic bert sentence embedding. *arXiv preprint arXiv:2007.01852*.

Nikou Günnemann and Jürgen Pfeffer. 2017. Predicting defective engines using convolutional neural networks on temporal vibration signals. In *Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications*, volume 74 of *Proceedings of Machine Learning Research*, pages 92–102. PMLR.

Grigor Iliev, Nadezhda Borisova, Elena Karashtranova, and Dafina Kostadinova. 2015. A publicly available cross-platform lemmatizer for bulgarian.

Ivanka Ivanova. 2018. Public attitudes to hate speech in bulgaria in 2018. Technical report, Open Society Institute Sofia.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Borislav Kapukaranov and Preslav Nakov. 2015. Fine-grained sentiment analysis for movie reviews in bulgarian. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 266–274.

Svetla Koeva, Nikola Obreshkov, and Martin Yalamov. 2020. Natural language processing pipeline to annotate bulgarian legislative documents. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 6988–6994.

David Lazer, Alex Pentland, Lada Adamic, Sinan Aral, Albert-László Barabási, Devon Brewer, Nicholas Christakis, Noshir Contractor, James Fowler, Myron Gutmann, Tony Jebara, Gary King, Michael Macy, Deb Roy, and Marshall Van Alstyne. 2009. Computational social science. *Science*, 323(5915):721–723.

Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. 2017. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *J. Mach. Learn. Res.*, 18(1):559–563.

Nikola Ljubešić, Tomaž Erjavec, and Darja Fišer. 2018. Datasets of Slovene and Croatian moderated news comments. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 124–131, Brussels, Belgium. Association for Computational Linguistics.

Denitza Lozanova, Sevdalina Voynova, Snezhina Gabova, and Svetlana Lomeva. 2017. Mapping out the national context of online hate speech in bulgaria. Technical report, Coalition of Positive Messengers to Counter Online Hate Speech.

Iva Marinova. 2019. Evaluation of stacked embeddings for Bulgarian on the downstream tasks POS and NERC. In *Proceedings of the Student Research Workshop Associated with RANLP 2019*, pages 48–54, Varna, Bulgaria. INCOMA Ltd.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mr Mohiyaddeen and Sifatullah Siddiqi. 2021. Automatic hate speech detection: A literature review. *Available at SSRN 3887383*.

Angelina Mooseder, Momin M. Malik, Hemank Lamba, Earth Erowid, Sylvia Thyssen, and Jürgen Pfeffer. 2022. Glowing experience or bad trip? A quantitative analysis of user reported drug experiences on erowid.org. In *Proceedings of ICWSM 2022*.

Karsten Müller and Carlo Schwarz. 2021. Fanning the flames of hate: Social media and hate crime. *Journal of the European Economic Association*, 19(4):2131–2167.

Preslav Nakov. 1998. Bulstem: Design and evaluation of inflectional stemmer for bulgarian.

Erida Nurce, Jorgel Keci, and Leon Derczynski. 2021. Detecting abusive albanian. *CoRR*, abs/2107.13592.

Alexis Okeowo. 2017. Hate on the rise after trump's election. *The New Yorker*.

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.

John Platt et al. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74.

Georgios Rizos, Konstantin Hemker, and Björn Schuller. 2019. Augment to prevent: short-text data augmentation in deep learning for hate-speech classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 991–1000.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge.

Ravi Shekhar, Pranjić. Marko, Senja Pollak, Andraž Pelicon, and Matthew Purver. 2020. Automating News Comment Moderation with Limited Resources: Benchmarking in Croatian and Estonian. *Journal for Language Technology and Computational Linguistics*, 34(1):49–79. https://jlcl.org/content/2-allissues/1-heft1-2020/ jlcl_2020-1_3.pdf.

Alice Tontodimamma, Eugenia Nissi, Annalina Sarra, and Lara Fontanella. 2021. Thirty years of research into hate speech: topics of interest and their evolution. *Scientometrics*, 126(1):157–179.

Alan Travis. 2017. Anti-muslim hate crime surges after manchester and london bridge attacks. *The Guardian*.

Zeerak Waseem. 2016. Are you a racist or am I seeing things? annotator influence on hate speech detection on Twitter. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pages 138–142, Austin, Texas. Association for Computational Linguistics.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Valentin Zhikov, Ivelina Nikolova, Laura Toloşi, Yavor Ivanov, Borislav Popov, and Georgi Georgiev. 2012. Enhancing social news media in bulgarian with natural language processing. *INFOtheca*, 2(13):6–18.