UNIVERSIDADE FEDERAL DE GOIÁS (UFG)
INSTITUTO DE INFORMÁTICA (INF)
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO (PPGCC)

RODOLFO COSTA CEZAR DA SILVA

A Comparative Study of Text Classification Techniques for Hate Speech Detection

Goiânia
2022

UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO (TECA) PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TESES

E DISSERTAÇÕES NA BIBLIOTECA DIGITAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio da Biblioteca Digital de Teses e Dissertações (BDTD/UFG), regulamentada pela Resolução CEPEC nº 832/2007, sem ressarcimento dos direitos autorais, de acordo com a Lei 9.610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou download, a título de divulgação da produção científica brasileira, a partir desta data.

O conteúdo das Teses e Dissertações disponibilizado na BDTD/UFG é de responsabilidade exclusiva do autor. Ao encaminhar o produto final, o autor(a) e o(a) orientador(a) firmam o compromisso de que o trabalho não contém nenhuma violação de quaisquer direitos autorais ou outro direito de terceiros.

1. Identificação do material bibliográfico

[X] Dissertação [ ] Tese

2. Nome completo do autor

Rodolfo Costa Cezar da Silva

3. Título do trabalho

A Comparative Study of Text Classification Techniques for Hate Speech Detection

4. Informações de acesso ao documento (este campo deve ser preenchido pelo orientador)

Concorda com a liberação total do documento [ ] SIM [X] NÃO[1]

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. Após esse período, a possível disponibilização ocorrerá apenas mediante:
a) consulta ao(à) autor(a) e ao(à) orientador(a);
b) novo Termo de Ciência e de Autorização (TECA) assinado e inserido no arquivo da tese ou dissertação.
O documento não será disponibilizado durante o período de embargo.
Casos de embargo:
- Solicitação de registro de patente;
- Submissão de artigo em revista científica;
- Publicação como capítulo de livro;
- Publicação da dissertação/tese em livro.

Obs. Este termo deverá ser assinado no SEI pelo orientador e pelo autor.

Documento assinado eletronicamente por **Thierson Couto Rosa**, **Professor do Magistério Superior**, em 03/02/2022, às 15:20, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **RODOLFO COSTA CÉZAR DA SILVA**, **Discente**, em 03/02/2022, às 16:18, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **2670955** e o código CRC **573AE4AD**.

Referência: Processo nº 23070.070037/2021-18 SEI nº 2670955

UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO (TECA) PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TESES

E DISSERTAÇÕES NA BIBLIOTECA DIGITAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio da Biblioteca Digital de Teses e Dissertações (BDTD/UFG), regulamentada pela Resolução CEPEC nº 832/2007, sem ressarcimento dos direitos autorais, de acordo com a Lei 9.610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou download, a título de divulgação da produção científica brasileira, a partir desta data.

O conteúdo das Teses e Dissertações disponibilizado na BDTD/UFG é de responsabilidade exclusiva do autor. Ao encaminhar o produto final, o autor(a) e o(a) orientador(a) firmam o compromisso de que o trabalho não contém nenhuma violação de quaisquer direitos autorais ou outro direito de terceiros.

1. Identificação do material bibliográfico

[X] Dissertação [ ] Tese [ ] Outro*:_____

*No caso de mestrado/doutorado profissional, indique o formato do Trabalho de Conclusão de Curso, permitido no documento de área, correspondente ao programa de pós-graduação, orientado pela legislação vigente da CAPES.

Exemplos: Estudo de caso ou Revisão sistemática ou outros formatos.

2. Nome completo do autor

Rodolfo Costa Cezar da Silva

3. Título do trabalho

A Comparative Study of Text Classification Techniques for Hate Speech Detection

4. Informações de acesso ao documento (este campo deve ser preenchido pelo orientador)

Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. Após esse período, a possível disponibilização ocorrerá apenas mediante:
a) consulta ao(à) autor(a) e ao(à) orientador(a);
b) novo Termo de Ciência e de Autorização (TECA) assinado e inserido no arquivo da tese ou dissertação.
O documento não será disponibilizado durante o período de embargo.
Casos de embargo:
- Solicitação de registro de patente;
- Submissão de artigo em revista científica;
- Publicação como capítulo de livro;
- Publicação da dissertação/tese em livro.

Obs. Este termo deverá ser assinado no SEI pelo orientador e pelo autor.

Documento assinado eletronicamente por **Rodolfo Costa Cezar Da Silva**, **Usuário Externo**, em 22/02/2024, às 10:24, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020 .

RODOLFO COSTA CEZAR DA SILVA

A comparative study of text classification techniques for hate speech detection

Dissertação apresentada ao Programa
de Pós-Graduação em
Ciência da Computação, do Instituto de Informática,
da Universidade Federal de Goiás (UFG),
como requisito para obtenção do
título de Mestre em Ciência da Computação.
Área de concentração : Ciência da Computação
Linha de pesquisa : Sistemas Inteligentes e Aplicações

Orientador : Professor Doutor Thierson Couto Rosa

GOIÂNIA
2022

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

Silva, Rodolfo Costa Cezar da
A comparative study of text classification techniques for hate speech detection [manuscrito] / Rodolfo Costa Cezar da Silva. - 2022.
LXXII, 72 f.: il.

Orientador: Prof. Dr. Thierson Couto Rosa.
Dissertação (Mestrado) - Universidade Federal de Goiás, Instituto de Informática (INF), Programa de Pós-Graduação em Ciência da Computação, Goiânia, 2022.
Bibliografia. Apêndice.
Inclui abreviaturas, gráfico, tabelas, lista de figuras, lista de tabelas.

1. Classificação de texto. 2. Desbalanceamento de classes. 3. Detecção de discurso de ódio. 4. Aprendizado de Máquina. I. Rosa, Thierson Couto, orient. II. Título.

CDU 004

UNIVERSIDADE FEDERAL DE GOIÁS

INSTITUTO DE INFORMÁTICA

ATA DE DEFESA DE DISSERTAÇÃO

Ata nº **01/2022** da sessão de Defesa de Dissertação de **Rodolfo Costa Cezar da Silva**, que confere o título de Mestre em Ciência da Computação, na área de concentração em Ciência da Computação.

Aos vinte e sete dias do mês de janeiro de dois mil e vinte e dois, a partir das quinze horas, via sistema de webconferência da RNP, realizou-se a sessão pública de Defesa de Dissertação intitulada **" A Comparative Study of Text Classification Techniques for Hate Speech Detection"**. Os trabalhos foram instalados pelo Orientador, Professor Doutor Thierson Couto Rosa (INF/UFG) com a participação dos demais membros da Banca Examinadora: Professor Doutor Edleno Silva de Moura (ICOMP/UFAM), membro titular externo; Professora Doutora Nádia Félix Felipe da Silva (INF/UFG), membra titular interna. A realização da banca ocorreu por meio de videoconferência, em atendimento à recomendação de suspensão das atividades presenciais na UFG emitida pelo Comitê UFG para o Gerenciamento da Crise COVID-19, bem como à recomendação de isolamento social da Organização Mundial de Saúde e do Ministério da Saúde para enfrentamento da emergência de saúde pública decorrente do novo coronavírus. Durante a arguição os membros da banca não fizeram sugestão de alteração do título do trabalho. A Banca Examinadora reuniu-se em sessão secreta a fim de concluir o julgamento da Dissertação, tendo sido o candidato **aprovado** pelos seus membros. Proclamados os resultados pelo Professor Doutor Thierson Couto Rosa, Presidente da Banca Examinadora, foram encerrados os trabalhos e, para constar, lavrou-se a presente ata que é assinada pelos Membros da Banca Examinadora, vinte e sete dias do mês de janeiro de dois mil e vinte e dois.

TÍTULO SUGERIDO PELA BANCA

Documento assinado eletronicamente por **Edleno Silva de Moura**, **Usuário Externo**, em 27/01/2022, às 16:58, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Thierson Couto Rosa**, **Professor do Magistério Superior**, em 27/01/2022, às 16:58, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Nadia Felix Felipe Da Silva**, **Professor do Magistério Superior**, em 27/01/2022, às 16:58, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **RODOLFO COSTA CÉZAR DA SILVA**, **Discente**, em 27/01/2022, às 22:11, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?

acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **2624038** e o código CRC **4C0F5A2B**.

Referência: Processo nº 23070.070037/2021-18 SEI nº 2624038

A toda minha família. Essa conquista é, em parte, de todos nós.

Agradecimentos

Em um período marcado por uma crise sanitária e humanitária, parece não haver muito a ser escrito em uma seção de agradecimentos. Porém concluir o mestrado em meio a tantas incertezas, tem um fator especial. Apesar dos desânimos e tristezas, a pandemia também trouxe esperança e evidenciou a importância da ciência ao redor do mundo.

Entre distanciamentos físicos, testagens virais, vacinas, enfermidades, incerteza de como continuar, experimentos falhos, escrita, reescrita e correções de capítulos e seções, dúvidas e certezas, eu aprendi muita coisa além da minha área de pesquisa. Encontrei respostas que foram muito além das perguntas de pesquisa da minha dissertação de mestrado. E por isso eu sou grato.

Além de ser grato pela jornada em si, é preciso agradecer pessoas que a tornaram possível e a deixaram mais leve.

Aos meus pais, Vilson e Maisa, pelo apoio incondicional durante todo o processo.

Agradecimento à minha companheira Gabriella que sem dúvida, foi minha inspiração e motivação durante todo os anos em que estamos juntos, e que durante períodos de incerteza se fez mais presente e querida.

À toda a minha família e amigos, que me incentivaram a trilhar o caminho da academia, mesmo quando o caminho parecia incerto.

À CAPES pelo apoio financeiro através da bolsa de estudos, e ao Governo Federal por proporcionar um programa de pós-graduação de altíssima qualidade, público e gratuito.

E por fim, um agradecimento especial ao Prof. Dr. Thierson Couto Rosa, que além de orientador, se tornou um modelo de professor e pesquisador, marcado pela dedicação, atenção e conhecimento. Durante esses dois anos, compartilhamos inúmeros momentos de conhecimento que eu, sem dúvidas, levarei para o resto da minha vida.

"As oportunidades multiplicam-se à medida que são agarradas."

Sun Tzu,

.

Resumo

Da Silva, Rodolfo. **A comparative study of text classification techniques for hate speech detection**. Goiânia, 2022. 73p. Dissertação de Mestrado. Instituto de Informática, Universidade Federal de Goiás.

A disseminação do discurso de ódio na Internet, especialmente nas plataformas de redes sociais, tem sido um problema recorrente. No presente estudo, comparamos onze métodos de classificação para discurso de ódio, incluindo métodos tradicionais de aprendizado de máquina, abordagens baseadas em redes neurais e *Transformers*, assim como a combinação com oito técnicas para resolver o problema de desbalanceamento de classes, uma característica inerente à classificação de discurso de ódio.As técnicas de transformação de dados que investigamos incluem técnicas de reamostragem de dados e uma modificação de uma técnica baseada em *features* compostas (c_features). Todos os modelos foram testados em sete coleções de dados com especificidades variadas, seguindo um rigoroso protocolo de experimentação que inclui validação cruzada e o uso de métricas apropriadas, bem como a validação dos resultados por meio de testes estatísticos apropriados para comparações múltiplas. Até onde sabemos, não há estudo comparativo mais amplo em técnicas de expansão de dados para detecção de discurso de ódio, nem qualquer trabalho que combine técnicas de reamostragem de dados com *Transformers*. Nossa extensa experimentação, baseada em mais de 2.900 medições, revela que a maioria das técnicas de reamostragem de dados são ineficazes para aumentar a eficácia dos classificadores, com exceção da técnica de *Random Oversampling* (ROS) que melhora a maioria dos métodos de classificação, incluindo os *Transformers*. Para a menor coleção de dados, ROS proporcionou ganhos de 60,43% e 33,47% para BERT e RoBERTa, respectivamente. Os experimentos revelaram que a técnica de *c_features* melhorou todos os métodos de classificação com os quais ele pôde ser combinado. A técnica de *features* compostas proporcionou ganhos satisfatórios de até 7,8% para SVM. Finalmente, investigamos a relação custo-efetividade de alguns dos melhores métodos de classificação. Essa análise confirmou que o método tradicional de Regressão Logística (LR) combinado com o uso de *c_features* proporciona grande eficácia com baixo *overhead* em todas as coleções de dados consideradas.

Palavras–chave

Aprendizado de Máquina, discurso de ódio, classificação de textos, desbalanceamento de classes

Abstract

Da Silva, Rodolfo. **A comparative study of text classification techniques for hate speech detection**. Goiânia, 2022. 73p. MSc. Dissertation. Instituto de Informática, Universidade Federal de Goiás.

The dissemination of hate speech on the Internet, specially on social media platforms, has been a serious and recurrent problem. In the present study, we compare eleven methods for classifying hate speech, including traditional machine learning methods, neural network-based approaches and transformers, as well as their combination with eight techniques to address the class imbalance problem, which is a recurrent issue in hate speech classification. The data transformation techniques we investigated include data resampling techniques and a modification of a technique based on compound features (c_features). All models have been tested on seven datasets with varying specificity, following a rigorous experimentation protocol that includes cross-validation and the use of appropriate evaluation metrics, as well as validation of the results through appropriate statistical tests for multiple comparisons. To our knowledge, there is no broader comparative study in data enhancing techniques for hate speech detection, nor any work that combine data resampling techniques with transformers. Our extensive experimentation, based on over 2,900 measurements, reveal that most data resampling techniques are ineffective to enhance the effectiveness of classifiers, with the exception of ROS which improves most classification methods, including the transformers. For the smallest dataset, ROS provided gains of 60.43% and 33.47% for BERT and RoBERTa, respectively. The experiments revealed that c_features improved all classification methods that they could be combined with. The compound features technique provided satisfactory gains of up to 7.8% for SVM. Finally, we investigate cost-effectiveness for a few of the best classification methods. This analysis provided confirmation that the traditional method Logistic Regression (LR) combined with the use of c_features can provide great effectiveness with low overhead in all datasets considered.

Keywords

Machine Learning, hate speech, text classification, class imbalance, data resampling

Sumário

Lista de Tabelas

CAPÍTULO 1

Introduction

Hate speech is characterized by any type of message that attacks or degrades people based on their ethnic origin, religion, physical or mental disability, gender, age or sexual orientation [39]. The spread of hate speech has been a recurrent problem in social media platforms like Facebook and Twitter. Thus it is mandatory that hate messages be detected as soon as possible so that their broadcast can be interrupted.

Given the difficulty in dealing with this problem manually due to the volume and speed that messages are exchanged in the Internet, it becomes necessary to use computational techniques to automatically classify messages containing hate speech. Machine Learning algorithms are among the most used computational techniques to deal with problems of this nature, because, although they are generally less accurate than manual solutions made by experts, they require little human intervention, allow continuous learning and are generally scalable to the exorbitant volume of data on the Internet.

Hate speech detection has been treated in the literature as an automatic document classification problem. In this way, a learning method is applied over a set of training documents (web page, message, tweet, post, etc). Each document in this set has a label (or class) manually assigned by human annotators that indicates whether the document contains hate speech or not. The learning method aims to "learn" the characteristics of the document that make it correspond to a specific label. The resulting product of the application of the learning method on the training set is a function (a program) called *classifier*. Once generated, the classifier is able to assign a label to documents outside the training set.

The study of hate speech detection from a computer science point of view is recent [19]. Most work in this subject are concentrated in the the second half of last decade. At that time many works were dedicated to propose labeled datasets to evaluate hate speech detection techniques [20, 11, 12, 54, 13, 25, 14, 46, 51]. Thus, past research articles usually use very few datasets for comparing text classification methods for hate speech detection [45, 1, 30, 9, 4, 34]. Also, most of the past works did not have the intention to make an extensive comparison of distinct methods.

Recent advances in text classification, notably the transformer models have been used/adapted for hate speech detection [30, 36, 37]. Most of these works compare their proposals which use transformers against other Deep Neural Networks (DNN) architectures, but do not compare with traditional learning methods, mainly in terms of cost-effectiveness trade-off. This is an important aspect because not only transformers but Deep Learning techniques in general require more sophisticate hardware like GPUs, besides having longer training time than traditional learning methods like Support Vector Machines (SVM), Logistic Regression (LR), Naive Bayes (NB) or Random Forest (RF). Also, there is a lack of comparisons among all the recent Deep Learning methods and traditional ones over multiple datasets to confirm the superiority of these new classification methods over classical ones.

Comparisons among classification methods for hate speech detection also have been affected by inappropriate experimental methodology. For instance, recent published solutions which were considered to be state-of-the-art were, afterwards, found to present important experimental flaws [5]. Thus, it is important that a comparative study of different learning methods (including traditional and recent techniques) on various datasets, preferably with distinct characteristics, makes use of appropriate experimental methodology, which includes experimenting all methods in the same train-test split of the datasets, the use of cross-validation, appropriate evaluation metrics, and statistical tests.

Another aspect that has been neglected when comparing hate speech detection methods is the *class imbalance* problem. It is expected that the number of messages containing hate speech to be smaller than that of non-hate speech messages in social media or other web sites. Thus, datasets sampled from these environments tend to present similar imbalanced distribution. Training a classifier under this class imbalanced condition leads to the generation of classifiers that exhibit bias towards the majority class, presenting difficulties in correctly classifying instances of the minority class. Class imbalance is a problem to automatic classification in general (not only text classification). Many techniques have been proposed to help learning methods to cope with this problem, notably the use of resampling techniques [47, 24, 50, 31, 56] for balancing the class distribution in the training set. However we are not aware of any work that studied the effects of combining resampling techniques with a large set of learning methods, specially transformers, for the hate speech problem.

Research Objectives

The main objective of this work is three-fold: *i*) To make a comparative study of classifiers trained with distinct learning methods on multiple datasets; *ii*) To evaluate how distinct resampling techniques affect the effectiveness of classifiers trained with

distinct classification methods; *iii*) To investigate how an adaptation of the technique of Compound Features (aka c_features) [18] acts on the effectiveness of classifiers generated by the classification methods that can be combined with it. We refine the main research objective by means of six research questions, stated as follows:

- **RQ1** - *How effective are the different classification methods when applied to multiple datasets about hate speech?* – The objective of RQ1 is to compare the effectiveness of classifiers generated by diverse learning methods using distinct data representations over multiple datasets.
- **RQ2** - *Do the data resampling techniques improve the classification methods?* – With RQ2 we want to investigate if the main resampling techniques used for data balancing are beneficial for the learning methods considered, i.e., if they allow for each method to generate more effective classifiers.
- **RQ3** - *Do the c_features enhance classification methods for hate speech detection?* – With RQ3, we investigate if our proposed adaptation to the feature expansion technique known as c_features allows for generating more effective classifiers.
- **RQ4** - *How do the best combinations of classification methods and data transformations compare to each other?* – When answering RQ2 and RQ3 we make pairwise comparisons between two classifiers generated by a same learning method *m*: one trained in the original training set and the other trained in a version of the training set transformed by a resampling technique or by the expansion with c_features. We keep the winners of these pairwise comparisons. With RQ4, we want to compare among these winners. In other words, we want to compare the best versions of the classifiers trained with each learning method, be these versions generated over the original data or over the transformed ones.
- **RQ5** - *Does the combination of c_features and resampling enhance effectiveness?* – The objective of RQ5 is to investigate if the c_features expansion combined with the best resampling techniques can produce more effective classifiers.
- **RQ6** - *How do the classification methods compare to each other in terms of cost-effectiveness trade-off?* – The objective of RQ6 is to analyze the trade-off between the training time (cost) and the effectiveness of the classifiers generated by the best learning methods found when answering the previous research questions.

Contributions of our study

To answer the above research questions, we performed an extensive comparative study of classifiers trained with distinct learning methods over multiple datasets about hate speech detection. Specifically, we compared 11 learning methods, ranging from traditional methods using the Vector Space Model (VSM) to Deep Neural network approaches

including Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (BiLSTM) and fine-tuned state-of-the-art (SOTA) transformer methods like BERT [16] and RoBERTa [32]. Classifiers generated with these learning methods were evaluated over seven cross-validated datasets, using appropriate experimental methodology (we follow the methodology proposed by Demšar [15] and adapted by Benavoli [7]).

Additionally, those classification methods were combined, whenever possible, with eight different data transformation techniques (resampling and c_features). All those combinations add up to 2,905 experiments[1]. To our knowledge, there is no previous comparative study in data enhancing techniques for hate speech detection so large as the one presented here. Also, we consider this is the first study which takes into consideration the cost-effectiveness trade-off of classification methods for the hate speech detection problem.

Our extensive experimental evaluation allowed us to obtain answers and interesting information regarding the research questions as we describe in the following paragraphs.

RQ1 When no data transformation is used, classification performed by BERT is the great winner. BERT generates the best classifiers for five out of the seven datasets. Followed by Logistic Regression (LR), Linear Support Vector Machine (SVM) and RoBERTa, in this order. However, the transformer methods were shown to be unstable in the sense that in the datasets where they perform badly, the difference in Macro-F1 in relation to LR and SVM is much greater than the difference in the datasets where they win. For instance, BERT is superior to LR in at most 7.9% in one dataset, while LR is better than BERT in 38% and in 37% in the the two datasets where BERT presents inferior performance. We also find, surprisingly, that traditional methods (LR, SVM, Naive Bayes (NB), Random Forest (RF) and Multilayer Perceptron (MLP)) surpass many Deep Learning methods, including LSTM [6] and BiLSTM [2] that were considered until recently state-of-the-art in hate speech detection.

RQ2 Almost all of the sophisticated and most used oversampling techniques (SMOTE [10], Bordeline-SMOTE [26], SVM-SMOTE [38], K-Means-SMOTE [17], ADASYN [28]) did not help enhance classifiers trained with the classification methods that could be used in conjunction to these techniques. There are few exceptions, but even in cases where these oversampling techniques help they are not sufficient to enhance the classification method considerably. However, the simple Random Oversampling (ROS)

[1] 11 classification methods multiplied by 8 resampling techniques, minus 12 of these pairs ⟨ classification method, resampling technique ⟩ which are not applicable, plus 7 classification methods using c_features, all these multiplied by seven datasets using 5-fold cross-validation.

technique benefited many classification methods. Specially, ROS was able to improve BERT and RoBERTa by 60.43% and 33.47%, respectively in the smallest dataset. Also ROS did not harm both methods with statistical significance in any of the other datasets.

RQ3 Our proposed modification of the c_features computation was shown to enhance the effectiveness of classifiers generated by all methods it could be used with, i.e., methods based on the Vector Space Model - VSM. Specifically, c_features benefited the SVM and LR with gains up to 7.8% and 6.1%, respectively, when compared to versions of these methods trained on the original data. Also, a combination of a properly tuned SVM with our modification of c_features proved to be the best overall combination for a class imbalanced hate speech detection problem, ranking higher than SOTA models like BERT [16], RoBERTa [32] and other neural network approaches. For instance, SVM with c_features outperformed the state-of-the-art BERT on the smallest and largest datasets by 45% and 43%, respectively.

RQ4 When analyzing both mean and median rank positions of the the analyzed classifiers over the seven datasets, we noticed that the best classifiers (smallest mean/median) are those generated with either c_features or ROS, showing that these data transformation techniques are consistently good choices. Most of the classification methods that are paired with this top data enhancing techniques are traditional methods, like SVM or Logistic Regression (LR), also proving that traditional methods are still among the top classification methods, outperforming sophisticated neural network approaches, while still proving to be more stable. If we consider the mean of the rank positions in all datasets, SVM classifiers using c_features is the best, followed by BERT with ROS, LR with c_features and RoBERTa with ROS.

RQ5 Despite ROS and c_features have enhanced individually many classification methods, when combining them together we found no significant additional gain for any classification method. The combination was not able to surpass results obtained using c_features only. Similar conclusions were found by combining c_features with undersampling.

RQ6 It is difficult to compare the training time of the distinct classification methods we considered in this study, because traditional methods can be executed entirely in CPU, while Deep Neural also require GPUs. However we considered in our study the complete clock time to train each method to have the notion of cost in terms of training time to generate the classifiers. Thus, our cost-effectiveness analysis is in terms of training time versus effectiveness. Our analysis were made in terms of the median of these two

dimensions. Thus, the best methods were those with training time inferior to the median training time and with effectiveness superior to the median effectiveness. We found that the great cost-effective method is LR using c_features, being the best method in all datasets. In fact LR training time is not much affected by any data transformation. On the other hand, the training time of transformers and SVM are affected considerably, almost doubling in some datasets. The answer of RQ6 is of practical importance because middle-sized companies or universities do not always have GPUs available and retraining is important specially in hate speech context, thus the cost to train a hate speech classifier can not be neglected in some environments.

Dissertation organization

The rest of the dissertation is organized as follows: Chapter 2 provides previous work related to hate speech detection as well as the class imbalance problem. Chapter 3 details the data resampling techniques analyzed in this work. Chapter 4 presents c_features as well as our modifications to this data expansion method. Chapter 5 presents the experimental setup, including: description of the datasets, classification methods covered, as well as metrics and model evaluation. The results are presented in Chapter 6. Finally, Chapter 7 concludes our study and provides some suggestions for future investigations.

CAPÍTULO 2

Related Work

The automatic hate speech detection field is relatively recent [19], and started to gain some notoriety around 2016, where the number of published works began to grow. Also according to the survey in [19], among the works described, only a small portion address Machine Learning algorithms specific to the problem of automatic classification of hate speech, and even a smaller portion are comparative works for hate speech classification.

The authors in [20] proposed the use of a LSTM for the hate speech classification problem. The method proposed achieve better F-score (0.78) than Multilayer Perceptron (MLP), XgBoost and SVM proposed in previous work [21]. Despite the authors have used cross-validation as a part of model evaluation, they used micro averaged F1-score as an evaluation metric, which is unrecommended for a class imbalanced dataset. The authors evaluate the models using only one dataset and do not validate results using statistical analysis.

The main contribution of [12] is a hate speech labeled dataset with 25k messages crawled on Twitter. The dataset is divided into three categories : hate speech, offensive language but not hate speech and not hate speech. This newly created dataset was validated by the authors using several classification methods including : Linear SVM, Decision Trees (DT), Logistic Regression (LR) and Random Forest (RF). The authors evaluated the methods using micro averaged precision, recall and F1-score. They also used the average results of the five fold cross-validation. The best performing method on all metrics was the Linear SVM. Despite the fact that the main objective of this paper was not to present the best classifiers for hate speech classification, but rather validate their newly created dataset. Nevertheless, the authors employed micro averaged metrics which may not reflect the model's perfomance in a class imbalanced dataset. Also, the authors did not use different type of classification algorithms, like Deep Neural Networks (DNN) and transformers.

The authors in [55] proposed a multi-class dataset containing about 16k divided into three classes : sexist (3,383 messages), racist (1,972 messages) and neutral (11,559 messages). They also investigate the influence of different features on the classification

algorithm, like size of messages, unigrams, bigrams and trigrams. The highly imbalanced dataset was validated using a Logistic Regression (LR) model using grid search in order to obtain the optimal combination of features. The authors only used a single classification method, LR, which achieved a F-score of 0.7693. The authors use the average measure of the 10-fold cross-validation and compare the obtained results using different combination of classifiers (LR with different types of features) with a paired t-test. Given that the main contribution of the work is the labeled dataset, the authors could have employed different classification methods, like Neural Network techniques and other traditional classification methods, to improve the quality of the resulting dataset.

A labeled hate speech dataset was proposed by Gibert et. al [13]. The dataset, containing about 10k instances, is formed by messages retrieved from a white supremacy forum (Stormfront). The binary dataset has 85% of the messages that do not contain hate speech, while only 15% contains hate speech. In order to inspect and validate the labeling process, the authors used three classification methods : SVM, Convolutional Neural Network (CNN) and Long Short-term Memory (LSTM). The classifiers were evaluated using the accuracy for each class separately as well as the overall accuracy. The best results were obtained by LSTM and SVM (0.76 and 0.72, respectively) using the accuracy for the hate speech class, given that it is the class of interest. Analyzing overall accuracy in a class imbalanced dataset is unadvised, since it may lead to overly optimistic evaluation of the classifier. Instead of using appropriate metrics that are well known in the literature, the authors use a variation of an overly-optimistic metric for a class imbalanced scenario. The authors used three different types of classification methods to validate the dataset. Also, in the model evaluation process, the authors did not employ cross-validation, in order to assess for the model's ability to generalize.

In a brief survey, MacAvaney et. al [33] bring up the challenges and solutions in hate speech detection, while proposing a new classification method entitled Multi-View SVM. The general ideal of the method is to stack several linear kernel SVMs. Each of the stacked SVM is a classifier trained on data that is coded with different document representation. To evaluate the proposed method, the authors use macro-averaged F1-score and experimented on four different hate speech datasets that are know for their class imbalance. The proposed method was compared with BERT and a technique the authors called "neural ensemble". The Multi-View SVM presented competitive results, outperforming BERT in one of the datasets, but presenting inferior Macro-F1 values on the other three. The authors did not thoroughly report details about the architecture and parameters of the methods that were compared with Multi-View SVM. In regards to model evaluation, the authors used appropriate metrics for class imbalanced datasets. On the other hand, they did not use cross-validation. The authors compared their proposed methods with only two types of classification methods, rather than expanding the range

and using other type of traditional and neural network classification methods.

In recent work [43] the authors proposed a model to detect offensive language on Twitter. The model is an ensemble of Long Short-Term Memory (LSTM). Other than the feature representation of the documents, the authors proposed three additional features. The novel features represent the tendency of the user towards offensive language. The three features pertain to the three classes of the analyzed dataset, proposed in [55]. The tendency represents the proportionality that a certain user publishes messages belonging to one of the classes. Each tendency feature can be added or removed from the feature set. The model is evaluated using 10-fold cross-validation and through precision, recall and F1-score. The best f-score result (0.932) was achieved by the neural ensemble and by adding all three user tendencies. The proposed methods achieved satisfactory results, but the authors did not employ the methods on different datasets, nor compared the methods against other classification algorithms. Despite the idea of using users' tendencies as features is innovative, it becomes impractical because for most datasets, information about the users are removed due to privacy issues.

The main goal of the work described in [41] is an ensemble technique for offensive language detection formed by three auxiliary classifiers : HateWord2Vec, HateDoc2Vec and SVM. The former two were also proposed by the author and consist of a lexicon-based classifier and a logistic regression classifier based on comment embeddings, respectively. The ensemble uses a Naive-Bayes as a meta-classifier to make final predictions. To evaluate the model, the authors used four cross-validated datasets and the metric analyzed were F-score and ROC-AUC. The ensemble outperformed the individual classifiers with both F-score and ROC-AUC in all the analyzed datasets, achieving an average F-score of 0.93. The results were then statistically validated using the Wilcoxon test, where the null hypothesis were rejected if $p < 0.05$. This work performs a solid and robust model evaluation, with the exception of the metric choice. While ROC-AUC might be an interesting and useful metric in this context, the F-score, as calculated, is not appropriate for imbalanced dataset, as is the case for all the analyzed datasets in this work. The authors propose an new ensemble method, but only compare it to a few classification methods that form the ensemble, rather than comparing it against other traditional and neural network techniques.

Detecting white supremacist hate speech is the main objective of the work described in [4]. They tackle the problem using domain specific word embeddings with deep neural networks and BERT. The authors employed three types of word embeddings pre-trained on different corpora and a fourth trained on 1,000,000 tweets that could potentially have white supremacy content. The authors use two class imbalanced datasets and a third one that is resampled by the former two datasets in order to be a perfectly balanced dataset. The domain-specific word embeddings was then paired with BiLSTM

techniques and compared with a LSTM from previous work. Their model using domain-specific word embeddings outperformed the previous work by only 2 points in accuracy. The authors were not satisfied with the results, given that the previous work used a much simpler random initialization of word embeddings. The authors also experimented with the traditional fine-tuning of BERT base and large models. The larger BERT model outperformed the base model only on the balanced the dataset, but was outperformed on the other two imbalanced datasets. The idea of using domain specific word embeddings is promising, but it did not vastly improve classification. The authors used this technique only on BiLSTM, rather than exploring this novelty word-embedding with different neural network methods. Also, the authors did not report how the dataset was split into training, test and validation, and neither how the models were evaluated.

Burnap and Williams [9] proposed to tackle the cyber hate speech problem on Twitter. They provide a private dataset composed of 2,000 labeled tweets where only 222 of them were labeled as having offensive or antagonistic content. The authors compared the following classification methods over their proposed dataset: Bayesian Logistic Regression (BLR), Random Forest Decision Tree (RFDT), SVM, and voted ensemble that was composed by the three individual classifiers where the final prediction was made by using maximum probability. Each classifier, including the ensemble, was paired with several types of document representation which included word n-gram (1-5) as well as the combination with features based on typed dependency relationships of the words in the document. The authors evaluated the classifiers using 10-fold cross-validation and under precision, recall and F1-measure for the minority class (messages containing cyber hate) only. The proposed ensemble is tied with the individual SVM classifier as the best models, obtaining 0.77 in F-score. The authors only experimented on one dataset, so the ensemble model they propose cannot be guaranteed to have the ability to generalize predictions across other datasets.

The authors in [6] proposed the use of Deep Neural Networks (DNNs) for the hate speech detection problem in tweets. They experiment with Logistic Regression (LR), Random Forest, SVMs, Gradient Boosted Decision Trees (GBDTs). The feature selection is defined by task-specific embeddings learned using deep learning techniques like FastText, CNN and LSTM. For feature selection baselines, the authors used traditional character n-grams, TF-IDF and BOW. The authors evaluated the models using 10-fold cross-validation and calculated weighted macro precision, recall and F1-score. They achieved their best results with the model "LSTM + Random Embedding + GBDT", which means that the tweet embedding vectors were randomly initialized, LSTM was trained with back-propagation to learn the embeddings, and then they were used to train a GBDT classifier. Despite the fact that the authors performed combinations that lead to a considerable number of classifiers and used appropriate model evaluation techniques, they only

experimented on one dataset. The authors also did not compared their proposed method to transformer methods, nor performed extensive analysis about their obtained results.

In [45] the authors provide a comparative study for binary hate speech classification methods for Hindi-English mixed-coded data. Code mixing refers to when a speaker uses two languages together in single line of communication. To assess this problem, and evaluate the models, the authors use three different datasets. Two of the datasets were previously available in the literature while the third one is a part of their work contributions. The classification methods analyzed were : SVM, Multinomial Naive-Bayes (MNB), kNN, Decision Trees (DT) and a Character-level CNN (CNN). The authors evaluate the classification methods using accuracy and Micro-F1, which are unrecommended metrics, given that there is a large class imbalance in one dataset. Not only that, the authors divide the training, testing and validation into a fixed 70/20/10 division, rather than use cross validation in order to assess for better model generalization capability. In this paper, the results were not statistically validated, so the claims of superiority made by the authors regarding the CNN may be due to chance, specially in the dataset division that was used.

The authors in [1] proposed a comparative study of classification methods for the hate speech problem. In their study, they compared eight machine learning algorithms combined with three different feature extraction techniques. That adds up to 24 comparisons overall. Among the realm of classification methods are : Logistic Regression (LR), Naive-Bayes (NB), SVM, kNN, Decision Trees (DT), AdaBoost and Multilayer-Perceptron (MLP). The author evaluated the techniques using micro-averaged precision, recall and F1, and the latter is not a good choice given the huge class imbalance. Not only did the author evaluated the classification techniques on only one dataset, but they did not perform cross validation nor validated the results statistically.

Kumar et. al [30] performed a comparative study in order to understand the relationship between offensive and aggressive language in Hindi-English coded mixed data. Mixed-coded data or language when two (or more) languages are used together in a single utterance [53]. In this study the authors use two datasets in order to evaluate the classification algorithms. In this work SVM is paired with different combinations of feature extraction techniques, ranging from word n-grams (unigrams, bigrams and trigrams) to character n-grams (bigrams to 5-grams). The combination of different word and character n-grams were also employed. Since the analyzed data is mixed between two languages, the authors also experimented with multilingual transformer BERT, as well as their derivatives DistilBERT and ALBERT. These classification methods were evaluated using micro-averaged precision, recall, and f-score and cross-validation. Despite the fact that this is one of the broader comparative studies in terms of classification methods, the authors only experimented with two datasets. Also, the authors only experiment with four different classification algorithms.

Due to the lack of enough hate speech training data [36], recent work have focused on the use of transfer learning and transformers to tackle hate speech detection. The authors in [30] propose a study to investigate the relation between aggressive and offensive language in three different languages : Hindi, Bangla and English. In order to achieve that, they experimented with different classifiers, using SVM as well as classifiers based on transformers : BERT and a few derivative like DistilBERT and AlBERT. The classifiers achieved up to 0.8 in F-score. The authors used the classifiers to cross-annotate two datasets to check for co-occurrence of different categories of aggression and offense. They concluded that, even though these categories overlap, one does not necessarily entail the other. This work study aggression and offensive speech, which falls into the category of hate speech, makes the use of classification algorithms, but its objective is not to perform comparisons between classification methods.

According to the authors in [36] the lack of enough labeled training data is one of the main issues in the hate speech detection domain. Because of that, their study proposed the use of fine-tuning pre-trained language models based on BERT. Using two datasets, the authors experimented with BERT and combinations with both LSTM and CNN as the final architecture layers. The CNN-based fine tuning BERT presented the best results, outperforming previous work done with the same datasets. After model evaluation, the authors provided great error analysis. They discovered the data labeling was bias and proved that their model had a great ability in detecting that bias in its predictions. The results were obtained using fixed splits on the dataset, instead of using cross validation, not allowing to check for model generalization. The authors completely neglected the class imbalance in both datasets and did not combine the transformer methods with any data resampling techniques in order to check if it would raise model effectiveness.

Mutanga et. al [37] performed a comparative study of transformer methods applied to hate speech detection on Twitter. The study compares BERT, two different implementations of RoBERTa, XLNet and DistilBERT. Besides the transformer, the authors also included a LSTM with Attention technique, which they considered to be state-of-the-art for assignments of sequence modeling, despite having long-term dependency problems. The methods were evaluated under accuracy, recall, precision and MCC. The methods were applied to a class imbalanced multi-class dataset, and concluded that DistilBERT was the overall best method. The dataset was split into a fixed split (80/20) and the authors made assumptions about the classification methods based on metrics that are not recommended for class imbalance training set and without any statistical test validation.

Given the recent comparative work on hate speech classification, our efforts in this work aim in bridging the gap in terms of robust model evaluation and analysis, amount of classification methods and datasets, as well as validation of results through statistical analysis. In our work not only do we employ a comparative study for several hate speech

classification methods, but we also cover several data enhancing techniques which aim in improving effectiveness of classification methods. Besides that one, this work entails other contributions, analysis and insights that are necessary towards solving the hate speech classification problem.

CAPÍTULO 3

Resampling techniques for class imbalanced text classification

Class imbalance is the problem in which the distribution of samples across the classes are highly skewed in the training set. Thus the classifier generated under those conditions is biased towards the majority class and can have unsatisfactory prediction power, specially for samples belonging to the minority class.

In most cases (including the hate speech one), the minority class is the class of interest, so techniques to deal with the problem of class imbalance are very important. The range of class imbalance techniques can be divided into data based techniques, that alter the datasets in some way to diminish imbalance, or method based techniques, which work towards creating novel or altering existing classification methods. This work approaches the former set of techniques. *Resampling* techniques for the imbalance problem aim to rearrange the data collection in order to bridge the gap, and reduce or eliminate entirely the discrepancy between the number of documents of the majority and minority classes. Such rearrangement can be done by *undersampling* which consists in removing documents from the majority class. Another option is *oversampling* which increasing the number of instances that belong to the minority class. Oversampling can be achieved through the replication of current minority class documents, or through the generation of synthetic instances. Both techniques can also be combined together to balance the training set.

After using undersampling techniques, it is safe to conclude that the training set as whole is reduced, since the classes will be reduced based on the size of the minority class. According to Padurariu and Breaban [40] and Akbani et. al [3], undersampling techniques are harmful to text classification. Despite the conclusions made by those authors, we wanted to draw our own conclusion in the realm of hate speech detection. Thus, in this work we decided to experiment with only one type of undersampling technique, the Random Undersampling (RUS), whereas for the oversampling techniques we expand through different approaches. All these techniques are described below.

3.1 SMOTE

Synthetic Minority Over-sampling Technique (SMOTE) is an oversampling technique first described in [10]. It consists in generating new synthetic instances for the minority class. For an instance d_i from the minority class, new synthetic examples are generated by combining d_i with each of its k nearest neighbors[1] also of the minority class. The combination with the neighbors is based upon the amount of oversampling that is required. Let $\overrightarrow{x_i}$ be the feature vector for the document d_i and $\overrightarrow{x_j}$ the feature vector for one of the k neighbors of d_i. Consider g to be a random number between 0 and 1. Formally, a feature vector $\overrightarrow{s}$ for a new synthetic sample d_s is create through the Equation 3-1.

$$\overrightarrow{s} = \overrightarrow{x_i} + g.(\overrightarrow{x_j} - \overrightarrow{x_i}) \tag{3-1}$$

3.2 Borderline-SMOTE

In a classification algorithm, the borderline instances are the samples that are close to the decision border that separates the instances from each class. Such samples are the ones that have the bigger chance to be misclassified by a classification model.

The *Borderline-SMOTE* technique, which will henceforth refer to as BSMOTE, was proposed in [26], and as the name suggest, it is derivative from the SMOTE technique [10]. The main difference is the way that the feature vectors are obtained in order to generate the new synthetic instances. For this method only the borderline samples and their k nearest neighbors are oversampled, which means BSMOTE only oversamples based on instances that have a high chance of being misclassified as instances from the majority class. After discovering the borderline samples, new synthetic samples are generated for them, as explained next, strengthening the borderline instances.

After discovering the instances that contain the risk of being misclassified as instances from the majority class, those instances form the P set. A minority class sample is said to have that risk if most of its k nearest neighbors belong to the majority class. Let $\overrightarrow{x_i}$ be a sample from the minority class that contains that risk and $\overrightarrow{p_i}$ a sample that is in the neighborhood of $\overrightarrow{x_i}$, where $\overrightarrow{p_i} \in P$. The new synthetic sample is generated through Equation 3-2, for $g \in [0,1]$.

$$\overrightarrow{s_i} = \overrightarrow{x_i} + g.(\overrightarrow{p_i} - \overrightarrow{x_i}) \tag{3-2}$$

[1] In this implementation, the authors use $k = 5$.

3.3 SVM-SMOTE

As another SMOTE [10] derived technique, the oversampling method of SVM-SMOTE differs in the way it selects the instances that will be used for the synthetic generation of instances. In this particular technique, a SVM classifier is used to find the hyperplane that separates the classes with the largest distance. After finding the optimal hyperplane and the support vectors, SVM-SMOTE creates new synthetic instances based on the support vectors and their neighborhoods. If the minority class sample, $\overrightarrow{x_i}$, has more neighbors that belong to the majority class, Equation 3-3 is used to interpolate $\overrightarrow{x_i}$ and the support vector $\overrightarrow{sv_i}$ of the minority class to form $\overrightarrow{s_i}$. If the minority class sample's neighborhood is formed most by samples from the minority class, then Equation 3-4 is used to generate the new synthetic sample. For both equations, consider $g \in [0,1]$.

$$\overrightarrow{s_i} = \overrightarrow{sv_i} + g.(\overrightarrow{sv_i} - \overrightarrow{x_i}) \tag{3-3}$$

$$\overrightarrow{s_i} = \overrightarrow{sv_i} + g.(\overrightarrow{x_i} - \overrightarrow{sv_i}) \tag{3-4}$$

3.4 *K-Means*-SMOTE

The *K-Means SMOTE* method (KSMOTE), presented in [17], employs the classical $k-means$ clustering algorithm paired with SMOTE [10] to oversample minority class from the data collections to deal with the imbalanced class problem. The novelty behind this method is that it does not generate synthetic samples indiscriminately without criteria. The method avoids the generation of noisy synthetic samples through oversampling using only samples that belong to predetermined clusters.

The method is divided in three basic steps : *clustering*, *filtering* and *oversampling*. On the first step, the training set is used to form k groups through the $k-means$ clustering algorithm. In the filtering stage, the clusters that are going to get resampled are selected. The selected clusters are the ones that have a high proportion of minority class samples. Finally, after the clusters are defined, SMOTE is used to oversample each cluster individually (i.e. SMOTE is applyed to the elements of the clusters, generating new artificial vectors), until it achieves the desired ratio between the majority and minority classes.

3.5 ADASYN

The *Adaptive Synthetic* algorithm (ADASYN), introduced in [28], was developed with the purpose of reducing bias and of learning in an adaptive manner. The algorithm

starts off by calculating the class imbalance ratio (d). If d is smaller than a preset threshold for the maximum tolerated degree of class imbalance ratio, then the algorithm can continue. After the class imbalance ratio (d) is calculated, the algorithm determines the amount of synthetic samples that need to be generated to achieve the desired ratio. The number of synthetic data examples that need to be generated for the minority class is expressed by G. For every minority class sample, its k nearest neighbors are found based on the Euclidean distance between samples. Also, one of k nearest neighbors is selected and g_i synthetic instances are generated, where g_i is the product between G and the distribution density ratio r_i, where $r_i \in [0,1]$.

For each minority class sample x_i, generate g_i synthetic samples. Each synthetic sample is created as follows:

i) Randomly select one sample „x_{zi}, from the k nearest neighbors of x_i;
ii) Generate the synthetic sample according to Equation 3-5

$$\overrightarrow{s_i} = \overrightarrow{x_i} + (\overrightarrow{x_{zi}} - \overrightarrow{x_i}) * \lambda \tag{3-5}$$

In Equation 3-5, $(\overrightarrow{x_{zi}} - \overrightarrow{x_i})$ is the difference vector in n dimensional space and λ is a random number between $[0,1]$.

After the generation of G synthetic samples, the training set has the desired class imbalance ratio.

The idea of ADASYN is to use the distribution density ratio as a criterion to determine the amount of synthetic samples that will be created for each minority class sample. Density distribution is a measurement for weight distribution for different minority class samples based on their learning difficulty by a classification method, which means ADASYN not only treats the class imbalance problem but also indirectly forces the classifier to focus on the hardest samples to learn during model training.

3.6 Random Oversampling

The Random Oversampling (ROS) technique consists in choosing randomly a sample from the minority class with replacement, and in duplicating it, augmenting the training set. Samples are duplicated until the ratio between the sizes of the classes achieves a desired value. ROS is the most simple and less time consuming oversampling technique and it does not create synthetic documents as do the previous presented ones.

3.7 Random Undersampling

The simplest undersampling technique, called Random Undersampling (RUS), randomly removes samples that belong to the majority class from the training dataset. The random choice and removing is repeated until the desired ratio between the sizes of the classes is achieved.

CAPÍTULO 4

Method based on c_features for class imbalanced text classification

The Compound Features a.k.a. c_features technique was proposed in [18]. It is a feature expansion method which generates new and informative features with the objective to enhance document representation for text classification. C_features are formed by a set of two or more words that co-occur in various documents. Observe that c_features differ from n-grams or phrases, because no order of occurrence in the documents is assumed among the component words of a c_feature. It is just sufficient for the components to occur together in the documents.

The objective is to find informative c_features, i.e., c_features that are more frequent in one class c than in others, so that these c_features can better characterize documents as belonging to c. The idea is that the individual occurrences of the component words of a c_feature may be spread among classes but their occurrence as a set, i.e., their occurrence together must be predominant in a unique class. Once a informative c_features is found, it is used to expand both training and test documents where its component words occur together. Thus a new feature representing these c_feature is inserted in the document representation.

C_features were proposed to be used with machine learning techniques where documents can be represented as vectors in a V-dimensional vector space, where V is the vocabulary of a collection of documents. C_features expand the dimensions of the space since they are used as new words. They were proposed to enhance text classification in general, so their use was not neither focused to the problem of class imbalance nor to binary classification specifically.

In what follows we explain how c_features technique works as proposed by Figueiredo et al. [18]. Next, we show how we adapted the technique to compute c_features more efficiently and also to use them in binary and imbalanced classification scenarios as the one of hate speech detection.

4.1 Original c_features expansion technique

The complete technique as described in [18] is composed of three steps which we describe as follows.

Enumeration step. This step determines how candidate c_features are generated. As in [18], we only describe the generation of c_features containing two words[1]. To avoid generating all the $2^{|V|} - |V|$, combinations of two words to form the candidate c_features, where V is the vocabulary of the training set, authors in [18] first rank the words in V in the descending order of their information gain (infogain) [35] and use only the top N words to generate the two-words c_features. Thus, they generate $2^N - N$ candidate c_features, where $N << |V|$. Finally the candidate set of c_features is pruned by eliminating those c_features that have support (i.e., number of documents where they occur) smaller than a threshold *min_s* informed as a parameter of the method.

Ranking step. In this step the discriminant power of each c_feature obtained in the previous step is computed using the *dominance* measure. The dominance of a feature f in a class c is the conditional probability of c given the occurrence of f in the training set, i.e $P(c|f)$. Let $F = \{f_1, f_2, ..., f_{|F|}\}$ be the c_features candidate set associated to the training set of a data collection, $C = \{c_1, c_2, ..., c_{|C|}\}$ the set of labels, and $df(f, c_j)$ be the number of training documents associated with class c_j that contains the c_feature f, the definition of *dominance* is stated in Equation 4-1.

$$dom(f, c_j) = \frac{df(f, c_j)}{\sum_{j=1}^{|C|} df(f, c_j)} \tag{4-1}$$

Expansion step. The third step is the actual document expansion where the c_features are added to the documents' vector representation. There are two preconditions for a c_feature f to be added to a document d: 1) d must contain the two words composing c_feature f; 2) there must be a class c, such that $df(f, c)$ is greater than a minimum threshold *mim_d* informed as a parameter of the method. Finally, the expansion of d with c_features satisfying the preconditions depends on whereas d is a training or a test document. If d is a training document we only insert the c_feature f in d if there is a class c, such that $df(f, c) > mim\_d$ and c is the class of document d. If d is a test document, we do not know its true class, so we include the c_feature f in it as long as f satisfies the two preconditions.

[1] Computing c_features with more than two words has a high computational cost. For c_features with x composing words the number of possibility of c_features in the order of $x^{|V|}$, where V is the vocabulary of the training set.

The parameters for the c_feature expansion method present in [18] are: *i) N* - the number of words with best infogain values to be used, *ii) min_s* - the minimum support value a c_feature needs to have to be considered a candidate c_feature, and *iii) min_d* the minimum dominance value a c_feature needs to have in some class c to be included in documents.

4.2 Proposed modifications to the c_feature expansion technique

There is a sensible point in first step of the original c_feature method which is the selection of the words to form c_features. In the original expansion step only the N words with the best infogain values are used. This is limiting because it is possible to exist a set of words $\{w_1, w_2\}$ where both or at least one of the words has low infogain and c_feature $\{w_1, w_2\}$ still having a high dominance and satisfying the minimum support threshold. This c_feature, although useful, would be filtered out in the first step of the original method.

Instead of filtering out words with low infogain and have to tune the parameter N, we modified the expansion step to include all **possible** non-ordered pairs of words. By "possible" we mean any pair formed by words that co-occur in at least one document of the training set. Notice that not all pairs of words in $2^{|V|} - |V|$ are possible, since there may be no document in the training set containing the pair. We compute all possible pair of words by first creating an inverted index for the training set. Next, for each word in the index we compare its inverted list with the inverted list of all the other words in the index. If the intersection of two inverted list is empty the corresponding pair of words is not possible. During this process, for every possible pair we can find the number of documents in each class where the pair occurs by inspecting the training documents present in the intersection of the two corresponding inverted lists of the words forming the pair. Thus, during the process of obtaining all possible pairs we also gradually compute for each set $\{w_1, w_2\}$ the value of $df(\{w_1, w_2\}, c)$ for all $c \in C$ and can also compute the support of $\{w_1, w_2\}$, which makes the second and third step to be more efficient. Thus, we substitute the enumeration step of the original method by the enumeration step described above.

The method described in [18] was proposed to the multi-class problem[2]. Also, the the technique was used to expand documents of all the classes. The use of the original expansion step as it is in a binary and imbalanced scenario like the one found in hate speech detection may result in a representation bias and may not be as effective. The

[2]In the multi-class classification there may have more than two class in the collection, although each document pertain to only one of the classes.

problem is that the number of c_features pertaining to the majority class is in general greater than that of c_features belonging to the minority class. Since test documents are expanded with any c_feature satisfying the preconditions of the expansion step, test documents which in reality pertain to the minority class tend to be overwhelmed with c_features of the majority class, worsening instead of benefiting the classifiers.

We propose a simple modification of the expansion step: we only use c_features satisfying the two preconditions but which have a great value of dominance associated with the minority class. Thus, we expand only training documents of the minority class and test documents with c_features with high dominance in the minority class, discarding c_features associated to the majority class.

CAPÍTULO 5

Experimental Setup

5.1 Datasets

We evaluated the effectiveness of both classification methods and data augmentation approaches on seven hate speech datasets proposed in the literature. Three of them contain sentences in Portuguese (those with prefix PT_ in their name) and four contain text written in English (those with prefix EN_). Statistics about the datasets used are shown in Table 5.1. Next we give a short description of each dataset.

PT_OFFCOM In [14] a dataset composed of comments about news from the Brazilian news site G1[1] is presented. Both the comments and the news texts are written in Portuguese. The dataset is about offensive speech (not only hate speech).The authors state that the news categories with the most offensive comments were politics and sports, thus, the dataset was limited to those subjects. The authors generate two versions of the dataset, the first one is the OffComBr-2 which was composed of comments that were considered offensive by at least two judges. The second one is OffComBr-3, which is formed by comments which have the agreement of tree judges about their offensiveness. Whenever a judge found a comment offensive, he was asked to further classify the comment into six disjoint categories. OffComBr-3 contains 1033 comments, where 831 of them were not considered offensive (*no*) and 202 were offensive (which included comments with *Xenophobia*, *Homophobia*, *Sexism*, *Racism* and *Cursing*). In this work we used only the binary version OffComBr-3 (i.e. *offensive* and *non offensive* which we refer to as PT_OFFCOM.

PT_HS The work in [20] presents a dataset of tweets written in Portuguese. The dataset was obtaining by both searching for hate-speech related keywords through Twitter search API and for hate related profiles via Twitter profile search API. The final set is composed

[1] g1.globo.com.

of 5,670[2]. The authors conducted two types of classification for the dataset: a) a binary classification where messages were classified as *hate speech* (1,788 messages) or *not hate speech* (3,882 messages) and b) a multi-label classification where the *hate speech* messages were further classified hierarchically under 81 subclasses. In this work we consider only the binary classification of the dataset obtained in the GitHub repository, maintaining the same class distribution, which are composed of 1,788 messages that belong to the hate speech class and 3,882 being messages that were not labeled as hate speech.

PT_RACISM Is a dataset proposed in [11], formed by text in Portuguese. It is composed of 2,022 Twitter messages which were obtained by searching a set of words, via Twitter API, that could have a racist connotation in Portuguese. The majority of the messages (1,676) were labeled *non-racist* by the annotators and 346 messages were labeled as *racist*. Although the authors in [11] did not coin a name to the dataset, here we refer to it as PT_RACISM to reinforce its main characteristics which are: a dataset of messages in Portuguese and being of a specific hate speech category - racism.

EN_HS_3 In [12] a dataset was obtained with the intention to separate hate speech text from just offensive text. The dataset is composed of tweets written in English. The messages were obtained by submitting queries to Twitter search API containing words obtained from a hate speech lexicon compiled by *Hatebase.org*. A set of 24,783 tweets containing words of this lexicon were labeled by workers from a crowd-sourcing service. Annotators were asked to label the messages into three disjoint categories: *hate speech* (HS), *offensive speech* (OFF) and *neither* (N). Contrary to the majority of the other datasets considered in this work, the category with non-offensive text (N) is much smaller than the category OFF, with only 4,163 messages. Category OFF contains 19190 messages and category HS contains 1,430 messages. Given that the dataset is overly skewed in favor of offensive messages, which makes it a non realistic sample of Twitter messages, we decided to not use the messages in OFF category. Thus, EN_HS_3 contains only 5,593 messages of the original dataset in [12], formed by messages in HS (74.5%) and in N (25.5%).

EN_HS_Z In [54] the authors presented an extended version of the dataset obtained previously in [55]. It is a collection of tweets obtained by searching with the Twitter API for a set of common slurs and terms frequently related to hate speech. The dataset is

[2]This is the number of messages in the dataset obtained from https://github.com/paulafortuna/Portuguese-Hate-Speech-Dataset. It diverges from the number reported by the authors in [20] where they report 5,668 messages.

formed by a tweet ID and its labels. For the dataset to be used, the text corresponding to a tweet ID must be recovered using Twitter API. However, since the dataset was collected in 2016, some of the tweets do not exist any more. Thus, EN_HS_Z is a subset of dataset in [54], formed by tweet messages we could recover, which amount to 6,909 messages. The original collection was partitioned in four categories: *Sexism*, *Racism*, *Both*[3] and *Neither*. We were able to recover 911 tweets of class *Sexism*, 50 tweets belonging to *Racism*, 50 of class *Both* and 5,850 of *Neither*. We performed the union of the *Sexism*, *Racism* and *Both* classes into a single class *Hate Speech* (H) which contains 1,059 tweets. We maintained the class *Neither* from which we were able to recover 5,850 tweets.

EN_GIBERT In [13] it is presented a dataset derived from Stormfront, a white supremacy forum. Contrary to the other datasets, the label annotation has been performed at sentence level as opposed to full-text annotation. Sentences were obtained by first collecting a subset of 22 sub-forums covering diverse topics and nationalities randomly sampled to gather individual posts uniformly distributed among sub-forums and users. Only English posts were considered. The collected posts were segmented in sentences which were grouped in batches. Each batch is composed of 500 complete posts with approximate 1,000 sentences per batch. The dataset is composed of 10,945[4] sentences contained in 10 batches that were manually annotated in three categories: *Hate*, *No Hate* and *Skip* which is formed by sentences where the annotator was not able to classify. We use a subset of this dataset which we refer as EN_GIBERT, where we remove sentences classified as *Skip*. Thus, EN_GIBERT contains 10,563 sentences. 9,367 of them pertain to class *No Hate* and 1,196 were labeled as *Hate*.

EN_GOLBECK the work in [25] developed a dataset intended to be used to detect online violent harassment. For that dataset 20,360 messages from the social network Twitter were tagged by up to three annotators. The annotators labeled the messages following a code book provided by the authors which guided them to identify “the worst, most offensive or violent tweets”, “threat”, “hate speech”, “directed harassment”(to an individual or group), “Potentially offensive” and “Non-Harassing” tweet messages. Finally, the messages were labeled under two disjoint classes: *Harassing* with 5,285 messages and *Non-harassing* with 15,075 messages. It is important to highlight that the authors did not report any experiment using machine learning methods with this dataset.

[3]Category Both is formed by messages which contains messages classified as both Sexism and Racism.

[4]This is the number of sentences in https://github.com/Vicomtech/hate-speech-dataset. It differs from the number of sentences reported in the article (10,568 messages).

Tabela 5.1: *Dataset Statistics*

Dataset	# Examples	Doc. Avg. Size	Vocab. Size	Class Distribution	Imb.Ratio
PT_OFFCOM	1033	61	3574	**N**:80.44%, **H**:19.56%	1:4
PT_HS	5670	61	11611	**N**:68.46%, **H**:31.54%	1:2.2
PT_RACISM	2022	77	6684	**N**:82.88%, **H**:17.12%	1:4.8
EN_HS_3	5593	52	19953	**N:**74.50%, **H**:25.50%	1:2.9
EN_HS_Z	6909	50	9512	**N**:84.68%, **H**:15.32%	1:5.4
EN_GIBERT	10563	55	14544	**N**:88.6%, **H**:11.32%	1:7.8
EN_GOLBECK	20360	66	28987	**N**:74.04%,**H**:25.96%	1:2.8

5.2 Methods, metrics and model evaluation

Classification Methods

In this section we present the classification algorithms we experiment in our comparative study. For this study, we use a wide range of different classification methods with different document representations, and they are separated into three categories : Vector Space Model based Methods (VSM), Deep Neural Networks with Word Embeddings (DNN-WE) and Transformers.

The VSM based classification methods are those which use the traditional and simple vector representation of documents, also known as Bag-of-Words (BoW), as feature representation. We used frequency of each distinct term in a document as the weight of the terms in the document vectors. We used the following classification methods in this category: Support Vector Machine (SVM) with a linear kernel, Logistic Regression (LR), included Naive-Bayes (NB) and Random Forest (RF), Multilayer-Perceptron (MLP), as well as more complex architectures as Convolutional Neural Network (CNN) [29], and a Bidirectional Long Short-Term Memory (BiLSTM).

The DNN-WE category is formed by Deep Neural Network (DNN) classification methods that use more sophisticated form of input. The category is formed by two DNN classification methods : a Long Short-Term Memory (LSTM) and another implementation of BiLSTM. Both of these DNN methods are extensively described in the works of [6] and [2], respectively. These two neural networks methods use GloVe [42] word embedding vectors as document representation. The authors in those papers only perform experiments with English datasets, so they only needed one Glove word embedding pre-trained model. For our English datasets we used the exact same model, but since in this study we also work with Portuguese datasets, we had to obtain a separate Glove pre-trained model, which we obtained in the NILC-Embeddings repository [5] which was first introduced in [27].

[5] http://www.nilc.icmc.usp.br/embeddings

In the transformers category, we perform the fine-tuning of state-of-the-art models BERT [16] and RoBERTa [32]. These methods have a specific form of document representation, where special tokens are inserted into the documents/sentences that are input to the model training process. We performed the fine-tuning of pretrained models for all the datasets, which include datasets in Portuguese and English. For both language models we used the available resources found at HuggingFace[6]. For BERT we used the base pre-trained model (*bert-base-cased*) introduced in [16] and made available by the HuggingFace team[7]. For the Portuguese datasets, we used *BERTimbau*, a pre-trained model for Portuguese, that has 12 layers and 110M parameters. It was introduced in [49] and also made available at HuggingFace[8].

For the RoBERTa model, we used the smaller version of the English pre-trained model (roberta-base), first introduced in [32], and it was pre-trained using the combination of five text corpora that together add up to 160GB worth of text. For the Portuguese datasets, we used the pre-trained model *BR_BERTo*, available at HuggingFace [9], which was trained using a corpus of 6,993,330 sentences with 150,000 tokens in the vocabulary.

Metrics and Model Evaluation

To evaluate effectiveness of the classifiers generated with the various classification methods we used the macro averaged F1-score (Macro-F1) which is the average of F1 measure computed for all classes. In our case, we consider only two classes: hate speech and non-hate speech. In order to compute the F1 measure for a class c, the system-made decisions on the test set with respect to c must be divided into three groups: True Positives (TP_c), False Positives (FP_c) and False Negatives (FN_c), respectively. The terms positive and negative refer to the classifier's prediction, and the terms true and false refer to whether that prediction corresponds to the external judgment. The F1 measure for a class c is described in Equation 5-1.

$$F1_c = \frac{2TP_c}{2TP_c + FP_c + FN_c}. \tag{5-1}$$

Thus, Macro-F1 is computed as described in Equation 5-2, where C is the set of class labels.

$$Macro - F1 = \frac{\sum_{c=1}^{|C|} F1_c}{|C|} \tag{5-2}$$

[6]https://huggingface.co/
[7]https://huggingface.co/bert-base-cased
[8]https://huggingface.co/neuralmind/bert-base-portuguese-cased
[9]https://huggingface.co/rdenadai/BR_BERTo

F1 of the minority class is a measure usually adopted for class imbalance problems. However it is also important in the hate speech scenario that classifier also performs well in the non-hate speech class. Thus, we opted to use Macro-F1 instead.

All techniques were evaluated using 5-fold cross-validation, where the training set is composed by 4/5 of the entire dataset and 1/5 is used as test set. We randomly chose the documents to compose each of the five folds and for each dataset we stored the five train/test splits obtained by cross-validation to guarantee that all classifiers are trained with the same training set and tested with same test sets.

Parameter tuning

The parameters for some of the classification methods were set by performing *Grid Search*[10] on the training set. Since performing grid search is a long process, we decided to restrain the range of parameters of the search, and they are outlined in Table 5.2. As stated before, due to the computational complexity of Grid Search, we only performed it on some of the faster classification methods, such as Linear SVM (SVM), Naive-Bayes (NB), Logistic Regression (LR) and Random Forest (RF). The aforementioned classification methods were implemented using the *scikit-learn* library [11].

All the parameters that were not described on Table 5.2 were used with their default values.

The grid search on the classification methods is performed on all five training sets of generated with five-fold cross-validation. For each training set, we perform an inner three-fold cross validation. We train the classifier on two parts (subdivision of the training set) and one part is used to test for Macro-F1. After performing the test, we find the best available parameters based on the ones that lead to the highest Macro-F1. The best parameters are then used for the final model training.

For the MLP method, two fully-connected layers were used, containing 256 neurons each, interspersed with Dropout layer of ratios 0.4 and 0.2, followed by an output fully-connected layer with k neurons, where k is the number of classes in the evaluated dataset, with softmax activation. This architecture is compiled using categorical cross-entropy as the loss function, and optimizer Adam [12] was used with a learning rate of 0.001. For the BiLSTM method, a simple architecture that is composed of an input layer, a layer with a dropout layer with ratio of 0.25, followed by a bidirectional LSTM layer, and a dropout layer with ratio of 0.5 with an output layer with softmax activation with k

[10] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html#sklearn-model-selection-gridsearchcv

[11] https://scikit-learn.org/stable/

[12] https://keras.io/api/optimizers/adam/

Tabela 5.2: *Range of parameters considered in the Grid Search process. All parameters that were not described were used with default values.*

Methods	Parameter
MLP	learn_rate = 0.001 loss= 'categorical_crossentropy'
BiLSTM	
CNN	
Random Forest (RF)	n_estimators = [30,50,70,90,100,200,300] criterion = ['gini','entropy'] warm_start = ['False','True']
Naive-Bayes	alpha=[0.001,0.005,0.001,0.1,0.5,1] fit_prior=[True, False]
Linear SVM	kernel = 'linear' C= $[2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, 2^{3}, 2^{5}, 2^{7}, 2^{9}, 2^{11}, 2^{13}, 2^{15}]$
Logistic Regression (LR)	penalty=['l1','l2'] dual = [True, False] solver= ['liblinear', 'newton-cg', 'lbfgs'] C=$[2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, 2^{3}, 2^{5}, 2^{7}, 2^{9}, 2^{11}, 2^{13}, 2^{15}]$
BERT	MAX_LEN = 156, batch_size = 16, learning_rate = 2e-05, epochs = 4
RoBERTa	

neurons, where k is the number of classes in the evaluated dataset. For the CNN method, we use a small variation of the model described in [29]. The neural network models were implemented using Keras (2.3.0) [13] and Tensorflow (2.1.0) [14].

As for the BiLSTM and the LSTM which are member of the DNN-WE, we used the model and parameters as described in [6, 2] and the code is available at the author's Github repository [15].

As previously mentioned, we do not perform grid search on the transformers, the neural network methods nor on the DNN-WE methods. We do this because performing a grid search on those methods would be too computationally expensive. For the neural network methods members of VSM, we relied on the literature to set the hyper parameters presented on Table 5.2. For instance, the authors in [29, 52] perform sentence and short text classification using neural network methods, namely Convolutional Neural Networks (CNN), and both works achieve satisfactory results using 0.001 as the learning rate for the optimizer. The authors in [57, 8] use neural network methods for hate speech detection, and they chose categorical cross entropy as their loss function.

Regarding the transformer methods, for BERT parameters we selected values based on authors' recommendation. In [16], the authors say that optimal hyper-parameter values are task-specific, but recommend a range of values that work well across all tasks. Devlin et. al [16] recommends that the fine-tuning of BERT be trained from two to four epochs, that batch size be either 16 or 32, and that learning rate be $5e-5$, $3e-5$ or

[13] https://keras.io/
[14] https://www.tensorflow.org/
[15] https://github.com/aymeam/User_distribution_experiments

$2e-5$. For RoBERTa, we used the same parameters we used for BERT. The RoBERTa is a transformer technique derived from BERT. The authors in [32] also recommend a range of parameters to follow for different task. For the classification task, the authors recommend learning rate $1e-5$, $2e-5$, $3e-5$, batch size of 16 or 32. The final decision on the parameters were done based on previous experiments that suggested that the values displayed on Table 5.2 were the best option for our task.

5.3 Data transformation techniques

In this study we combine, whenever possible, the classification methods with the data transformation (resampling and c_features) techniques described in Sections 3 and 4 to alleviate the class imbalance problem in hate speech detection. For the resampling techniques we used the implementations available in Imbalanced-Learn[16] and we use a 1:1 ratio, meaning that we resample the training set until both classes are equally sized. For the c_features we use our own implementation.

Due to the specificity of the input data representation required by some classification methods, not all data transformations could be used with all of them. The transformer methods (BERT and RoBERTa), use a specific type of document representation, formed by the raw sentences of the text composing the documents with some specific control tokens inserted in the text. The order in which the words appear in the sentences is fundamental to these methods. Oversampling techniques which generate synthetic documents and the expansion with c_features work in the Vector Space Model (VSM) generate new vectors. The order of the words in the vector representation is completely lost. Thus, none of these techniques can be used with transformers. In fact, only ROS and RUS can be used in conjunction with transformer methods, because document representations are only duplicated by the first technique and only removed by the second one i.e., the original document representation expected by transformers is not affect by both ROS and RUS.

The two DNN-WE methods (LSTM [6] and BiLSTM [2]) use word embeddings for word representation which are dense vectors with low dimension. The code for the resampling techniques available in the *Imbalanced-Learn* [17] library can cope with word embeddings, except the code for SVM-SMOTE, because the internal SVM implementation does not support word embeddings. Thus, SVM-SMOTE was the only resampling technique we could not use with the two DNN-WE classification methods. We also could

[16] https://imbalanced-learn.org/stable
[17] https://imbalanced-learn.org/stable

not use the two DNN-WE with c_features because a c_feature is an artificial feature, thus there is no word embedding for it.

For the c_features expansion method, we found, for each dataset, the best values for the two parameters min_s (minimum support) and min_d (minimum dominance) by means of grid search as we show next. As explained in Section 5.2 we performed five-fold cross-validation previously to any experiment and kept the five splits for each dataset. To find the two parameters for the c_feature method, we perform grid search only in the training set of the first of the five splits for each dataset[18]. We then apply another five-fold cross-validation intern to this training set. For each classification method we train a classifier on four parts (subdivision training set) and assess Macro-F1 on the other part (subdivision test set). We perform this grid search by combining the values of min_s and min_d in the ranges described in Table 5.3, while keeping record of which combination resulted in the higher Macro-F1. The pair of threshold values that generates the best Macro-F1 value is used as the definitive values both for min_s and min_d for that dataset and that specific classification method, in all five splits.

Tabela 5.3: *Range of values for support and dominance used in the grid search*

Dominance	0.55,0.6,0.65,0.7,0.75,0.8,0.85,0.9,0.95,1
Support	0.001,0.002,0.003,0.004,0.005,0.006,0.007,0.008,0.009,0.01,0.05,0.1,0.5

For both resampling and c_features, when training classifiers for datasets transformed with these techniques, we adopted the same parameter tuning described in section *Parameter tuning* of Section 5.2.

5.4 Hardware configurations

In our experiments, some classification methods like the neural networks methods, Deep Learning networks and transformers make use of hardware acceleration (GPU) . We also experimented with classification methods that use traditional CPU cores. The use of two types of cores (with and without acceleration) allowed us to run both CPU and GPU methods at the same time.

We used two hardware configuration set. The first set, C_1, was used to perform the experiments necessary to answer the first five research questions. C_1 is formed by two Intel Xeon Gold 6148 processors with 6GB of RAM memory for the methods that use

[18]We are conscious that this decision does not allow for obtaining the best possible values for both parameters in the five training sets, however performing grid search in all the training sets of all the five splits for the seven dataset would be too time consuming. Despite this decision, the expansion with c_features could enhance all the classification methods it could be used with as we show in Section 6.3

only CPU. In C_1 the methods that take advantage of acceleration were executed on GPU server containing a NVIDIA Tesla V100 with 6GB of RAM memory.

In order to answer the research question RQ6 (Section on 6.6), we needed a more controlled environment in order to be able to fairly assess cost-effectiveness. For that particular research question, we used a different hardware configuration (C_2), which is composed of: a) one CPU with AMD EPYC 7452 processor, 4GB of DDR4 RAM memory and b) one GPU NVIDIA Tesla V100 with 6GB of RAM memory. All CPU and GPU configurations use CentOS Linux 7.8 as the operating system.

5.5 Statistical test

In this work we compare the effectiveness of a series of classification algorithms and combinations of classification algorithms with data engineering approaches, using seven hate speech datasets. In this case, it is a consensus in literature [15, 23, 7] that the Friedman Test [22] should be applied to accept or reject the null hypothesis (H_0) that all classifiers perform equally on the set of datasets considered. Friedman Test is the non-parametric equivalent to the *repeated-measures* ANOVA. According to Demšar [15] repeated-measures ANOVA is not recommended in this case because it makes assumptions about the sample data that can not be assured in the case of the distribution of effectiveness values of classifiers on multiple datasets.

Demšar [15] recommends that when comparing classification algorithms on multiple datasets, the effectiveness values should be obtained from repeated execution of the algorithms in each dataset and that all algorithms should use the same training/test splits in all datasets. In our case, we use 5-fold cross validation in all seven data sets and all classifiers are trained and tested on the same partitions. We report the mean of the effective measure (Macro-F1) of each algorithm on each dataset.

Whenever the Friedman Test rejects H_0, that is, confirms that there are statistical differences among algorithms, a post-hoc test comparing pairs of algorithms on the multiple datasets must be applied [15, 23, 7]. In our work we use The Wilcoxon Signed-Rank Test with Bonferroni correction as post-hoc test, as suggested in [7].

In some of our experiments we need to compare only two classifiers with each other: one derived from a learning algorithm applied to the original datasets and the another derived from the same algorithm but applied to specific versions of the datasets[19]. Both classifiers are compared over multiple (seven) datasets. In this case we use the Wilcoxon Signed-Rank Test as recommended in [15]. However we also apply

[19]For instance a version of the training set obtained by oversampling or undersampling approaches or by using the c_features feature engineering approach.

the Bonferroni correction to this test to control the family-wise type I error [7]. In all our experiments, we have set the experiment level of significance at α=0.05. Thus, H_0 is rejected whenever the p-value is inferior to α.

CAPÍTULO 6

Results

In this chapter we answer our research questions by means of an extensive set of experiments on the considered datasets. Thus, each section is dedicated to answer a specific research question.

6.1 RQ1 - How effective are the different classification methods when applied to current datasets about hate speech?

We compared classifiers generated by the eleven classification methods discussed in Section 5.2 using the datasets presented in Section 5.1. The non gray cells of Table 6.1 show the average Macro-F1 values (outside the parentheses) over 5-fold cross-validation of each method on each dataset. Given a non gray cell in the interception of a line l and a column c of the table, the value between parentheses, just after the Macro-F1 value, represents the rank position of method corresponding to line i on dataset corresponding to column c. The column with gray background color contains the average rank position of each method across the datasets.

We applied the Friedman test using the rankings of the classification methods and it rejected the null hypothesis that methods have similar effectiveness in the collection of datasets considered. Thus, we applied the Wilcoxon signed-Rank test as suggested in [7] and confirmed that there are statistical differences in the rankings of each pair of methods. Thus, we can conclude that the differences in the position of each method in the ranking presented in last column of Table 6.1 are statistical significant.

Table 6.1 shows that the Deep Neural Networks based on transformers have proved to generate very good classifiers. In Particular, BERT is the top performer, generating the best classifier in four out of the seven datasets considered. RoBERTa, comes in fourth place in the general rank of Table 6.1 and generated the best classifier for the EN_GOLBECK dataset.

Tabela 6.1: *Macro-F1 results for the methods applied to the datasets. The values in parenthesis indicate the ranking of each classification method in a particular dataset. Last column correspond to the average ranking position of each method.*

	PT_OFFCOM	PT_HS	PT_RACISM	EN_HS_3	EN_HS_Z	EN_GIBERT	EN_GOLBECK	Av. Rank.
BERT	0.4700(9)	0.7200(1)	0.7920(1)	0.9200(1)	0.8401(1)	0.712(2)	0.43(9)	3.4286
LR	0.6487(2)	0.6775(3)	0.7337(4)	0.8989(5)	0.8168(4)	0.6721(5)	0.5926(2)	3.5714
SVM	0.6333(4)	0.6709(5)	0.7396(3)	0.9043(4)	0.8274(2)	0.6723(4)	0.5890(4)	3.7143
ROBERTa	0.484(8)	0.698(2)	0.501(9)	0.916(2)	0.776(6)	0.766(1)	0.63(1)	4.1429
RF	0.5761(7)	0.6762(4)	0.7859(2)	0.9076(3)	0.821(3)	0.5588(8)	0.592(3)	4.2857
NB	0.6975(1)	0.6631(6)	0.6868(7)	0.8912(6)	0.7638(7)	0.6786(3)	0.5819(5)	5.0000
MLP	0.5990(6)	0.6448(7)	0.7092(5)	0.8881(7)	0.7899(5)	0.5474(9)	0.5798(6)	6.4286
BiLSTM [2]	0.6188(5)	0.6389(8)	0.6988(6)	0.8857(8)	0.7095(8)	0.6224(6)	0.5623(8)	7.0000
LSTM [6]	0.6445(3)	0.6091(9)	0.6813(8)	0.7153(9)	0.6319(9)	0.5636(7)	0.5627(7)	7.4286
CNN	0.4458(10.5)	0.4064(10.5)	0.4685(10)	0.4266(11)	0.458(10.5)	0.47(10.5)	0.4255(10)	10.4286
BiLSTM	0.4458(10.5)	0.4064(10.5)	0.4532(11)	0.4267(10)	0.458(10.5)	0.47(10.5)	0.4252(11)	10.5714

It is important to highlight the competitive performance of LR and SVM learning algorithms. They generate, respectively the second and third best classification models. These traditional methods, based on the simple vector space model (VSM) of documents, outperformed neural network methods that use sophisticated word embeddings as input (LSTM [6] and BiLSTM [2]).While analyzing each dataset separately, we can see that these two methods always present competitive Macro-F1 results. In fact Also whenever BERT is superior to LR the difference in Macro-F1 between them is at most of 8% (in PT_RACISM dataset). This shows that a well calibrated traditional algorithm is still a good choice for hate speech classification, as we see SVM and LR proven to be superior to RoBERTa.

It is also interesting that MLP based on VSM, a simple neural network architecture, performed slightly better than more complex Deep Learning methods based on word embeddings (LSTM [6] and BiLSTM [2]) and largely better than CNN e BiLSTM which also used VSM for document representation.

Table 6.1 shows variability in the performance of transformer based methods, specially among datasets. Even though most datasets are originated from messages from social medial platforms and belong to the same domain (hate speech), the Macro-F1 values fluctuates a lot among datasets. For instance, we can see that BERT is outperformed by LR in 38% in terms of Macro-F1 in the PT_OFFCOM dataset and in 37.8% in the EN_GOLBECK dataset. On the other hand, BERT is at most 7.9% better than LR in the PT_RACISM dataset. Similar relations are kept between BERT and SVM in these datasets. Thus, LR and SVM are more stable across datasets in that their positions in the ranking for each dataset do not vary much.

From the discussion above we note that the learning algorithm based on Vector Space Model (VSM) tend to have competitive effectiveness when compared to more sophisticate methods. One possible hypothesis for the good performance of VSM (i.e. bag-of-words) based algorithms could be that most hate speech messages are characteri-

zed by a group of words generally used for offensive speech. Thus, a good VSM based algorithm which is able to capture term importance for a class is sufficient to perform well on most messages. For some less obvious but also less common hate speech messages it is possible that VSM based methods do not classify them correctly because of their inability to learn more complex syntactic and/or semantic relations between words that are learned by methods which use word embedding or by the transformers. However, we left the investigation of this hypothesis for future work.

It is worth noting that, except for the EN_HS_3 and EN_HS_Z where Macro-F1 are higher than 0.8, hate speech detection considering only textual features is shown to be a difficult task to solve. Even for transformer based classifiers that make use of pre-training knowledge about the language via extensive non-supervised learning have difficulties to differentiate hate speech from non-hate speech. Thus, all methods considered ended up presenting low Macro-F1 values. To further investigate the errors of the classifiers, we generated the confusion matrix for all methods and all datasets. We found that the relation $FN > FP$, i.e the number of False Negatives (FN) being greater than the number of False Positives (FN) is prevalent. The above relation is inverted only for BERT with the EN_HS_3 dataset and for RoBERTa with the EN_GIBERT and EN_HS_Z datasets, where FP is slightly greater than FN. This strongly suggests that for all learning methods, the trained classifiers are influenced by class imbalance which is present in all the datasets. The $FN > FP$ is a prevalent relation even in datasets with the lowest class imbalance, i.e., PT_HS, EN_GOLBECK and EN_HS_3 (see Table 5.1). Thus, class imbalance seems to be an important obstacle for all the classification methods at least in the collection of datasets we considered. Thus, in the next Section we investigate how effective are the techniques most used to cope with class imbalance.

6.2 RQ2 - Do the data resampling techniques improve the classification methods?

As discussed at the end of the last section, class imbalance affects the performance of all classifiers we experimented with. Thus, it is reasonable to ask if data resampling techniques (commonly used as a preprocessing step in class imbalance scenarios) could help enhance results of the classification methods. We considered the use of both undersampling and oversampling techniques in our investigations.

In order to answer RQ2 and make claims of superiority, for each classification method m we confront (whenever possible[1]) two classifiers: m_o obtained by applying m

[1]Some re-sampling techniques cannot be used with some classification methods. For instance, oversampling techniques that generate artificial documents in the vector space model cannot be used with transfor-

to the original dataset and m_r obtained by applying m to a training set (from the cross-validation stage) version of the dataset obtained with some data resampling technique r, where $r \in \{$ SMOTE, ADASYN, BOS, SVMSMOTE, KSMOTE, ROS, RUS$\}$.

For each classifier c in the pair of classifiers $\langle m_o, m_r \rangle$, we first compute the rank position of c in each dataset in regard to its opponent in the pair considering Macro-F1 values. Whenever there is a tie, we assign the mean of the two positions to each classifier (i.e. 1.5 to each one). Next, we compute $mrank(c)$ which is the mean position of component c of pair $\langle m_o, m_r \rangle$ over all datasets. Table 6.2 shows for each method m (lines of the table), the mean rank position of m_o (column ORIG) and the mean rank position of its opponent m_r (columns SMOTE, ADASYN, BOS, SVMSMOTE, KSMOTE, ROS and RUS).

Tabela 6.2: *Paired comparison of average ranking between each method applied to the original datasets and the same method applied to a sampled versions of the datasets. Results in bold represent the best classification methods. Results for the original datasets appear in bold if and only if there is no sampling technique able to improve the corresponding method.*

	ORIG	SMOTE	ORIG	ADASYN	ORIG	BOS	ORIG	SVMSMOTE	ORIG	KSMOTE	ORIG	ROS	ORIG	RUS
Linear SVM	**1**	2	**1**	2	**1**	2	**1**	2	**1**	2	**1.43**	1.57	**1.29**	1.71
NB	**1.43**	1.57	**1.14**	1.86	**1.14**	1.86	**1.14**	1.86	**1.14**	1.86	**1**	2	**1**	2
LR	1	2	1	2	1	2	1	2	1	2	1.57	**1.43**	1.29	1.71
RF	1	2	1.14	1.86	1	2	1.29	1.71	1.14	1.86	1.86	**1.14**	1.29	1.71
MLP	1.17	1.83	1.29	1.71	1.33	1.67	1.29	1.71	1.29	1.71	1.71	**1.29**	1.43	1.57
CNN	1.29	1.71	1.57	**1.43**	1.43	1.57	1.29	1.71	1.43	1.57	1.29	1.71	1.43	1.57
BiLSTM	**1.14**	1.86	**1.14**	1.86	**1.14**	1.86	**1.14**	1.86	**1.14**	1.86	**1.14**	1.86	**1.43**	1.57
LSTM [6]	1.86	**1.14**	1.71	**1.29**	1.86	**1.14**			1.71	**1.29**	2	**1**	1.71	**1.29**
BiLSTM [2]	1	2	1	2	1	2			1.14	1.86	1.86	**1.14**	1.57	**1.43**
BERT											1.86	**1.14**	1.07	1.93
RoBERTa											1.86	**1.14**	1.47	1.57

Our comparison methodology is described as follows. For a given pair $\langle m_o, m_r \rangle$, if $mrank(m_o) >= mrank(m_r)$, we consider that the specific sampling technique r corresponds to a negative answer to RQ2 in regard to classification method m. Thus, we discard classifier m_r from our future considerations, because its safe to say that for our covered datasets not only does it not improve classification, but it also implies in computational overhead (due to oversampling or undersampling). However, if $mrank(m_r) > mrank(m_o)$, we use the Wilcoxon Signed Ranks test, as suggested in [15] for pairwise comparison in multiple datasets, to check if the difference between the two classifiers is statistically significant. If that is the case, r is a positive answer to RQ2 in regard to classification method m and thus, we exclude m_o from future analysis. Otherwise, r is a negative answer to $RQ2$ and we discard m_r. Consequently, whenever there is one m_r, which is statistically superior to m_o we show the value of its mean rank position in bold in Table 6.2. A mean

mer based methods, since these methods assume raw text as input. Also, SVMSMOTE, could not be used with LSTM [6] and BiLSTM [2] because the internal SVM method used in SVMSMOTE is not able to cope with word embedding vectors.

rank position in column ORIG for a method m (a line of table) appears in bold if and only if there is no m_r classifier, based on any sampling strategy that was statistically superior to m_o. Thus methods in bold are considered the "survivors"of the pairwise competition described above.

As Table 6.2 shows, no re-sampling technique was able to enhance classifiers trained with SVM, NB or BiLSTM classification methods in the original datasets. On the other hand, the LSTM [6] method was the only classification method where all sampling techniques produced classifiers with better average ranking than that trained in the original datasets. This is curious because both oversampling and undersampling techniques were able to enhance considerably the LSTM [6] method. At first glance, it seems that LSTM is more sensible to class imbalance than the other methods, but we left further investigations about this hypothesis for future work.

The Random Oversampling technique (ROS) improves seven of the eleven classification methods considered, being the sampling technique that enhanced more classification methods. This is interesting because ROS is the most simple oversampling technique and also the one demanding less processing time. ROS just replicates some training instances, thus it is independent of instance input format (VSM or raw text) and can be used with any classification method. Consequently, it was the only oversampling strategy we could use with BERT and RoBERTa which use the raw text as input. ROS was able to enhance both of these methods. The good performance of ROS is probably because it does not create new samples, but works by replicating already existing training samples that tend to be more similar to the ones from the test set than those artificial samples generated by the other oversampling techniques.

Considering the more sophisticated oversampling techniques, they were only able to improve the LSTM [6] method and ADASYN was the only sampling strategy able to improve CNN. In this case the improvement is small, but statistical significant. Possibly, the general bad results for these techniques are due to the fact that the artificial documents they generate tend to be very different from the real test documents. These artificial documents induce the classifiers to consider non-real regions of vector space. Besides, these techniques demand more processing time than ROS. Despite being useful in other scenarios, oversampling based on synthetic samples generally do not help in text classification as indicated by previous work [58, 44, 24] and confirmed by us in the realm of hate speech detection.

Tables 6.3 and 6.4 show the Macro-F1 values for the seven oversampling strategies for both Portuguese and English dataset, respectively. We separated the datasets in the two tables for space restriction only. Column entitled **ORIG** in the tables contains classifier values obtained on the original datasets (i.e. without any resampling technique).

Table 6.3 shows that oversampling with ROS benefit almost all methods (except

Tabela 6.3: *Macro-F1 results for the oversampling techniques in datasets of Portuguese text.*

PT_OFFCOM							
	ORIG	**SMOTE**	**ADASYN**	**BOS**	**SVMSMOTE**	**KSMOTE**	**ROS**
Linear SVM	0.6333	0.5961	0.5984	0.5977	0.5844	0.5967	0.6513
NB	0.6975	0.6955	0.7074	0.6999	0.6955	0.7095	0.6978
LR	0.6487	0.6052	0.6140	0.6029	0.6266	0.6052	0.6543
RF	0.5761	0.5461	0.5532	0.5610	0.5717	0.5592	0.6199
MLP	0.5990	0.6654	0.6717	0.6427	0.6770	0.6526	0.6374
CNN	0.4458	0.4886	0.4886	0.4886	0.4245	0.4886	0.3097
BiLSTM	0.4458	0.2326	0.3544	0.3948	0.2785	0.3585	0.3383
LSTM [6]	0.6445	0.6192	0.6052	0.6104	-	0.6239	0.6591
BiLSTM [2]	0.6188	0.6215	0.6052	0.6104	-	0.6239	0.6471
BERT	0.4700	-	-	-	-	**-**	0.754
RoBERTa	0.484	-	-	-	-	**-**	0.646
PT_HS							
	ORIG	**SMOTE**	**ADASYN**	**BOS**	**SVMSMOTE**	**KSMOTE**	**ROS**
Linear SVM	0.6709	0.5769	0.5667	0.5748	0.5925	0.5781	0.6772
NB	0.6631	0.6551	0.6479	0.6592	0.6547	0.6489	0.6194
LR	0.6775	0.6162	0.6018	0.6117	0.6287	0.6182	0.6787
RF	0.6762	0.5938	0.5908	0.5912	0.5946	0.5927	0.6888
MLP	0.6448	0.6252	0.6307	0.6216	0.6255	0.6327	0.6469
CNN	0.4064	0.4063	0.4071	0.4063	0.3810	0.4063	0.4550
BiLSTM	0.4064	0.3073	0.2397	0.2397	0.3063	0.3731	0.3759
LSTM [6]	0.6091	0.6095	0.6201	0.6159	-	0.6067	0.6464
BiLSTM [2]	0.6389	0.6161	0.6170	0.6167	-	0.6082	0.6430
BERT	0.7200	-	-	-	-	-	0.7260
RoBERTa	0.698	-	-	-	-	-	0.692
PT_RACISM							
	ORIG	**SMOTE**	**ADASYN**	**BOS**	**SVMSMOTE**	**KSMOTE**	**ROS**
Linear SVM	0.7396	0.6139	0.5941	0.6234	0.6823	0.6918	0.7422
NB	0.6868	0.7008	0.6669	0.6808	0.7007	0.6770	0.6454
LR	0.7337	0.6443	0.6346	0.6583	0.6862	0.6838	0.7570
RF	0.7859	0.6379	0.6155	0.6408	0.6723	0.6672	0.7774
MLP	0.7092	0.6644	0.6386	0.6698	0.6674	0.6630	0.7025
CNN	0.4685	0.3902	0.4563	0.2623	0.4553	0.3060	0.5596
BiLSTM*	0.4532	0.3724	0.3302	0.3302	0.3302	0.3583	0.2756
LSTM [6]	0.6813	0.6862	0.6821	0.7043	-	0.6893	0.6822
BiLSTM [2]	0.6988	0.6921	0.6911	0.6777	-	0.6959	0.7248
BERT	0.7920	-	-	-	-	-	0.7960
RoBERTa	0.501	-	-	-	-	-	0.618

NB and CNN) in the PT_OFFCOM, our smallest dataset. Specially in this dataset, ROS helped considerably both transformer methods, BERT and RoBERTa, providing gains of 60.43% and 33.47% in Macro-F1, respectively, in comparison with using the original dataset. In fact, in all datasets, when ROS does not contribute to the transformer methods it also does not provide loss in effectiveness (except the small loss for BERT in the EN_GIBERT dataset). Thus, we suggest that the use of ROS in conjunction with transformers should be considered in future studies of hate speech detection. When considering the other classification methods, in spite of ROS has enhanced most of them

considering all the seven datasets, there are some benefited methods that did not perform well in some specific dataset (see some example in Tables 6.3 and 6.4) . Thus, ROS is not a "silver bullet" for all cases, but our experiments indicate that its worth to experiment with it in new datasets.

In regard to the undersampling, this technique consists in removing samples from the training set that belong to the majority class until both classes (in a binary classification problem) are equally sized. The undersampling technique covered here, called *Random Undersampling (RUS)*[2], is extremely simple and consists of randomly removing samples from the majority class until all classes have the same number of samples.

Table 6.2 shows that RUS was able to improve only the DNN-WE methods (BiLSTM [2] and LSTM [6]). The problem with undersampling is that it discards many training set samples when reducing the size of the biggest class. This, in most cases affects the learning capacity of the classification methods. In the specific case of LSTM [6] and BiLSTM [2] it seems that class imbalance affects these methods more than a reduction in the training set and both methods are benefited with RUS. However, even for these methods RUS deteriorates the Macro-F1 values for some datasets as shows Table A.3 (in the Appendix).

6.3 RQ3 - Do the c_features enhance classification methods for hate speech detection?

In this section, we investigate whether the use of c_features enhance the effectiveness of classification algorithms for hate speech detection. The method used to compute c_features (see Chapter 4) takes as input documents that are represented by vectors in the VSM, thus c_features cannot be used with the transformer models (BERT and RoBERTa) that have a method-specific form of document representation. Also, c_features cannot be used, at least directly, with the classification methods that use word embeddings as document representation (LSTM [6] and BiLSTM [2]) because there is no word embedding vector learned for a pair of terms. For this reason we use c_features only with classification methods which work with the VSM model, i.e., SVM, NB, LR, RF, MLP, CNN, BiLSTM.

To answer RQ3, we again rely on the comparison methodology we used in Section 6.2 for answering RQ2. Thus for each classification method m, we compare the mean rank position of the classifiers in the pair $\langle m_o, m_c \rangle$ with each other over the seven

[2]https://imbalanced-learn.org/stable/references/generated/imblearn.under_sampling.RandomUnderSampler.html#imblea

datasets, where m_o is the classifier generated by m on an original dataset and m_c is the classifier generated by m on the corresponding dataset expanded with c_features. As in the answer of RQ2, we only consider m_c superior to m_o when $mrank(m_c) > mrank(m_o)$ and the difference between the two mean values is statistical significant according to the Wilcoxon Signed Ranks test.

Table 6.5 shows the result of the pairwise comparison described above. Values in bold represent the best component of each pair. Clearly c_features benefited all the classification methods they could be used with. Specially, they enhanced classification methods that were not improved by any resampling technique, i.e., SVM, NB and BiLSTM.

It is important to highlight that c_features were able to enhance considerably the two best VSM based classification methods, LR and SVM indicated in the results for RQ1. LR was benefited in all the seven datasets (see results for LR in Table 6.5), while SVM was benefited by c_features in six of the seven datasets (except in dataset EN_HS_Z, see Tables A.1 and A.2 in the Appendix). Specially the better gains for both methods were obtained in the three most difficult datasets. The gains for SVM were 7.8% in PT_OFFCOM, 5% in PT_RACISM and 4.7% in EN_GOLBECK datasets, comparing with SVM using original data. For LR, the gains were 6.1% in PT_OFFCOM, 5% in PT_RACISM and 1.7% in EN_GOLBECK datasets.

6.4 RQ4 - How do the best combinations of methods and preprocessing compare to each other?

In the last two sections we investigated if preprocessed versions of a dataset obtained by applying resampling or expansion with c_features were able to generate better classifiers for a classification method m, than that generated by m on the original dataset. In this section we investigate how the "winners"of those pairwise comparisons of the last two sections compete among each other. For this, we computed the mean rank position of these methods regarding Macro-F1 and applied the Friedman test which confirmed differences among the mean positions. Next, we applied the Wilcoxon signed-rank test which also confirmed the statistical difference in the ranking of each pair of methods. The results are shown in Table 6.6.

Considering the mean rank position, the combination of SVM and c_features stands out as the best overall method for hate speech detection, outperforming slightly the state-of-the-art transformer model BERT combined with Random Oversampling, which occupied second place. Logistic Regression (LR) combined with c_features outperforms RoBERTa+ROS. Thus c_features not only enhance all methods they can be used with

(as shown in the answer to RQ3), but they also make the traditional SVM and LR methods very competitive. However, since the differences between the mean positions of two consecutive methods is too tight among the top four methods in Table 6.6, we further investigated thees these ranking positions. We found that, inspite of both BERT and RoBERTa (combined with ROS) occupy the top two methods for most datasets, they appear as one of the worst classification methods in at least one of the datasets. More precisely, BERT + ROS occupy the 21st ranking position for the EN_GOLBECK dataset, while RoBERTa + ROS occupy the 10th and 20th for the PT_OFFCOM and PT_RACISM, respectively. Those disparities drive their mean rank position up. On the other hand, SVM + c_feature always occupy one of the top five rank positions across in datasets.

While looking at the median rank position instead of the mean, the transformer methods combined with ROS, are tied as the best classification methods (1.5), because those extreme high rank positions in some dataset highly affect the mean, but not the median. SVM + c_features stand as the next best model occupying the median ranking of 4. The LR and RF, both combined with c_features and ROS are, respectively, the next best models. Besides the BERT and RoBERTa combined with ROS on the first place, most of the top median ranking positions are occupied with traditional methods with or without data enhancing techniques. Thus, the best method depends on the way we look at the results. If we consider the methods that are the best in most datasets, BERT is the best model although performing very bad in one of the datasets. On the other, SVM + c_features is the most stable one, in the sense that it has good rank positions and its rank positions in all databases do not vary much.

It is worth noting in Table 6.6 that, except with RF, whenever there are two versions of a method, one using ROS and the other using c_features, the one using c_features performs better. Also, CNN with c_features performed better CNN with ADASYN, the only resampling technique able to enhance CNN as commented in Section 6.2. The use of ROS has also been very effective when compared across all the best classification models. Also, traditional techniques like SVM and NB, without the use of c_features or any resampling technique, still prove to be good choices for hate speech detection. These methods alone outperformed neural network methods with sophisticated forms of feature representation and different data resampling strategies, for instance, CNN, MLP, LSTM and BiLSTM.

6.5 RQ5 - Does the combination of c_features and resampling enhance effectiveness?

Given that: *i*) oversampling with ROS and c_features were shown to be beneficial separately to many classification methods and *ii*) resampling and c_features work on different levels, i.e., the former works by augmenting/reducing samples, while the latter works at feature level, it is natural to ask if the combination of the these two techniques could enhance hate speech detection even further.

To answer this research question, we first performed a resampling technique to the training data and next we generated the c_features over the augmented/reduced training set. The oversampling technique used was ROS, since it was the only one able to enhance a large number of classification methods. As the undersampling technique we used RUS. Table 6.7 shows the results for the PT_OFFCOM dataset, since it was the one most difficult for the majority of methods. Again, we only show results for classification methods we could apply c_features.

As can be seen, in both cases (ROS + c_features and RUS + c_features) the results obtained using the combinations are either worse or statistically tied with the Macro-F1 obtained using only c_features. In the case of ROS + c_features, the problem is that c_features are computed over the augmented class. This makes a set of c_features to satisfy the support and confidence threshold that they didn't satisfy when oversampling was not used. Thus, the the number of c_features is augmented causing a bias in the learned classifier. With ROS + c_features the number of false positives generally augments in relation to using only c_features, because of the excess of c_features generated.

In the case of RUS + c_features, we observed that the number of c_features generated is not sufficient to compensate for the damage caused by undersampling, thus the right and wrong indications of the classifier are very similar to those of the classifier trained with RUS only. Thus, in any case the combination of resampling and c_features did not benefit the classification methods.

6.6 RQ6 - How do the classification methods compare in terms of cost-effectiveness trade-off?

The time taken to generate a classifier ready to make new predictions is a important aspect in selecting a classification method. However, model selection cannot be made by only analyzing time or cost separately. Certain hate speech applications may require a faster/cheaper classifier, even if the effectiveness might be compromised in the

process. On the other hand, there might be situations where the time taken to generate a classifier or special hardware might not be an issue, but a higher model effectiveness is an absolute necessity. On this research question, we investigate the trade-off between cost and effectiveness for several hate speech classifiers.

We use the average elapsed training time in seconds of the five-fold cross-validation as a measure of (computational) cost. To measure effectiveness we use the mean Macro-F1 values over the five-fold cross-validation. Also, we compare only the classification methods that generated the top ten classifiers of Table 6.6. These methods are: BERT, RoBERTa, SVM, LR, NB and RF. We analyzed the cost-effectiveness trade-off of classifiers produced by this classification method both using the original training set and on corresponding training set pre-processed with ROS or c_features when they are applicable.

The displayed training time of a specific method is composed by the time taken by the feature extraction stage, time taken by the data enhancing technique (if applicable) and the model fitting/training time. For this trade-off analysis, we compared both traditional methods that use CPU and more complex models that take advantage of acceleration hardware (GPU). Models that are designed to use GPU, such as BERT and RoBERTa as well as their variations using ROS, were executed in that specific hardware environment. The other methods and their variations, including c_features, were executed in a CPU environment only.

Figures 6.1 and 6.2 present the cost-effectiveness trade-off for the Portuguese and English datasets, respectively. On each of the presented figures, we used dotted lines to separate the image into quadrants to get better perspective and facilitate comparison and analysis. The dotted lines represent the median time (vertical line) and median Macro-F1 (horizontal line) for a specific dataset. If, for a dataset d, a classification method m stands on the left side of the vertical dotted line, it means that m is "faster"than the median of the covered methods on d. If m is above the horizontal dotted line, it means that m has a better effectiveness (Macro-F1) than the median of the covered methods on d. So the first quadrant contains the best cost-effective methods, i.e, they are methods that present the best Macro-F1 and are also the most computationally inexpensive (fastest). Observing the quadrants in a clockwise manner, on the second quadrant stand the methods that are effective (high Macro-F1), but they are slower than the median. The third quadrant is composed by the methods that are faster than the median, but are ineffective. And finally, the last quadrant represents the overall worst methods, that have both effectiveness and efficiency worse than the median for a dataset.

Looking from both time and effectiveness stand point, it is interesting to see the performance of LR with c_features. This specific method is in the first quadrant for all datasets, meaning that it has better prediction power and it is less computational expensive

than the median of the covered methods. This points to the fact that LR + c_features can be considered the best overall cost-effective throughout all the datasets. Not only that, but LR + ROS or LR without any sort of data enhancing proved to be on the first quadrant for most of the datasets, proving to be great cost-effective choices. It is also interesting that LR training time is not much affected by its use with c_features or ROS.

We have noticed (in Section 6.3) that SVM when used with c_features produces more effective classifiers than when not using c_features, specially in the three most difficult datasets. However, in the case of SVM, c_features affect considerably the SVM training time. As show Figures 6.1 and 6.2 SVM with c_features appears int the second quadrant of all datasets, with the exception PT_OFFCOM.

In the PT_OFFCOM dataset we can see the traditional methods dominate, as well as their combination with c_features. The best overall method, in terms of cost-effectiveness, is the Naive-Bayes classifier using with c_features; also SVM and LR combined with that particular data enhancing technique occupy the first quadrant. For that same dataset, we can see BERT and RoBERTa occupy the last quadrant, but we can also notice their effectiveness increase with the Random Oversampling technique at the cost of increasing training time.

As for the PT_HS and PT_RACISM we can see the the best overall methods consist of the transformers and LR combined with both data enhancing techniques analyzed here (c_features and ROS). For the first dataset, BERT, RoBERTa and their variations are on the cusp of the median time dotted line in the first quadrant so they can be considered the best methods. The LR with all its variations may not present Macro-F1 as high as the transformers, but on the other hand they are faster. For the other dataset, BERT is the best cost-effective method, followed by the data enhanced LR variations.

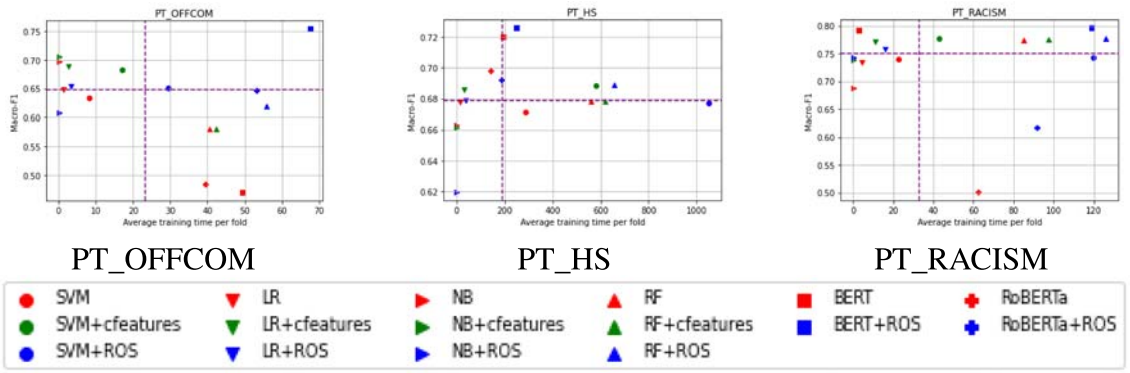

Figura 6.1: *Cost vs. Macro-F1 trade-off for the Portuguese datasets.*

For the English datasets, we can see that for EN_HS_3 and EN_HS_Z the first quadrant is occupied by traditional methods with data enhancing techniques, and BERT

also occupy the first quadrant with the best overall Macro-F1 but barely at left of the time median dotted line.

For EN_GIBERT, the second largest datasets and with the highest class skewness, the use of the oversampling technique vastly increases the training time while not improving the classifier's effectiveness. The exception of that statement is RoBERTa, that has its ROS variation as the best Macro-F1 but on the second quadrant, as RoBERTa with no data enhancing techniques is slightly less effectively but is on the first quadrant, left of the median time line.

The transformers method as well as their variations are on the second and fourth quadrant in the largest dataset EN_GOLBECK. RoBERTa and its oversampling variation present the best Macro-F1 for this dataset, but they are computationally more expensive. The LR classifier combined with c_features is the best cost-effective method, presenting competitive Macro-F1, and taking significantly less training time then RoBERTa.

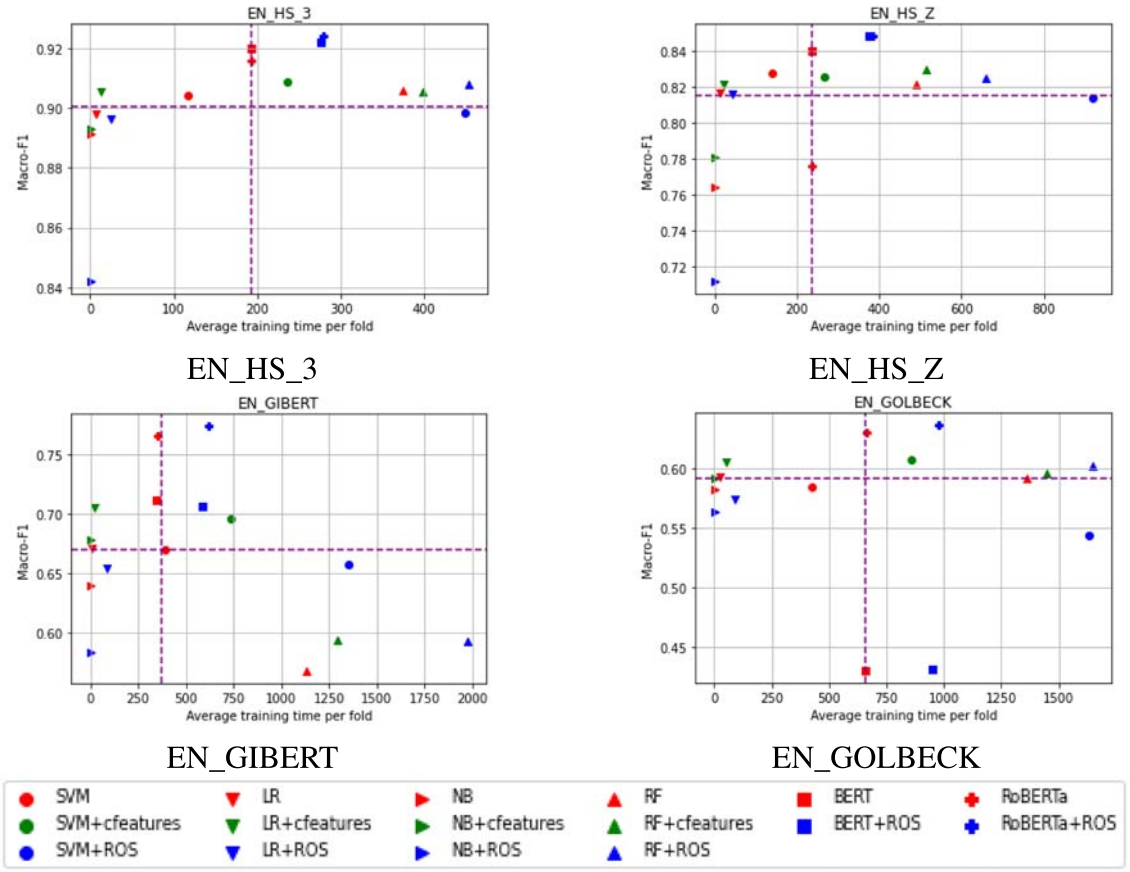

Figura 6.2: *Cost vs. Macro-F1 trade-off for the English datasets.*

Despite not always presenting the best overall results, traditional methods, combined or not with data enhancing techniques, proved to be a competitive choice for cost-effective classifiers. The LR + c_features is considered a consistent good cost-effective method, given that it occupied the first quadrant for all seven datasets. LR alone

or with the combination with ROS are also good cost-effective choices since at least one of them occupy the first quadrant most of the times.

It is important to highlight that LR as well as the other traditional methods covered here, do not make use of GPUs. Thus, its fast execution can be obtained in a cheaper hardware. On the other hand, most of transformer models require hardware acceleration (GPU), removing their applicability in most low-resource environment and have a slower inference time, making them unsuitable for situations that require real-time response time [48]. In a real-world scenario where model retraining is necessary, due to the availability of new labeled training data, traditional methods can still be useful as our cost-effectiveness analysis shows. Besides performing competitively against state-of-the-art complex transformer models, the traditional methods are faster to train.

Tabela 6.4: *Macro-F1 results for the oversampling techniques datasets of English text.*

EN_HS_3							
	ORIG	**SMOTE**	**ADASYN**	**BOS**	**SVMSMOTE**	**KSMOTE**	**ROS**
Linear SVM	0.9043	0.8572	0.8346	0.8388	0.8328	0.8620	0.8982
NB	0.8912	0.8722	0.8357	0.8363	0.8470	0.8716	0.8420
LR	0.8989	0.8541	0.8318	0.8424	0.8497	0.8591	0.8964
RF	0.9076	0.8087	0.7936	0.8125	0.8386	0.8292	0.9080
MLP	0.8881	0.8805	0.8569	0.8749	0.8797	0.8786	0.8923
CNN	0.4266	0.3623	0.4279	0.4279	0.4472	0.4278	0.3393
BiLSTM	0.4267	0.2929	0.3821	0.3460	0.3374	0.3821	0.3376
LSTM [6]	0.7153	0.8747	0.8776	0.8751	-	0.8776	0.8890
BiLSTM [2]	0.8857	0.8794	0.8778	0.8772	-	0.8767	0.8916
BERT	0.9200	-	-	-	-	-	0.9222
RoBERTa	0.916	-	-	-	-	-	0.924
EN_HS_Z							
	ORIG	**SMOTE**	**ADASYN**	**BOS**	**SVMSMOTE**	**KSMOTE**	**ROS**
Linear SVM	0.8274	0.6969	0.6689	0.6708	0.6707	0.6729	0.8135
NB	0.7638	0.7669	0.7422	0.7476	0.7293	0.7438	0.7118
LR	0.8168	0.7581	0.6892	0.6925	0.7365	0.7242	0.8159
RF	0.8210	0.6936	0.6783	0.6970	0.7176	0.6879	0.8248
MLP	0.7899	0.7500	0.7399	0.7402	0.7595	0.7462	0.7778
CNN	0.4580	0.4171	0.4630	0.4622	0.4629	0.4637	0.3284
BiLSTM	0.4580	0.4046	0.4090	0.4059	0.3933	0.3350	0.3941
LSTM [6]	0.6319	0.6999	0.6969	0.7012	-	0.7142	0.6865
BiLSTM [2]	0.7095	0.7163	0.7057	0.7141	-	0.7082	0.6847
BERT	0.8401	-	-	-	-	-	0.848
RoBERTa	0.776	-	-	-	-	-	0.848
EN_GIBERT							
	ORIG	**SMOTE**	**ADASYN**	**BOS**	**SVMSMOTE**	**KSMOTE**	**ROS**
Linear SVM	0.6723	0.5395	0.6562	0.5423	0.6565	0.6473	0.6573
NB	0.6786	0.5902	0.5773	0.6016	0.5480	0.5770	0.5835
LR	0.6721	0.5426	0.6681	0.5480	0.6680	0.6142	0.6541
RF	0.5588	0.5021	0.5465	0.5176	0.5832	0.5423	0.5925
MLP	0.5474	0.5552	0.5485	0.5460	0.5462	0.5474	0.6339
CNN	0.4700	0.4679	0.4671	0.3942	0.4688	0.4672	0.3565
BiLSTM	0.4700	0.3058	0.3931	0.3116	0.3963	0.3886	0.3785
LSTM [6]	0.5636	0.6052	0.5991	0.6095	-	0.6135	0.6564
BiLSTM [2]	0.6224	0.6064	0.5985	0.6091	-	0.6052	0.6435
BERT	0.712	-	-	-	-	-	0.706
RoBERTa	0.766	-	-	-	-	-	0.774
EN_GOLBECK							
	ORIG	**SMOTE**	**ADASYN**	**BOS**	**SVMSMOTE**	**KSMOTE**	**ROS**
Linear SVM	0.5890	0.5395	0.5518	0.5379	0.5379	0.5395	0.5432
NB	0.5819	0.5882	0.5496	0.5797	0.5800	0.5552	0.5637
LR	0.5926	0.5414	0.5671	0.5437	0.5752	0.5652	0.6034
RF	0.5920	0.5311	0.5992	0.5336	0.6025	0.5940	0.6034
MLP	0.5798	0.5612	0.5612	0.5584	0.5647	0.5658	0.5856
CNN	0.4255	0.4258	0.3825	0.3380	0.3377	0.3385	0.2940
BiLSTM	0.4252	0.3816	0.3377	0.3377	0.3381	0.3379	0.3386
LSTM [6]	0.5627	0.5631	0.5606	0.5660	-	0.5644	0.5680
BiLSTM [2]	0.5623	0.5612	0.5615	0.5571	-	0.5628	0.5728
BERT	0.4300	-	-	-	-	-	0.4307
RoBERTa	0.6300	-		-	-	-	0.6360

Tabela 6.5: *Average rank positions for the classifier trained in the original datasets - m_o (column ORIG) using a classification method m (corresponding to a line of the table) and the classifier generated by m, trained in the corresponding datasets expanded wit c_features - m_c (column c_features). Results in bold represent the best classification method of the pair.*

	ORIG	**c_features**
Linear SVM	1.85	**1.14**
NB	1.71	**1.28**
LR	2	**1**
RF	1.71	**1.28**
MLP	1.85	**1.14**
CNN	1.92	**1.07**
BiLSTM	1.64	**1.35**

Tabela 6.6: *Average ranking for all the best remaining classification models.*

Method	Avg. Rank.
SVM + c_features	3.8571
BERT + ROS	4.2143
LR + c_features	5.0000
RoBERTa + ROS	5.2143
LR + ROS	6.7143
SVM	7.4286
RF + ROS	7.8571
NB + c_features	8.2857
RF + c_features	9.2857
NB	9.7143
MLP + c_features	10.2857
MLP + ROS	11.1429
BiLSTM [2] + ROS	12.1429
LSTM [6] + ROS	12.8571
BiLSTM [2] + RUS	14.4286
LSTM [6] + KSMOTE	15.3571
LSTM [6] + RUS	15.5714
LSTM [6] + BOS	15.8571
LSTM [6] + SMOTE	17.4286
LSTM [6] + ADASYN	18.0714
CNN + c_features	21.0714
BiLSTM + c_features	22.3571
CNN + ADASYN	22.5714
BiLSTM	23.2857

Tabela 6.7: *Macro-F1 results for the PT_OFFCOM dataset using a combination of resampling (ROS or RUS) with c_features.*

	ORIG	***c_features***	**ROS + *c_features***	**RUS + *c_features***
Linear SVM	0.6333	0.6829	0.6749	0.6644
NB	0.6975	0.7057	0.7046	0.7091
LR	0.6487	0.6883	0.6802	0.6633
RF	0.5761	0.5798	0.5577	0.5832
MLP	0.5990	0.6682	0.6391	0.6480
CNN	0.4458	0.4475	0.4458	0.4458
BiLSTM	0.4458	0.4475	0.4458	0.4458

CAPÍTULO 7

Conclusions and Future Work

Hate speech is characterized by a discriminating and offensive form of communication that, with the advent of social networks, has been widely disseminated through the Internet. Due to the massive amount of information generated on the Internet, it is impossible for the detection of such speech to be done manually, so it is necessary to use computational techniques for this, more specifically the use of Machine Learning and text classification algorithms.

In this work we performed a thorough and rigorous comparative study of traditional classification methods and diverse neural network methods, including two recent transformer architectures, for the problem of hate speech detection. More specifically, we compared eleven classification methods where seven of them were based on the Vector Space Model (VSM) (SVM, NB, LR, RF, MLP, CNN and BiLSTM), two Deep Learning methods based on word embeddings (LSTM [6], and BiLSTM [2]) and two transformer based methods (BERT and RoBERTa). These methods were compared using seven datasets, three of them are formed by Portuguese messages and the rest composed by English text. Our comparative study focused on detecting hate speech using only the text of the messages. There are other sources of information that are useful for this task, like the author of the message, people who liked the message, etc., but our intent was to evaluate the methods' performance in the more difficult situation where only the text is available. Our study is rigorous in the sense of using appropriate experimentation, including cross-validation, suitable evaluation measure[1] and statistical test.

By means of our first research question (RQ1) we investigated how effective are the eleven methods over the seven datasets. Our results show that BERT, a transformer based architecture is the great winner, followed by LR, SVM, RoBERTa and RF. Although BERT is the best method, it is not as stable as LR and SVM in the sense that its rank positions in individual datasets present variation greater than that for LR and SVM. The instability of RoBERTa, the other transformer, is even greater.

[1] We used the Macro-F1 measure which is appropriate when the datasets present the class imbalance problem, which usually is the case for hate speech datasets.

Class imbalance occurs in all the datasets analyzed and we showed that it affects the classification of hate speech and non-hate speech text. Thus, we investigated if resampling techniques (when answering RQ2) and an adaptation of the c_features expansion technique (when answering RQ3) could enhance the classification methods. With a set of extensive experiments comparing classifiers trained with the original dataset with classifiers trained on resampled data, we found that sophisticated oversampling techniques did not help most of the classification methods. However, Random Oversampling (ROS) a simple oversampling technique not only enhanced many VSM based methods, but to our surprise, it also enhanced the transformer methods. Specially, BERT alone figures as the worst method in the PT_OFFCOMM dataset, but when used with ROS it becomes the best method in that dataset. Besides, ROS does not harm BERT with statistical significance in any of the other datasets. Also, we found that our proposed adaptation to the c_features expansion technique was able to enhance all the classification methods based on the VSM.

Finally, we compared the best classifiers for all methods, i.e., those winning the pairwise comparison between the classifier trained over the original data versus one using the same method but trained on a resampled or expanded (with c_features) version of the data (RQ4). We found that that if we consider the mean rank position of the classifiers in the datasets, the one generated by SVM with c_features is the best classifier, followed by BERT using ROS, LR with c_features and RoBERTa with ROS. BERT and RoBERTa figure as the top method for some datasets, but even using ROS they are not so stable as SVM and LR and for some datasets as their rank positions are far from the top. Also, when considering the cost-effectiveness trade-off (RQ6), we found that the LR with c_features is the best one. The training time of LR is the least affected by the use of ROS or c_features and it is above the median line in terms of effectiveness for all datasets. BERT and RoBERTa are too time consuming. SVM training time is much affected by both types of preprocessing: ROS or c_features. In some cases its training time is superior to that of the transformers.

In future work, we intend to further explore some results we obtained in this work. For instance, sample duplication provided by ROS proved to be better than creating artificial documents, however its random sampling may select some “easy to classify” documents to duplicate instead of choosing more interesting documents. Thus we intend to investigate if we can make better choices (than random choice) for documents to be duplicated. This investigation may be useful not only to enhance oversampling but also to enhance the undersampling generated by RUS which is also random.

Since different classification methods combined with different data preprocessing (ROS and c_features) generated the four best classifiers for the seven datasets used, we think that there are many opportunities to explore the use of classifier ensembles for

hate speech detection. The first and obvious one is to investigate if an ensemble (stacking) with LR + c_features, BERT + ROS, SVM + c_features and RoBERTa + ROS could perform better than each component classifier individually. Another possibility is to try explore the combination of ROS with an ensemble of methods which were benefited by ROS. One suggestion is to apply a classification method m to k different training sets generated by k application of ROS over the original training set and obtain k distinct classifiers. These k classifiers are then combined by means of an ensemble.

Referências Bibliográficas

[1] ABRO, S.; SHAIKH, Z. S.; KHAN, S.; MUJTABA, G.; KHAND, Z. H. **Automatic hate speech detection using machine learning: A comparative study**. *Machine Learning*, 10(6), 2020.

[2] AGRAWAL, S.; AWEKAR, A. **Deep learning for detecting cyberbullying across multiple social media platforms**. In: *European conference on information retrieval*, p. 141–153. Springer, 2018.

[3] AKBANI, R.; KWEK, S.; JAPKOWICZ, N. **Applying support vector machines to imbalanced datasets**. In: *European conference on machine learning*, p. 39–50. Springer, 2004.

[4] ALATAWI, H. S.; ALHOTHALI, A. M.; MORIA, K. M. **Detecting white supremacist hate speech using domain specific word embedding with deep learning and bert**. *IEEE Access*, 9:106363–106374, 2021.

[5] ARANGO, A.; PÉREZ, J.; POBLETE, B. **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*, SIGIR'19, p. 45–54, New York, NY, USA, 2019. Association for Computing Machinery.

[6] BADJATIYA, P.; GUPTA, S.; GUPTA, M.; VARMA, V. **Deep Learning for Hate Speech Detection in Tweets**. In: *Proceedings of the 26th International Conference on World Wide Web Companion*, WWW '17 Companion, p. 759–760, Republic and Canton of Geneva, Switzerland, 2017. International World Wide Web Conferences Steering Committee. event-place: Perth, Australia.

[7] BENAVOLI, A.; CORANI, G.; MANGILI, F. **Should we really use post-hoc tests based on mean-ranks?** *The Journal of Machine Learning Research*, 17(1):152–161, 2016.

[8] BIERE, S.; BHULAI, S.; ANALYTICS, M. B. **Hate speech detection using natural language processing techniques**. *Master Business AnalyticsDepartment of Mathematics Faculty of Science*, 2018.

[9] BURNAP, P.; WILLIAMS, M. L. **Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making**. *Policy & internet*, 7(2):223–242, 2015.

[10] CHAWLA, N. V.; BOWYER, K. W.; HALL, L. O.; KEGELMEYER, W. P. **Smote: Synthetic minority over-sampling technique**. *Journal of Artificial Intelligence Research*, 16:321–357, Jun 2002.

[11] DA SILVA, R. C. C.; FERNANDES, D. S. A.; FERNANDES, M. G. C. **Classificaçao de mensagens em lingua portuguesa com traços de racismo no twitter**. *Revista de Sistemas de Informaçao da FSMA*, (23):2–9, 2019.

[12] DAVIDSON, T.; WARMSLEY, D.; MACY, M.; WEBER, I. **Automated hate speech detection and the problem of offensive language**. In: *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, p. 512–515, 2017.

[13] DE GIBERT, O.; PEREZ, N.; GARCÍA-PABLOS, A.; CUADROS, M. **Hate speech dataset from a white supremacy forum**. In: *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, p. 11–20, Brussels, Belgium, Oct. 2018. Association for Computational Linguistics.

[14] DE PELLE, R. P.; MOREIRA, V. P. **Offensive comments in the brazilian web: a dataset and baseline results**. In: *Anais do VI Brazilian Workshop on Social Network Analysis and Mining*, Porto Alegre, RS, Brasil, 2017. SBC.

[15] DEMŠAR, J. **Statistical comparisons of classifiers over multiple data sets**. *J. Mach. Learn. Res.*, 7:1–30, Dec. 2006.

[16] DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. **Bert: Pre-training of deep bidirectional transformers for language understanding**, 2019.

[17] DOUZAS, G.; BACAO, F.; LAST, F. **Improving imbalanced learning through a heuristic oversampling method based on k-means and smote**. *Information Sciences*, 465:1–20, Oct 2018.

[18] FIGUEIREDO, F.; ROCHA, L.; COUTO, T.; SALLES, T.; GONÇALVES, M. A.; MEIRA JR, W. **Word co-occurrence features for text classification**. *Information Systems*, 36(5):843–858, 2011.

[19] FORTUNA, P.; NUNES, S. **A Survey on Automatic Detection of Hate Speech in Text**. *ACM Comput. Surv.*, 51(4):85:1–85:30, July 2018.

[20] FORTUNA, P.; ROCHA DA SILVA, J.; SOLER-COMPANY, J.; WANNER, L.; NUNES, S. **A hierarchically-labeled Portuguese hate speech dataset**. In: *Proceedings of the Third Workshop on Abusive Language Online*, p. 94–104, Florence, Italy, Aug. 2019. Association for Computational Linguistics.

[21] FORTUNA, P. C. T. **Automatic detection of hate speech in text: an overview of the topic and dataset annotation with hierarchical classes**. 2017.

[22] FRIEDMAN, M. **A comparison of alternative tests of significance for the problem of m rankings**. *The Annals of Mathematical Statistics*, 11(1):86–92, 1940.

[23] GARCIA, S.; HERRERA, F. **An extension on"statistical comparisons of classifiers over multiple data sets"for all pairwise comparisons**. *Journal of machine learning research*, 9(Dec):2677–2694, 2008.

[24] GLAZKOVA, A. **A comparison of synthetic oversampling methods for multi-class text classification**. *arXiv preprint arXiv:2008.04636*, 2020.

[25] GOLBECK, J.; ASHKTORAB, Z.; BANJO, R. O.; BERLINGER, A.; BHAGWAN, S.; BUNTAIN, C.; CHEAKALOS, P.; GELLER, A. A.; GNANASEKARAN, R. K.; GUNASEKARAN, R. R.; OTHERS. **A large labeled corpus for online harassment research**. In: *Proceedings of the 2017 ACM on web science conference*, p. 229–233, 2017.

[26] HAN, H.; WANG, W.-Y.; MAO, B.-H. **Borderline-smote: A new over-sampling method in imbalanced data sets learning**. In: Huang, D.-S.; Zhang, X.-P.; Huang, G.-B., editors, *Advances in Intelligent Computing*, p. 878–887, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[27] HARTMANN, N.; FONSECA, E.; SHULBY, C.; TREVISO, M.; RODRIGUES, J.; ALUISIO, S. **Portuguese word embeddings: Evaluating on word analogies and natural language tasks**, 2017.

[28] HE, H.; BAI, Y.; GARCIA, E. A.; LI, S. **Adasyn: Adaptive synthetic sampling approach for imbalanced learning**. In: *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, p. 1322–1328. IEEE, 2008.

[29] KIM, Y. **Convolutional neural networks for sentence classification**. *arXiv preprint arXiv:1408.5882*, 2014.

[30] KUMAR, R.; LAHIRI, B.; OJHA, A. K. **Aggressive and offensive language identification in hindi, bangla, and english: A comparative study**. *SN Computer Science*, 2(1):1–20, 2021.

[31] LI, C.; LIU, S. **A comparative study of the class imbalance problem in twitter spam detection**. *Concurrency and Computation: Practice and Experience*, 30(5):e4281, 2018.

[32] LIU, Y.; OTT, M.; GOYAL, N.; DU, J.; JOSHI, M.; CHEN, D.; LEVY, O.; LEWIS, M.; ZETTLEMOYER, L.; STOYANOV, V. **Roberta: A robustly optimized bert pretraining approach**, 2019.

[33] MACAVANEY, S.; YAO, H.-R.; YANG, E.; RUSSELL, K.; GOHARIAN, N.; FRIEDER, O. **Hate speech detection: Challenges and solutions**. *PloS one*, 14(8):e0221152, 2019.

[34] MADUKWE, K. J.; GAO, X.; XUE, B. **A ga-based approach to fine-tuning bert for hate speech detection**. In: *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, p. 2821–2828. IEEE, 2020.

[35] MITCHELL, T. M. **Machine learning, International Edition**. McGraw-Hill Series in Computer Science. McGraw-Hill, 1997.

[36] MOZAFARI, M.; FARAHBAKHSH, R.; CRESPI, N. **A bert-based transfer learning approach for hate speech detection in online social media**, 2019.

[37] MUTANGA, R.; NAICKER, N.; OLUGBARA, O. O. **Hate speech detection in twitter using transformer methods**. *International Journal of Advanced Computer Science and Applications*, 11(01), 2020.

[38] NGUYEN, H. M.; COOPER, E. W.; KAMEI, K. **Borderline over-sampling for imbalanced data classification**. *International Journal of Knowledge Engineering and Soft Data Paradigms*, 3(1):4–21, 2011.

[39] NOBATA, C.; TETREAULT, J.; THOMAS, A.; MEHDAD, Y.; CHANG, Y. **Abusive language detection in online user content**. In: *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, p. 145–153, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee.

[40] PADURARIU, C.; BREABAN, M. E. **Dealing with data imbalance in text classification**. *Procedia Computer Science*, 159:736–745, 2019.

[41] PELLE, R.; ALCÂNTARA, C.; MOREIRA, V. P. **A classifier ensemble for offensive text detection**. In: *Proceedings of the 24th Brazilian Symposium on Multimedia and the Web*, WebMedia '18, p. 237–243, New York, NY, USA, 2018. ACM. event-place: Salvador, BA, Brazil.

[42] PENNINGTON, J.; SOCHER, R.; MANNING, C. **Glove: Global vectors for word representation**. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, p. 1532–1543, Doha, Qatar, Oct. 2014. Association for Computational Linguistics.

[43] PITSILIS, G. K.; RAMAMPIARO, H.; LANGSETH, H. **Detecting Offensive Language in Tweets Using Deep Learning**. *Applied Intelligence*, 48(12):4730–4742, Dec. 2018. arXiv: 1801.04433.

[44] PUTRI, T.; SRIADHI, S.; SARI, R.; RAHMADANI, R.; HUTAHAEAN, H. **A comparison of classification algorithms for hate speech detection**. In: *IOP Conference Series: Materials Science and Engineering*, volume 830, p. 032006. IOP Publishing, 2020.

[45] RANI, P.; SURYAWANSHI, S.; GOSWAMI, K.; CHAKRAVARTHI, B. R.; FRANSEN, T.; MCCRAE, J. P. **A comparative study of different state-of-the-art hate speech detection methods in Hindi-English code-mixed data**. In: *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, p. 42–48, Marseille, France, May 2020. European Language Resources Association (ELRA).

[46] ROSS, B.; RIST, M.; CARBONELL, G.; CABRERA, B.; KUROWSKY, N.; WOJATZKI, M. **Measuring the Reliability of Hate Speech Annotations: The Case of the European Refugee Crisis**. In: Beißwenger, M.; Wojatzki, M.; Zesch, T., editors, *Proceedings of NLP4CMC III: 3rd Workshop on Natural Language Processing for Computer-Mediated Communication*, volume 17 de **Bochumer Linguistische Arbeitsberichte**, p. 6–9, Bochum, sep 2016.

[47] RUPAPARA, V.; RUSTAM, F.; SHAHZAD, H. F.; MEHMOOD, A.; ASHRAF, I.; CHOI, G. S. **Impact of smote on imbalanced text features for toxic comments classification using rvvc model**. *IEEE Access*, 2021.

[48] SAJJAD, H.; DALVI, F.; DURRANI, N.; NAKOV, P. **Poor man's bert: Smaller and faster transformer models**. *arXiv e-prints*, p. arXiv–2004, 2020.

[49] SOUZA, F.; NOGUEIRA, R.; LOTUFO, R. **BERTimbau: pretrained BERT models for Brazilian Portuguese**. In: *9th Brazilian Conference on Intelligent Systems, BRACIS, Rio Grande do Sul, Brazil, October 20-23 (to appear)*, 2020.

[50] SUN, A.; LIM, E.-P.; LIU, Y. **On strategies for imbalanced text classification using svm: A comparative study**. *Decision Support Systems*, 48(1):191–201, 2009.

[51] TULKENS, S.; HILTE, L.; LODEWYCKX, E.; VERHOEVEN, B.; DAELEMANS, W. **A dictionary-based approach to racism detection in Dutch social media**. In: *Proceedings of the LREC 2016 Workshop on Text Analytics for Cybersecurity and Online Safety (TA-COS)*. European Language Resources Association (ELRA), 2016.

[52] WANG, J.; WANG, Z.; ZHANG, D.; YAN, J. **Combining knowledge with deep convolutional neural networks for short text classification.** In: *IJCAI*, volume 350, 2017.

[53] WARDHAUGH, R.; FULLER, J. M. **An introduction to sociolinguistics**. John Wiley & Sons, 2021.

[54] WASEEM, Z. **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*, p. 138–142, Austin, Texas, Nov. 2016. Association for Computational Linguistics.

[55] WASEEM, Z.; HOVY, D. **Hateful symbols or hateful people? predictive features for hate speech detection on twitter**. In: *Proceedings of the NAACL Student Research Workshop*, p. 88–93, San Diego, California, June 2016. Association for Computational Linguistics.

[56] YOUNG, J. C.; RUSLI, A.; OTHERS. **A comparison of supervised text classification and resampling techniques for user feedback in bahasa indonesia**. In: *2020 Fifth International Conference on Informatics and Computing (ICIC)*, p. 1–6. IEEE, 2020.

[57] ZHANG, Z.; ROBINSON, D.; TEPPER, J. **Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network**. In: Gangemi, A.; Navigli, R.; Vidal, M.-E.; Hitzler, P.; Troncy, R.; Hollink, L.; Tordai, A.; Alam, M., editors, *The Semantic Web*, Lecture Notes in Computer Science, p. 745–760. Springer International Publishing, 2018.

[58] ZHENG, Z.; CAI, Y.; LI, Y. **Oversampling method for imbalanced classification**. *Computing and Informatics*, 34(5):1017–1037, 2015.

APÊNDICE A

Appendix

A.1 Macro-F1 values for c_features

Tabela A.1: *Macro-F1 for the c_features method on the Portuguese datasets.*

	PT_OFFCOM		PT_HS		PT_RACISM	
	ORIG	**c_features**	**ORIG**	**c_features**	**ORIG**	**c_features**
Linear SVM	0.6333	0.6829	0.6709	0.6884	0.7396	0.7763
NB	0.6975	0.7057	0.6631	0.6615	0.6868	0.7387
LR	0.6487	0.6883	0.6775	0.6855	0.7337	0.7704
RF	0.5761	0.5798	0.6762	0.6784	0.7859	0.7738
MLP	0.5990	0.6682	0.6448	0.6651	0.7092	0.7360
CNN	0.4458	0.4458	0.4064	0.6275	0.4685	0.5112
BiLSTM	0.4458	0.4458	0.4064	0.3854	0.4532	0.4537

Tabela A.2: *Macro-F1 for the c_features method on the English datasets.*

	EN_HS_3		EN_HS_Z		EN_GIBERT		EN_GOLBECK	
	ORIG	**c_features**	**ORIG**	**c_features**	**ORIG**	**c_features**	**ORIG**	**c_features**
Linear SVM	0.9043	0.9086	0.8274	0.8248	0.6723	0.6741	0.5890	0.6167
NB	0.8912	0.8931	0.7638	0.7797	0.6786	0.6389	0.5819	0.5906
LR	0.8989	0.9058	0.8168	0.8206	0.6721	0.6811	0.5926	0.6028
RF	0.9076	0.9085	0.8210	0.8267	0.5588	0.5988	0.5920	0.5956
MLP	0.8881	0.8792	0.7899	0.7953	0.5474	0.6046	0.5798	0.5959
CNN	0.4266	0.4366	0.458	0.4607	0.47	0.4773	0.4255	0.4713
BiLSTM	0.4267	0.4544	0.458	0.4744	0.47	0.5426	0.4252	0.4285

A.2 Macro-F1 values for Random Undersampling

Tabela A.3: *Macro-F1 results for all datasets using an undersampling strategy.*

	PT_OFFCOM		PT_HS		PT_RACISM		EN_HS_3		EN_HS_Z		EN_GIBERT		EN_GOLBECK	
	ORIG	RUS	ORIG	RUS	ORIG	RUS	ORIG	RUS	ORIG	RUS	ORIG	RUS	ORIG	RUS
Linear SVM	0.6333	0.5466	0.6709	0.6838	0.7396	0.7515	0.9043	0.8984	0.8274	0.8021	0.6723	0.6355	0.5890	0.5791
NB	0.6975	0.6266	0.6631	0.6137	0.6868	0.6432	0.8912	0.8178	0.7638	0.6981	0.6786	0.5176	0.5819	0.5435
LR	0.6487	0.5598	0.6775	0.6816	0.7337	0.7512	0.8989	0.8903	0.8168	0.8037	0.6721	0.6361	0.5926	0.5733
RF	0.5761	0.4552	0.6762	0.6944	0.7859	0.7464	0.9076	0.8976	0.821	0.8078	0.5588	0.5471	0.5920	0.595
MLP	0.5990	0.5575	0.6448	0.6465	0.7092	0.7257	0.8881	0.8864	0.7899	0.7759	0.5474	0.6016	0.5798	0.578
CNN	0.4458	0.4458	0.4064	0.4071	0.4685	0.4532	0.4266	0.4267	0.458	0.458	0.47	0.4698	0.4255	0.4252
BiLSTM	0.4458	0.4458	0.4064	0.4064	0.4532	0.4532	0.4267	0.4267	0.458	0.458	0.47	0.4666	0.4252	0.4252
LSTM [5]	0.6445	0.609	0.6091	0.6332	0.6813	0.6931	0.7153	0.882	0.6319	0.7016	0.5636	0.6108	0.5627	0.5525
BiLSTM [5]	0.6188	0.6222	0.6389	0.6391	0.6988	0.7026	0.8857	0.8811	0.7095	0.7038	0.6224	0.6252	0.5623	0.548
BERT	0.47	0.45	0.72	0.41	0.7920	0.45	0.92	0.43	0.8401	0.46	0.712	0.47	0.43	0.43
RoBERTa	0.484	0.548	0.698	0.692	0.501	0.552	0.916	0.874	0.776	0.798	0.766	0.714	0.63	0.522