Journal of Computer Science

Original Research Paper

Long Short-Term Memory Model for Classification of English-PtBR Cross-Lingual Hate Speech

[1]Thiago D. Bispo, [2]Hendrik T. Macedo, [3]Flávio de O. Santos, [2]Rafael P. da Silva, [2]Leonardo N. Matos, [2]Bruno O.P. Prado, [2]Gilton J.F. da Silva and [4]Adolfo Guimarães

[1]Instituto Federal de Sergipe, Aracaju, Brazil
[2]Departamento de Computação, Universidade Federal de Sergipe, São Cristóvão, Brazil
[3]Centro de Informática, Universidade Federal de Pernambuco, Recife, Brazil
[4]Departamento de Computação, Universidade Tiradentes, Brazil

Article history
Received: 28-08-2019
Revised: 24-09-2019
Accepted: 31-10-2019

Corresponding Authors
Thiago D. Bispo
Instituto Federal de Sergipe, Aracaju, Brazil
Email: thiago.bispo@ifs.edu.br

Abstract: Automatic and accurate recognition of hate speech is a difficult job. In addition to the inherent ambiguity of the natural language, deep understanding of the linguistic structure is imperative. Usually, discriminatory discourse does not make use of typical expressions and often abuse of sarcasm. Good knowledge of world and assessment of context are thus highly demanded. Several approaches have been proposed for automating hate speech recognition task. Many of them consider a combination of strategies in order to achieve better results: character-based or word-based N-grams, lexical features such as the presence or absence of negative words, classes or expressions indicative of insult, punctuation marks, repetition of letters, the presence of emoji, etc. The solitary use of linguistic features such as POS tagging have shown itself inefficient. The recent usage of neural networks to create a distributed representation of the sentences within a hate speech corpus is a promising path. Unfortunately, providing such a corpus is hard. Except for the English language, hate speech corpora are rarely found. This work proposes a cross-lingual approach to automatically recognize hate speech in Portuguese language, leveraging the knowledge of English corpora. A deep Long Short-Term Memory (LSTM) model has been trained and many different experimentation scenarios were set to deal with embeddings, TFIDF, N-grams, GloVe vocabulary and so on. At the end, a Gradient Boosting Decision Tree (GBDT) was used to improve classification results. We achieved accuracy of up to 70% in the better scenarios. Two important contributions of this work are: (i) An effective approach to deal with the lack of hate speech corpora in the desired language and (ii) a hate speech database in Portuguese to contribute to research community.

Keywords: Hate Speech, Portuguese Language, Deep Learning, (Bi) LSTM, GBDT

Introduction

Internet is increasingly seeing as the main source of information and communication channel leader in the world. Although its undoubted relevance to world progress, it is also responsible for the spread of cyber hate as an extension of hatred and intolerance of human beings. Facebook and Twitter are recently taking action to combat such spread[1,2]. Other initiatives also exist. Nobata *et al*. (2016), for instance, provided a model which has been adopted by Yahoo! as a mechanism for detecting abusive comments. Although the number of works and approaches that deal with the problem of classifying hate speech in English is well-known (Schmidt and Wiegand, 2017), most do not make their publicly labeled data available. For Portuguese language, the scenario is even worse: until recently, only the work of Fortuna (2017) seems to effectively contribute.

Since Natural Language Processing activities are highly dependent of the language of corpora, focusing on the creation of databases from scratch and detection of abusive speech in Portuguese is central. Moreover, Economic reports has shown a huge growth in the

[1]https://blog.twitter.com/official/en_us/topics/company/2017/safetycalendar.html
[2]2https://newsroom.fb.com/news/2017/06/ hard-questions-hate-speech/

amount of people who use Portuguese on the Internet and concerned number of victims of such abuse[3].

Detecting hate speech is still a challenge for two main reasons. Firstly, it requires an advanced level of understanding of the structure and semantics of comments, involving detection of user intent and the presence of irony/sarcasm, factors that encompass, among other things, world knowledge. That is, it involves the computer representing common concepts that objectively explain and delimit real-world elements. Second, the language and artifices used to express or mask comments such as hate speech are not static, varying considerably even between regions of the same country, such as slang and language vices.

The study of cross-lingual models stands out in such scenario for being able to leverage the use of well established *corpora*, available in other languages, minimizing the need to perform the costly task of database creation and labeling. Da Silva *et al*. (2018) has shown that this approach is a promising path towards the solution to the problem.

A research hypothesis we raise is that LSTM models can be trained as cross-lingual models from automatically translated datasets and produce good test results from Portuguese datasets. This can lead to a minimization of the impact on the task of detecting hate speech.

The main goal of this work is to train a cross-lingual LSTM model of hate speech classification in Portuguese using a training set written in English. Specific goals are:

1. Construct and label a hate speech dataset in Portuguese and make it publicly available to the concerned scientific community
2. Validate the trained template with the dataset created in this work
3. Define promising preprocessing and vectoring techniques within the context of hate speech detection with a cross-lingual LSTM model using dataset in English as a training basis

At first, due to the scarcity of datasets in the target language (Fortuna, 2017), we focused on the creation of a *dataset* of speeches and labeled them with the help of volunteers at Internet. Next, we've used an Automatic Translation System (STA) to translate the dataset (Waseem and Hovy, 2016), containing more than 16,000 tweets labeled as sexist, racist or neither. Then, using the Long Short-Term Memory (LSTM) model (Goodfellow *et al*., 2016) and based on the work of Badjatiya *et al*. (2017), we have tried different approaches organized in 24 scenarios that demonstrated the performance of the model when configured and trained with distinct datasets for binary classification: presence or absence of hate speech. Finally, we investigate the model's cross-lingual ability in the task of detecting hate speech using, in some of the created scenarios, the dataset constructed as a basis for validation.

Two important contributions of this work are: (i) proposal of an alternative research approach to attack the problem based on the translation of *corpora* and (ii) provision of a dataset of hate speech in Portuguese to the community.

The rest of the paper is organized as follows. Section 2 presents the main concepts and techniques related to the experiments in this work. In section 3, we delve deeper into the definition and concepts related to hate speech. We also do a survey of works related to the detection of hate speech. Section 4 describes step-by-step the construction of the created dataset. In section 5, we describe the experiments used to evaluate the model and discuss the results. Section 7 concludes the work.

Theoretical Background

In this section we introduce some building blocks of our approach: Recurrent Neural Networks, including the LSTM model, the *Word Embeddings* concept and the Gradient Boosting Decision Tree.

Recurrent Neural Networks

Recurrent Neural Networks (RNN) are types of neural networks focused on the recognition of information chains, sequential data (Goodfellow *et al*., 2016). Sequences are represented as a set of inputs $x^{(1)}$, ..., $x^{(\tau)}$, in which the input $x^{(t)}$ is the vector in time/step t, with t ranging from 1 to τ. RNNs share weight matrices over several steps in input chain processing, allowing them to learn patterns that arise at different positions in the sequences. This was one of the advances in traditional feedforward neural networks, where weights for each feature of the entry at index t are not shared among each other, requiring that possible positions of the desired patterns in entry chains were identified, a task that is impractical.

Figure 1 illustrates the general representation of a RNN. For a sequence of size τ, each time entry t is mapped to the input t +1 by a f function which operates on the s_t state. Operation Unfold is simply the dismemberment of the RNN in its recurrent form (left side of Figure) to its complete form (right side of Figure). The input of the state s in time t is weighted by the matrix of weights U; its output is multiplied by the matrix V to generate the *output* o. The state value s_t is used as *input* for the state s_{t+1} and weighted by the matrix W. States s_t are treated as the network memory since it retains information of previous occurrences.

[3]3https://epoca.globo.com/tecnologia/experienciasdigitais/noticia/2017/02/ha-um-aumento-sistematico-de-discursode-odio-na-rede-diz-diretor-do-safernet.html

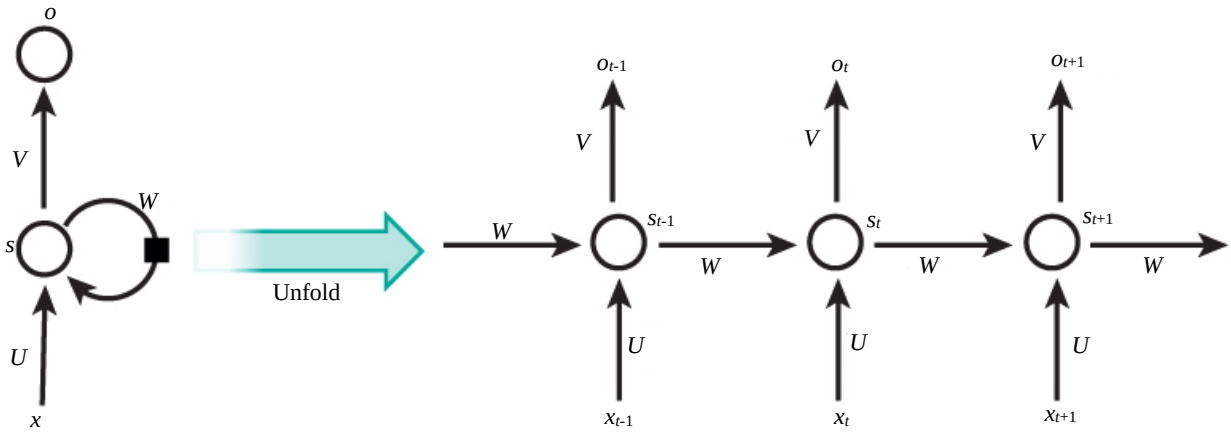


Fig. 1: General representation of an RNN

For the Vanilla RNN, one of the simplest RNN models, the values of s_t and o are computed as:

$$\begin{aligned} s_t &= f\left(Ux_t + Ws_{t-1}\right) \\ o_t &= Vs_t \\ y_t &= soft\max\left(o_t\right) \end{aligned} \tag{1}$$

Function "f" is often nonlinear, such as the hyperbolic tangent (*tanh*) or *ReLu*. Arrays U, W and X are shared among all states. Not all models require the output to be generated for each state, but for the final state.

Gradient descent calculation in *feedforward* is modified to take into account RNNs' parameter sharing and is known as *BackPropagation Through Time* (BPTT). Let the cross-entropy error function defined by the Equation (2):

$$E_t(y_t, \hat{y}_t) = -y_t log(\hat{y}_t) \tag{2}$$

$$E_t(y, \hat{y}) = \sum_t E_t(y_t, \hat{y}_t) \tag{3}$$

$$= -\sum_t y_t log(\hat{y}_t) \tag{4}$$

In which y_t represents the correct value of the feature in time t and $\hat{y}_t$ is the value predicted by the network according to the Equation (1). Total error is the summation over all errors in each state s_t. Once the matrix V is one of the model parameters, the error in function of V is calculated as the sum of the partial derivatives of E_t with respect to V, according to Equation (5):

$$\frac{\partial E}{\partial V} = \sum_t \frac{\partial E_t}{\partial V} \tag{5}$$

In the above formula, the error function depends only on the values of y_t, $\hat{y}_t$ and s_t, making the results of $\frac{\partial E_t}{\partial V}$ calculable from a single matrix multiplication, as we can observe in the Equation (6):

$$\begin{aligned} \frac{\partial E_t}{\partial V} &= \frac{\partial V}{\partial \hat{y}_t}\frac{\partial \hat{y}_t}{\partial V} \\ &= \frac{\partial V}{\partial \hat{y}_t}\frac{\partial \hat{y}_t}{\partial z_t}\frac{\partial z_t}{\partial V}, z_t = Vs_t \end{aligned} \tag{6}$$

On the other hand, the error in function of W and U, since both depend on s_t (which in turn depends on s_{t-1}), involves the application of the chain rule, since we cannot consider s_t as a constant. For this reason, $\frac{\partial E_t}{\partial W}$ is calculated as follows:

$$\frac{\partial E_t}{\partial W} = \sum_{k=0}^{t} \frac{\partial E_t}{\partial \hat{y}_t}\frac{\partial \hat{y}_t}{\partial s_t}\frac{\partial s_t}{\partial s_k}\frac{\partial s_k}{\partial W} \tag{7}$$

Otherwise, the gradient for calculating the error E in time t is propagated recursively to the time $t = 0$, resulting in the so-called *Vanishing Gradient Problem - VGP*, which consists in the degradation of gradient values to zero in a few steps as a consequence of the successive matrix multiplications. Since the gradient on each recurring unit tends to zero, it will boost the gradient of the previous cells to zero as well; and the higher the τ value in the input string is, the greater the network's chances of suffering from this problem.

There are several proposals to avoid the VGP. By far, the most prominent and widely used is the LSTM model (Hochreiter and Schmidhuber, 1997). One of its main characteristics is the inclusion of special units known as gates. These units calculate the weights that connect them in order to avoid degradation of the gradient through manually-chosen or parameterized values (Goodfellow *et al*., 2016).

Like all gated RNNs, LSTMs have the ability to both remember and forget about the previous state when that information is no longer needed. At the time of training, the network has the ability to learn exactly what to forget, mechanism that is executed through the parameters of forget gate. The values of previous state, the current memory and the input are, thus, combined to form the output of the unit (or cell). This is a mechanism that proved quite efficient in learning long dependencies among the terms of a sequence. An LSTM cell is composed of three gates that control different behaviors: *input gate*, *forget gate* and *output gate* (Fig. 2).

The forget gate (Fig. 3) controls the input of cell C_t in order to determine which of the index values of the output vector of the previous cell C_{t-1} will be maintained, through its σ function, which returns values in the range of 0 to 1. W_f and b_f are the weights and bias values for the gateway, respectively. The LSTM cell can also calculate what input information will be stored/updated. As can be seen in Fig. 4, this task is divided in two steps: (1) function σ decides which of the information of C_{t-1} will update (in the *forget gate* this step is responsible for determining what information will be forgotten) and (2) a new candidate input value $(\tilde{C}_t)$ is calculated by the *tanh* function to be further multiplied to the vector resulting from the first step. Note that operations i and $\tilde{C}$ have their own parameters, which can also be learned by the network during training.

After these operations, there is enough information to update the cell state $C_t = f_t * C_t–1 + i_t * \tilde{C}_r$. Two further steps are needed to properly provide the output h_t. At first, the σ function determines what "portions" of the vector/state C_t should be part of the output (value o_t of Fig. 5). Next, the C_t state is submitted to a hyperbolic function (*tanh*) and multiplied by the result of the previous operation.

Word Embeddings

LSTMs (actually, any RNN) can be applied to any problem that requires the recognition of string sequences such as Machine Translation (Liu *et al*., 2014), Sentiment Analysis (Singhal and Bhattacharyya, 2016) and Hate Speech (Badjatiya *et al*., 2017). In fact, any problem involving dealing with sequences can benefit with RNNs, as Human Action Recognition (Jaouedi *et al*., 2019) and Speech Emotion Recognition (Praseetha and Vadivel, 2018). The first one is based on video processing (image sequence) and the second one treat speech (sequence of sounds).

In NLP problems, we need to provide good representation of the sentences as input to the LSTM network. The simplest representation is the one-hot coding. However, enriched representations are required when dealing with domains that highly suffers from ambiguity and context dependence, which is the case of Hate Speech identification.

The so-called word embeddings is currently the best shot the research community have to deal with such challenges. They consist of vector representations capable of capturing the relationship semantics between two words without losing the ability to encode them in different ways (Goodfellow *et al*., 2016). In the space of embeddings, words that often appear in similar contexts are close to each other, constructing a neighborhood of similar words.

Different algorithms were developed for the purpose of generating embeddings. They can be divided into two families of methods (Hartmann *et al*., 2017). The first are those methods that work with the co-occurrence matrix of words, such as GloVe (Pennington *et al*., 2014). The other family are those that work with predictive models (based on the word's neighborhood), such as Word2Vec (Mikolov *et al*., 2013). Hartmann *et al*. (2017) summarize some of the major embeddings generation models:

- The Global Vectors (**GloVe**) - A non-supervised learning algorithm that computes the vectors by analyzing the M matrix of word co-occurrence constructed through the contextual information of the words of the corpus
- **Word2vec** - It has two different training strategies: (i) Continuous Bag-of-Words (CBOW), in which the model attempts to predict the deleted middle word within a word sequence and (ii) Skip-Gram, in which the model predicts the vicinity of one of the word
- **Wang2Vec:** Modification of Word2vec whose purpose is to take into account the sequence order of words in the sentence
- **FastText:** In this architecture, embeddings are associated with character N-grams with the words coded as the combination of these representations. As a consequence, this method attempts to capture morphological information to construct its word embeddings

The Word Embeddings Repository of the Inter-Institutional Nucleus of Computational Linguistics (NILC)[4] contains several publicly available word vectors for free download. They were generated by means of a corpus in Brazilian Portuguese and European Portuguese, using all the models mentioned above and for different dimensions. In this work, for the generation of embeddings using corpora in Portuguese, we adopted the NILC vectors constructed through the 100-dimensional GloVe model, hereafter referred to as GloVe100.

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

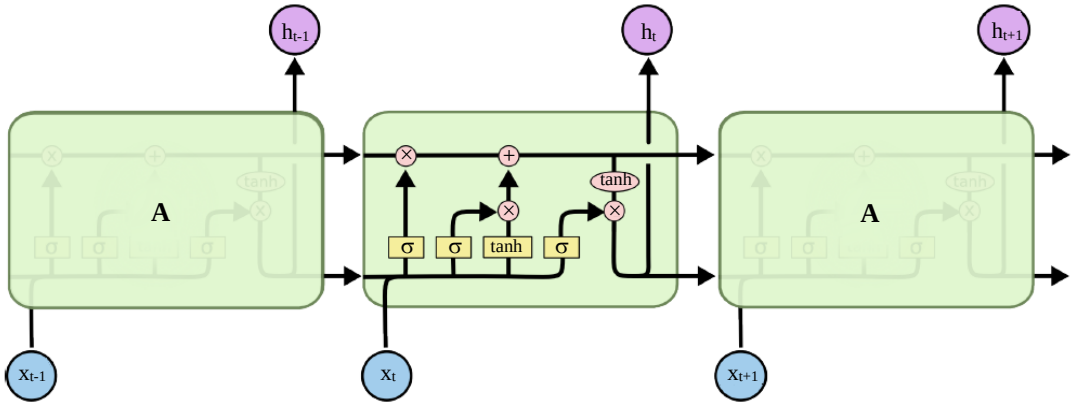
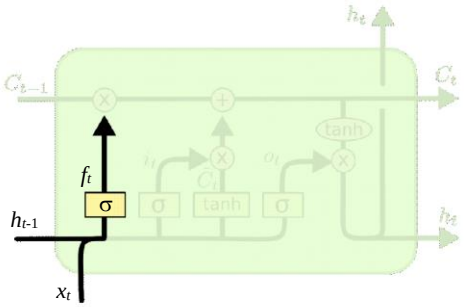
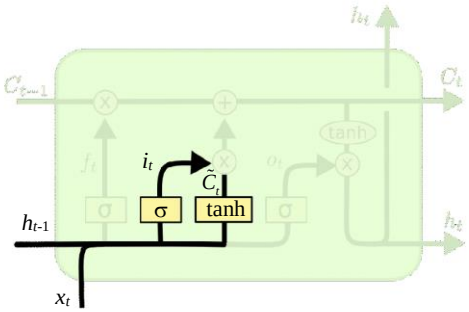


Fig. 2: Structure of an LSTM cell; Source: Christopher Olah blog



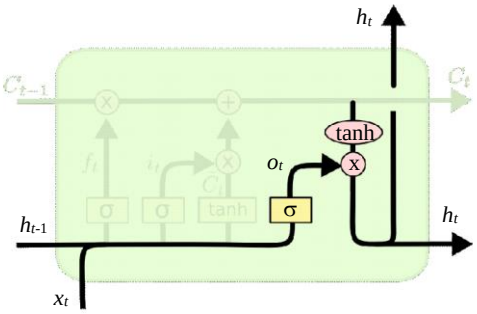
$$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right] + b_f\right)$$

Fig. 3: LSTM: Forget gate; Source: Christopher Olah blog



$$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right)$$
$$\tilde{C}_t = \tanh\left(W_C \cdot \left[h_{t-1}, x_t\right] + b_C\right)$$

Fig. 4: LSTM: Input gate; Source: Christopher Olah blog



$$o_t = \sigma\left(W_o \left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Fig. 5: LSTM: Output Gate

Gradient Boosting Decision Tree (GBDT)

Tree-based sorting or regression methods divide the feature space into recursively sub-areas in which each step selects the most representative characteristic of the input data in the training process (Marsland, 2014). They are called "decision trees" because each sub-area within the feature space is represented by a root node (which contains its most relevant feature) and its leaf nodes, each containing the next relevant characteristics resulting induction process on the tree. One of the most critical decisions in the construction and partitioning of decision trees is the choice of which feature best represents the data on the node being partitioned. One of the most common ways to evaluate the quality of partitioning is to use the concepts of Cross-Entropy and Information Gain.

Popular error functions used for GBDT training are AdaBoost and logistic regression (Friedman, 2002). The latter consists of the cross-entropy function adapted for essemble models. AdaBoost in turn proposes the creation of weights to be applied to each weak learners. Its effect is to allow greater influence to the classifiers with greater accuracy, weakening the relevance of the outputs of those that contribute to decrease the overall performance of the model. Once the global error is calculated, it is propagated to each classifier by adjusting its respective weights.

According to Sutton (2005), "boosting" technique usually benefits classifiers whose method of classification is unstable, drastically reducing the error rate resulting from the classification method: unstable classifiers have a high variance and the boosting decreases its value without increasing the bias. In addition, there is strong evidence that GBDTs are resistant against over-fitting, possibly due to their ability in producing reasonably strong and uncorrelated classifiers.

Detection of hate Speech in Texts

Definition of hate speech is often misinterpreted. To assure detection accuracy, such misinterpretation must be solved. Table 1 presents some currently used definitions for hate speech and their respective sources.

We are concerned with the searching for definitions from both the Internet and authors who have studied hate speech in other ways such as Moura (2016). Both emphasize that any kind of comment that has or is capable of instigating discrimination against a certain group of people should be considered hatespeech. Note that this definition is more comprehensive than that of Facebook (Table 1), which allows the use of humorous and offensive content, making the border of what would be reprehensible more difficult to establish. We believe that this definition is permissive and encourages the dissemination of subtle comments and implicit violence that are equally damaging to victims, as usually the jokes that reinforce stereotypes (such as blondes, gay, fat and certain physical aspects) are. Repetition of jokes of this kind, even without discriminatory intent, shapes the relationship between the group of those who utter them and the target group of victims. In other words, repeating jokes is a way of reinforcing bad attitudes or thoughts. In the scope of this work, thus, we adopt the definition provided by Fortuna (2017), as it includes delicate situations, including humorous situations in general, like jokes. There are a plenty of concepts related to hate speech: hate, cyberbullying, abusive language, discrimination, profanity, toxicity and so on, that are often improperly used. In Table 2, we briefly reproduce proper distinctions made by Fortuna (2017).

Decision trees can be combined to improve its individual predictive power. This method is known as *Tree Essemble* and there are several models based on it. One of the best known is *Random Forest* (Breiman, 2001). In this work we use another model called Gradient Boosting Decision Tree (GBDT) (Friedman, 2002). Like Random Forest (RF), GBDT uses the partitioning of the prediction process through the essemble for the tasks of classification or linear regression. However, while RF generates several trees in parallel and applies majority voting to provide finall prediction, in GBDT, decision trees are sequentially constructed and each of them is focused in correcting the mistakes of the previous one; the trees created in such process are weak learners with a training technique known as boosting (Friedman *et al*., 2001).

As we can see from the definitions provided, abusive language does not necessarily contain the kind of discrimination that characterizes a discourse of hate speech. However, within the scope of this paper, for the sake of simplicity, we adopt the definition that a comment is abusive if it contains some kind of hate speech and otherwise clean.

Table 3 displays the main types of hatespeech and some example of its main targets.

Additionally, Cavalcante Segundo (2016) talks about a type of discourse that contains political intolerance and how much it is present in the world. In this scenario, the hate speech is usually manifested in comments with explicit intent or not to downgrade or offend the opponent. In Brazil, political intolerance is evident in the "petistas" (person who is a member or sympathizer of the PT political party) and "antipetitas" (on the contrary).

Table 1: Examples of definitions of hate speech

Source	Definition
Facebook5	"Contents that attack people based on their race, ethnicity, nationality, religion, gender, gender or gender identity, sexual orientation, disability or illness, whether actual or presumed, are not allowed. However, we allow clear attempts at jokes or satire that do not have the character of threats or attacks. This includes content that many people may find distasteful (e.g., jokes, stand-up comedy, certain lyrics of popular songs, etc.)."
Twitter6"	Hateful conduct: You may not promote violence against or directly attack or threaten other people on the basis of race, ethnicity, national origin, sexual orientation, gender, gender identity, religious affiliation, age, disability, or serious disease. We also do not allow accounts whose primary purpose is inciting harm towards others on the basis of these categories."
(Moura, 2016)	"The speech of hatred refers to words that tend to insult, intimidate or harass people because of their race, color, ethnicity, nationality, sex or religion, or who have the ability to instigate violence, hate or discrimination against such persons."
(Cavalcante Segundo, 2016)	"Hate speech is one that aims to disseminate and promote hate on the basis of race, religion, ethnicity or nationality [...], even though it is not limited to such vectors and can also be example, according to gender, sexual orientation, etc."
(Fortuna, 2017)	"Hate speech is language that attacks or diminishes, that incites violence or hate against groups, based on specific characteristics such as physical appearance, religion, descent, national or ethnic origin, sexual orientation, gender identity or other and it can occur with different linguistic styles, even in subtle forms or when humour is used."

Table 2: Hate speech and related concepts

Concept	Definition	Difference to hate speech
Hate	Expression of hostility without any stated justification.	Hate speeches contain hatred directed against specific groups.
Cyberbullying	Name given to aggression and offenses practiced electronically among children and adolescents repeatedly and over time with the intention of humiliating and undermining the other.	Hate speech is more general and not necessarily aimed at a specific person.
Discrimination	Process through which a difference is identified is identified and used as the basis for unfair treatment.	Hate speech is a form of discrimination that manifests verbally or in writing.
Abusive language	Refers to harmful languages and includes discourses of hatred, profanity and derogatory messages.	Hate speeches are a form of abusive language.
Profanity	Offensive or obscene language.	Hate speeches may use profane language, but not necessarily.
Toxic language	Rude, disrespectful and irrational comments aimed at causing the person to quit.	Not all toxic comments contain hate speech.

Source: Adapted of Fortuna (2017)

Table 3: Types of hate speech and their targets

Category	Hate Targets
Race	people with light skin, afro-descendants
Behavior	insecure people, sensitive people
Physical appearance	obese people, beautiful people
Sexual orientation	gay, straight
Social class	poor people, marginalized people, rich people
Genre	pregnant, feminist
Xenophobia	Chinese, Indian, Northeastern
Deficiency	bipolar, autistic, people with specific needs
Religion	religious people, islamic
Other	drunks, people with little education

Source: Adapted of Silva *et al*. (2016)

Why is Automatic Detection of Hate Speech a Difficult Task?

Identifying hate speech in comments is difficult for different reasons. The language used in the text, for example, can be very noisy and contain misspellings, expressions or informal linguistic constructions. In addition, discrimination can occur veiled, requiring knowledge of user intent and context analysis to infer the existence of some hate speech.

For example, the comment "essi accent and very funny" (sic), posted on Facebook, was made in response to Northeastern rap singers, who were shriveled by their peers from other regions. Without this context, it is difficult for both the computer and a human judge to discern whether or not a comment contains abusive content. In the possession of this information, it is clear that the intention of the comment is to ridicule the singer specifically for the Northeastern accent, using it as a criterion for discrimination. The posting, therefore, is xenophobic. The commentary "rubbish accent, northeastern junk, has only sugarcane there kk" (sic) (made in the same context as the previous paragraph), on the other hand, contains explicit discrimination, making it easier to classify it as abusive and, specifically, xenophobic, since it attacks all north-easterners, even using stereotypes (the fact that the term "rapadura" is very common in the northeast) for this purpose.

Recently, in 2018, some racist declarations have gained prominence in the media, among which we highlight the following phrase: "always wanted to play a black hair" (sic). We identify in it a subtle discrimination, which is not determined by the presence of the word "negro", but by the way the author speaks of the Negro, putting him as someone differentiated, precisely by the color of his skin and the type of his hair, discriminating against him and thus practicing racism. This comment illustrates another situation in which it is difficult to classify hate speech by considering only the words of the sentence, without acknowledging the user's intention or the implicit meaning of his words.

In the work of Nobata *et al*. (2016), the authors list what they consider to be the basis for automatic classification of hate speech, which we summarize below:

- **Do not search for keywords only:** Use of regular expressions in the recognition of hate speech may result in false positives, since comments with typical discriminating expressions are not always discriminatory. In addition, the list of keywords based on blacklists often varies over time and can be overshadowed in different ways by users, making their use insufficient for sorting tasks
- **Abusive comments can be well constructed:** Not all hateful comments are written in informal words. Some may be very fluent and grammatically clean. Therefore, considering the presence of noise as grammatical errors is not sufficient for the automatic detection of hate speech
- **Hate speeches can cross sentences:** We often need to consider more than one sentence to determine whether a comment is abusive or not. Discriminatory ideas may be manifested in different sentences and world knowledge is often required to properly characterize them
- **Irony/Sarcasm:** It is not difficult for users to grasp ironic phrases to discriminate their targets, making recognition of their meaning even more difficult for humans

Related Works

In Nobata *et al*. (2016), the authors used three different datasets to analyze the impact of the model developed for identifying abusive content messages (which include hate speech). The first is the "Primary Data Set" which consists of comments provided and moderated by Yahoo! Finance and News in the period of October 2012 to January 2014. The second, called "Temporal Data Set", is used to analyze the influence of language change on the classification of comments as abusive or clean; the dataset was also provided by Yahoo! Finance and News and collected between April 2014 to April 2015. Finally, the third dataset, named "WWW2015 Data Set", was created by Djuric *et al*. (2015) and used by Nobata *et al*. (2016) for the purpose of comparison to the state-of-the-art in the field at the time of work completion. Authors trained a model using the machine learning program called Vowpal Wabbit with 4 different types of features: Linguistic, Syntactic and Semantic. Semantic features include pre-trained word embedding templates and an embedding template created with word2vec through a text corpus of Finance and News. *The work outperformed the state-of-the-art at 10 AUC* (0.9055 versus 0.8007). One of the most important contributions of this work was to prove that character-based n-grams are robust against datasets with a lot of noise, as is the case with those coming from social networks.

Badjatiya *et al*. (2017) performed several experiments using distinct deep learning architectures to classify the tweets collected and classified in the work of Waseem and Hovy (2016), used as benchmark: in all of them, there were more 16k tweets labeled as racist, sexist, or neither. The work explored deep techniques and used various aproache combinations like embeddings trained with LSTM, character N-Grams, TF-IDF, Bag of Words Vectors (BoWV) and (GloVe). They used both a CNN architecture and an LSTM, since their individual characteristics could make a difference in speech detection. In fact, the combination of an LSTM network, randomly generated embeddings plus a Gradient Boosted Decision Trees (GBDT) proved to be the best method, surpassing the State-of-the-Art in classifying hate speech with a F-Measure of 0.93. The embeddings were trained by LSTM and then submitted to GBDT. The LSTM architecture of Badjatiya *et al*. (2017) is represented in Fig. 8. It is worth mentioning that our work leverages this architecture (see Section 6).

Surprisingly, the randomly generated template performed better than the one initialized with GloVe, a fact that may indicate that the inherent power of LSTMs

to capture long series of word dependencies in tweets is maximized when the vector representation of the input data is randomly initialized. Another hypothesis is that GloVe does not have enough semantic representation power to improve the performance of a hate speech classification model as expected. A further hypothesis, which is most likely, is that the method of combining the embeddings of words to represent the embedding of texting tweets by word2vec is not robust enough. A paragraph embedding approach is probably more appropriate (Schmidt and Wiegand, 2017).

Schmidt and Wiegand (2017) studied a wide range of works on hate speech detection with natural language processing, highlighting the relevance of using features in various models discussed so far. Some of the most relevant features suggested by the authors were: character-based N-Grams, use of paragraph embeddings, comment analysis, lexical features such as presence of negative words, linguistic resources such as the class of words (POST-tag) and knowledge bases with expert knowledge of writing patterns in hate speech, to cite some.

Park and Fung (2017) used word-based convolutional (CNN) neural networks (represented as embedgings by means of word2vec), characters (converted to the one-hot coding) and a hybrid approach (words + characters) for the detection of abusive language. The dataset used was that of Waseem and Hovy (2016). Firstly, classification of the speeches as abusive or not was provided and, then, this information was used to classify its subtype (racist or sexist). Best result was obtained by the combination of Hybrid CNN with the logistic regression algorithm, achieving 0.828, 0.831 and 0.824 for accuracy, coverage and F-measure, respectively.

Gao and Huang (2017) demonstrated the importance of using the context of comments to be sorted. With a handcrafted dataset, they combined the logistic regression model with an LSMT architecture and extracted character-based N-Grams using semantic and lexical information for each word of the vocabulary. The best results in terms of accuracy, precision, coverage, Measurement-F, AUC were, respectively: 0.779, 0.650, 0.678, 0.600, 0.804.

The work of Fortuna (2017) is almost the only one dealing with corpora in Portuguese. The author has carried out an extensive study on several definitions that seek to delimit what are hate speech and related concepts, such as cyberbyllying. A manually typed dataset of tweets was constructed, totaling 5,668 messages of which 22% were declared as one of the 85 types and subtypes of hate speech considered in the paper. For the labeling task, the hierarchical classification approach was used: a technique that decomposes the classification task into a set of smaller problems, which can be efficiently solved and combined to classify documents composed of those (Hao *et al*., 2007). Fortuna (2017) also conducted experiments with binary classification (named as "unimodel") of speeches and multi-class classification (named "multimodel"). In terms of precision, the best unimodel and multimodel algorithm was Rpart (Kuhn, 2008) with 0.778 and 0.883, respectively. The best coverage value was achieved with the SVM Linear for both models, with 0.720 and 0.765, respectively.

Table 4 summarizes main works cited so far together with the achieved results, features and algorithms used.

PtBR Initiative

Despite the already mentioned shortage of scientific work to deal with the detection of hate speech in Portuguese languages, there are important initiatives that could help. One of them is the blog "Comunica que Muda" (CQM), which built a dossier called "Dossiê da Intolerância" (in English, Intolerance Dossier)[5,6] in regards to the digital world in Brazil. The dossier catalogs the most obvious types and the most common expressions and phrases used in social networks in Brazil. Ten different types of intolerance were monitored for three months - from April to June 2016. The types of hatred highlighted were those regarding the appearance of people, their social classes, the numerous disabilities, homophobia, misogyny, politics, age/generation, racism, religion, appearance and xenophobia. Whenever a word or phrase referring to one of these subjects was identified in a Facebook post, Twitter post, Instagram, some blog or comment on websites, it was collected and analyzed by the project team. In total, 542,781 mentions were analyzed. The method of classifying CQM comments was therefore based on blacklists. This is a known fragility from both capture and selection methods because many hate speech does not necessarily contain expressions present in such lists. Moreover, intolerant comments can be structured into several sentences in order to take into account previous sentences to determine whether the other is abusive (Nobata *et al*., 2016). So, it is very likely that the filter used in the project has left out a representative amount of abusive comments. The Table 5 displays some expressions/mentions used by the CQM blog to capture the desired content.

Expressions of Table 5 and details of the capture method were provided promptly via email contact by the project organizers. The list of classified comments, however, has not been released by the CQM group.

Although the capture method is based on blacklists, the extracted and classified comments were relevant insofar as they made clear the types of speeches that are given in social networks, as can be evidenced in the publicly available document of the Dossier. Clearly offensive or discreetly intolerant phrases were highlighted and showed the perceived and proven preconception scenario of the group.

[5]http://www.comunicaquemuda.com.br/dossie/quando-intolerancia-chega-as-redes/
[6]http://s18628.pcdn.co/wp-content/themes/comunica/dist/dossie/dossie_intolerancia.pdf

Table 4: Main approaches and results in terms of Accuracy (Acc), Precision (P), Coverage (C) and Measure-F (F), its features and algorithms used

Ano	Acc	P	C	F	AUC	Features	Algorithms	Paper
2017	–	0.93	0.93	0.93	–	*Word embeddings*, BoWV	Logistic Regression, Random Forest, GBDT, SVM, DNN, CNN	Badjatiya *et al*. (2017)
2017	0.78	0.78	0.72	0.764	–	word N-Grams	Logistic Regression, MLP, SVM	Fortuna (2017)
2017	–	0.828	0.831	0.824	–	Caracteres,*Word embeddings*,	Logistic Regression, SVM, CNN	Park and Fung (2017)
2017	0.78	0.65	0.68	0.60	0.80	N-Grams, *Word embeddings*, *Features* semantics and lexical,	Logistic Regression, LSTM, BiLSTM	Gao and Huang (2017)
2016	–	0.82	0.82	0.82	–	Tokens size, N-Grams, punctuations, POS-tag	skip-bigram	Nobata *et al*. (2016)

Table 5: Examples of expressions used by CQM to capture comments

#	Discoursetype	Expressions
1	Intolerance against appearance	“Narigudo”/ “seu” “gordo” / “gordo fazendo gordice” / “cabelo ruim”/ “cabelo de bombril”
2	Intolerance against social class	“Bolsa esmola” / “pobraiada” / “parece favelado” / “favelado é foda” / “coisa de favelado”
3	Intolerance against the disabled	“retardadomental”/“temdown”/“alejado” / “demente” / “leproso” / “aidético”
4	Homophobia	“boiola” / “baitola” / “gay” “desperdicio” /“cara de traveco” / “voz de traveco”
5	Misogyny	“feminazi” / feminista mal comida / odeio vagabunda/ vadia vagabunda / tudo vagabunda / “vai lavar louça” / “mal comida”
6	Political Intolerance	/ “comunista safado" / “coxinha fascista” / “comunista” "ladrão" / “bolsa esmola” / “ bolsa” “compra votos” /“petista vagabundo”
7	Prejudice against ag /generation	/ "velho asilo" / “não tenho idade” /"adolescente preguiçoso"/ "adolescente chato / “adolescente
8	Racism	“Cabelo ruim”/ “cabelo de bombril” / “não sou tuas nega” / “preto é foda” / “nego é foda”
9	Religious intolerance	“crente do rabo quente” / “crente do cu quente” /. “odeio crente” / “sem Deus no coração” / “muçulmano bomba”
10	Xenophobia	/ “arabe” “bomba” / “muçulmano” “bomba” / “japones é tudo igual” "voltapra sua terra” / "caiçara folgado"

Source: CQM Project

For this reason, we use these expressions as a starting point for capturing tweets and pre-selecting comments to be voted on in the construction and labeling process of the dataset we present in this paper. Details are discussed in Section 4.

Dataset Tagged with Hate Speech in PtBR

In this section, we depict the construction and labeling of the hate speech dataset for Portuguese language we have built and used in the experiments and which is publicly available. Manual labeling of a dataset is a laborious and error prone task, but it is highly required for supervised learning methods. We used the hatespeech comment database which Fortuna (2017) had made public to improve our dataset and help the validation of our models.

Selection of Data Sources

In the process of collecting comments, we need to carefully select the sources to maximize the likelihood that extracted texts contain some sort of hate speech, so that the proportion of true positive texts (those that actually contain hate speech) be representative (Schmidt and Wiegand, 2017). This strategy also makes it possible to direct the search process to specific sub-topics and sub-types of desired hate speech.

In the scope of this work, we do not intend to classify comments within specific types of hatespeech, but to determine the presence or absence of abusive content. That said, we carefully selected some websites that were highly likely to contain controversial issues and comments from haters and other discriminatory opinions or ideas. In total, 35 URLs have been listed, including news sites, Facebook communities, YouTube pages and forums.

The topics of news sites aggregate the list of articles whose theme is related to it. For instance, in the topic (listed by employees) http://g1.globo.com/politica/, we find all the G1 news relevant to the topic "politics". In the topic https://veja.abril.com.br/noticias-sobre/homofobia/, there

are articles categorized as "homophobia". The comments in the articles of these topics, as a consequence, has a high probability of containing content on the subject homophobia or politics, be they positive or negative.

Selection of Facebook Communities

Facebook communities were mostly selected by means of the hate map[7] created by the Laboratory of Image and Cyberculture Studies (Labic)[8], which is a map of admirers (extremists) of the Military Police on Facebook. The following quotation of CartaCapital magazine about the map[9] (translated for the purpose of the paper) summarizes it content:

> *"They are pages dedicated to defending the use of violence against what they call "bandits", "vagabonds" and "robbers", apologizing to lynchings and murder, defending police officers, publishing photos of "violated" or violently killed people, selling war equipment and combating human rights."*

Criterion for Capturing Tweets

Tweets were collected in regards to the keywords listed in Table 5. We also considered some keywords specially selected or copied from blacklist Hatebase[10]. Below, in Table 6, we quote the expressions added and separated by category.

It is worth emphasizing that the existence of one of these expressions in the text does not mean that it conveys hate speech Schmidt and Wiegand (2017). Conversely, the complete absence does not guarantee that it is clean. We use only a hint of probable relevance, considering its existence as a pre-selection criterion of tweets, since capturing all of them without criterion would be inefficient and not qualitative.

All the tweets that was captured by the above method were considered regardless of their size. Even the 1-word sized comments could be voted on.

Collection Method

We used facebook-sdk[11] to extract content from Facebook; for each community, we accessed all comments of the posts by its users, as well as the metadata provided by the API. In regards to Tweeter, we used the Tweetpy[12] library and connected to the tweets online streaming web site that the Twitter API provides, detecting those that contained one of the expressions present in the Tables 5 and 6. Forums and news websites do not provide API for extracting your content. For this reason, we used web scraping techniques to recognize users' comments and their available metadata. Some news websites like Estadão and Veja have an HTML structure that is difficult to predict, so we were not able to collect their information. The extraction process and parsing were executed by the Selenium wrapper for Python[13]. This was necessary because some page contents are only displayed by means of some sort of user-centred interaction like clicks, for example. The total number of pages and comments extracted for each collection source is listed in Table 7.

Following is a detailed list of the steps we have taken to collect the comments:

1. For sources without available API (Like news webpages):
   - We manually analyzed the HTML page of each collection source to determine the tags concerned to user comments
   - Using a web scraping tool, we extracted and saved each comment in the database
   - In addition to the comments, we saved the creation/editing data of the comment as well as the hyperlink of the webpage
2. Twitter:
   - We selected the catch expressions defined in the Tables 5 and 6 and introduced some linguistic variations to better fit the users' informal style of writing. As an example of that, we added the variation "vc" for the word "você" ("you"), "n" for "não" ("not") and so on
   - We used the Twitter API to capture all the tweets containing at least one of the previously defined expressions
   - We saved the serialized representation of each tweet for any needs
3. Facebook:
   - After selecting communities, we used the Facebook API to access all comments on posts on each of the pages
   - Additionally, we saved all the author's as well as the post data

Labelling

To facilitate the help of volunteers in the labelling process, a dedicated web application was developed for the task; user-friendly interface and responsiveness were central requirements.

[7]9https://www.cartacapital.com.br/blogs/outras-palavras/facebook-um-mapa-das-redes-de-odio-327.html/
[8]http://www.labic.net/
[9]https://www.cartacapital.com.br/blogs/outras-palavras/facebook-um-mapa-das-redes-de-odio-327.html
[10]https://www.hatebase.org/
[11]http://facebook-sdk.readthedocs.io/en/ latest/api.html
[12]http://www.tweepy.org/
[13]http://selenium-python.readthedocs.io/

Table 6: Expressions added to the table list 5 for capturing tweets

#	Discoursetype	Expressions
1	Xenofobia	“nordestino burro”/ "sotaque ridículo" / "sotaque lixo" / "nordeste lixo" / "nada contra o nordeste" / “nada contra nordestino”/ “nordeste não tem água” / "sotaque de viado" / "nordestino viado" / "volta pra sua terra"
2	Racismo	“caboclo” / "mestiço" / "volta para a senzala" / "carcamano"

Table 7: Total number of pages and comments extracted from different sources

Source	Pages	Comments
G1	11,774	724,997
Facebook	11	658
Youtube	81	74,013
Twitter	–	136,118
Stormfront	129	1,249
TOTAL	11,995	937,035

The system

The labelling Web application was developed in Python/Django and it is available at[14]. Figure 6 shows the welcome page, whose main function is to encourage the user to collaborate with the project.

The menu item "COMO CLASSIFICAR?" ("How to classify?") assist the volunteers to properly identify and distinguish between comments with hate speeches and clean comments and distinguish other types of offenses that are not hate speech, increasing their likelihood of making the best possible decision at the moment of the vote. After clicking the "OK, VAMOS CLASSIFICAR" ("Ok, let’s classify") button, the system is redirected to the classification page, shown in Fig. 7. The user is thus asked whether the shown comment conveys a hate speech. In the event of any doubt, it is possible to skip or check the "Não tenho certeza" ("I’m not sure") option, then press the "Salvar" ("Save") button.

In short, the system provides the volunteer with the opportunity to vote each comment in three different ways: (1) "Contém discurso de ódio" ("Conveys hate speech"), "Não contém discurso de ódio" ("Doesn’t convey") and "Não tenho certeza" ("I’m not sure"). We chose to record the vote of doubt for the comment in a way that would allow it to be voted on by another volunteer and thus increase the chances of the speech being recognized as clean or abusive, since if it were submitted to the same users who had doubts about its content the vote would probably be maintained.

We set up the system so that we do not repeat the next 150 comments for the current volunteer, based on his/her session data. Thus, in addition to increasing the variety of voted speeches, we limit the amount of votes that the same users can give for each comment.

Choosing Comments for Voting

Because we did not filter the comments from some collection sources, the size of the database grew considerably, this leads to being necessary to find out an efficient criterion for elicitation of comments at the time of voting and thus increase the amount of Positive Labeling by volunteers, as well as the number of speeches of different types and characteristics.

The comments were randomly selected according to the priority order set out below. When there is no comment that fits into a given selection criterion, the next criterion in the sequence is evaluated.

1. Comments with 2 votes exactly.
2. Comments with 1 vote exactly.
3. Comments voted undefined more than 1 time if there are more than 10 of them.
4. Comments containing or not one of the filter expressions defined for tweets and without any vote.
5. Any comment already voted.

Criteria 1 and 2 were placed to maximize the number of speeches with at least 3 votes, since without them most of the comments would not be submitted to a sufficient number of volunteers given the total amount of comments present in the database (Table 7).

Criterion 3, in turn, was used as a way to minimize the amount of comments with a vote of doubt. As we defined earlier, the next 150 comments were not repeated for the same volunteer and, in addition, Criterion 3 gives him the chance to vote speeches in which other volunteers had some uncertainty.

Using Criterion 4, we aimed to balance the number of voted comments that contains any of the expressions present in the Tables 5 and 6 and the number of comments that did not contain them. In this way, we kept the chance of those hate speech with more subtle discriminations being selected.

Criterion 5 ensures that there is always a candidate comment for voting, even when everyone has already been voted on. In practice, we never need to use this criterion since our total database is considerably large and the amount of volunteers is insufficient to vote for all of its comments.

[14]http://thiagodiasbispo.pythonanywhere.com/

DOI: 10.3844/jcssp.2019.1546.1571

Fig. 6: Welcome screen of the labelling Web application

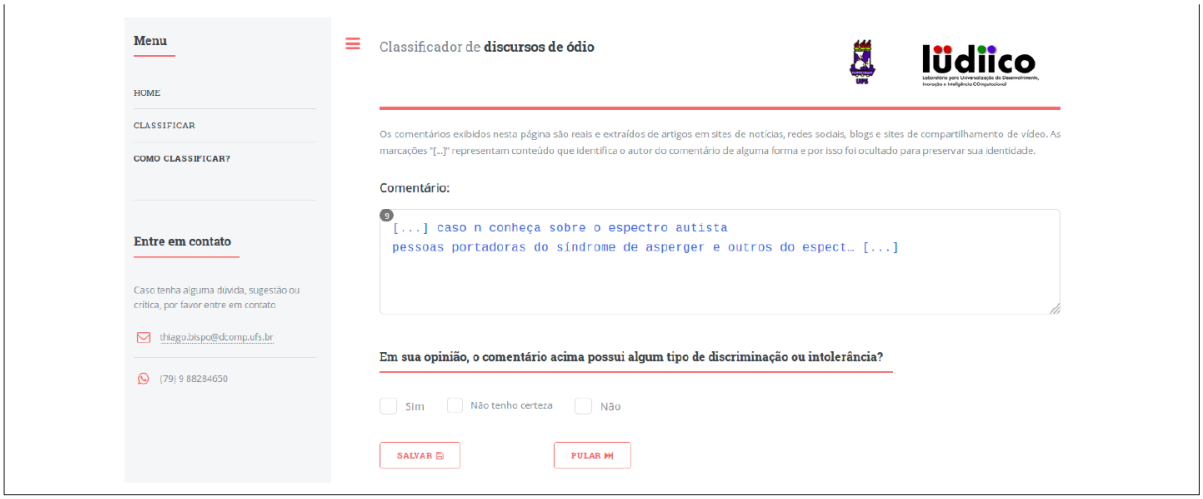


Fig. 7: Labelling system: classification screen

Labeling Results

During the period in which the volunteers contributed with the lettering, from 09/27/2017 to 05/15/2018, we got exactly 7,673 votes. The comments were randomly selected in order to increase the number of speeches with at least 3 votes and allow to classify those comments already marked with the option "NÃO TENHO CERTEZA" (I am not sure). Thus, whenever a speech reached at least 3 votes of the same type, i.e., 3 votes as "abusive" (abusive) or 3 votes as clean, we consider it classified.

We noticed that many comments were voted on several times as "NÃO TENHO CERTEZA" (I am not sure). According to some volunteers, they were faced with texts that, without the context to which the discourses were inserted, it was difficult to gauge the presence or not of abusive content.

The total of 1,191 speeches remained with 2 votes and 1,797 with 1 vote and for that reason we did not include them in our final dataset. Considering only those with at least 3 votes, we obtained a total of 1024 labeled comments, of which 491 were classified as abusive and 533 as clean.

In the total of classified speeches, we detected that 299 of them contained some of the expressions listed in the Tables 5 and 6, representing 24.83% of the 1,024 comments classified and made available in our dataset. This result demonstrates that there was a tendency for our method to choose more comments without such expressions, which is an important behavior for allowing exposure to a greater proportion of comments with abusive content less common to volunteers.

In the Table 8 we display the quantitative of comments by collection source. We note that all sources, from which any comments were saved in our database, were represented in our dataset. The data summarized therein illustrates that, not coincidentally, the amount of voted tweets is close to the amount of comment with any of the hate expressions (299). This result was expected, since tweets were captured using as a criterion the presence of such expressions.

Table 8: Quantitative discourse by source of collection

Source	Quantity
G1	493
Youtube	132
Twitter	248
Stormfront	92
Facebook	59

Table 9: Statistics of the size of abusive and clean comments

	Minimum	Maximum	Median	Mean
Abusive	1	119	12	13.12
Clean	1	88	15	15.75

Size of Comments

The comments showed a wide variation in sentence size, considering all the words originally present and disregarding punctuation marks. There is no significant difference between abusive and clean comments (Table 9). Since we did not have established the minimum amount of words in the comment. The minimum size of the labeled speech is 1 for both classes.

Being aware of the number of words present in our text is important because we can measure the maximum size that the training vectors need to have. It avoids to lose information.

Vocabulary Size

Vocabulary size was got based on the total amount of unique unigrams existing for the entire base, for the clean comment set and for the abusive ones. We disregard hashtags, links, mentions, punctuation marks (including emojis) and retweet ("RT") tags. As result, we extracted 3,607 unique unigrans from all 1,024 labeled messages. The abusive discourses have vocabulary of size 2022 unigrams. Clean discourses, however, have a unique 2,342 unigrams, 13.66% more than the vocabulary of hate speech.

Performance of Models Using Labeled Dataset

In this section we will detail the method used to evaluate the dataset annotated through different scenarios. Each scenario consists of a combination of one of the datasets used for training or testing LSTM model addressed here using distinct forms of preprocessing. We will explain the reasons for the construction of each scenario, the performance of the models using the metrics adopted (discussed later) and analysis of the results obtained.

Evaluation Metrics Adopted

Considering that TP is the number of positive examples correctly classified, FP is a number of negative examples classified as positive and FN represents the number of positive examples classified as negative, the metrics proposed to evaluate the method presented is as follows Alpaydin (2014):

- **Precision** - Defined as:

$$Precision = \frac{TP}{FP + TP} \tag{8}$$

  Intuitively, precision measures the model's ability to classify negative examples as negative. The higher the value, the greater the number of examples correctly classified as positive.
- **Recall** - Defined as:

$$Recall = \frac{TP}{TP + FN} \tag{9}$$

  In other words, Recall measures the efficiency of the model in "finding" all positive examples present in the dataset.
- **F-Measure:** Consists of the harmonic mean between accuracy and recall. This measure is approximately the average of both when their values are close.

$$F - Measure = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{10}$$

A model of high recall and low precision is able to classify many examples as positive, but few of them will actually be positive (FP). On the other hand, a model with low recall and high precision is able to classify few examples as positive, but in contrast there is a high probability that the positive labels are correct (TP).

In an ideal classification system, high Precision and Recall result in most precise one. When both metrics are equally important the balance between them, i.e., F-Measure, is quite adequate.

In this work, we adopt as positive the discourses classified as abusive and negative those classified as clean, that is, in which no discriminatory content was recognized.

We consider that in the real world, the higher the hit rate that a hate speech classifier has abusive content in comments, the better. For this reason, we understand that Precision is the ideal metric to evaluate models that intend to analyze hatespeech.

Preprocessing

In some PLN jobs, the preprocessing phase includes both the cleaning and tokenization steps as well as the data vectorization step when the task requires some machine learning activity.

The cleanup step can include several subtasks, such as: removing unnecessary content such as stopwords, grammar correction, converting text to lowercase letters, removing noisy characters such as coding problems, etc.

Radicalization can also be applied in the pre-processing step. Words are reduced to their morphological or inflected bases. This task has the effect of reducing the size of the textit dataset vocabulary. This is particularly important when we need to vectorially represent texts as input to machine learning algorithms.

Tokenization corresponds to the identification and separation of the important parts of the input data known in PLN as tokens, which can be words, sentences, characters, or any information extracted from the text as the grammatical classes of words (POS Tag).

The vectorization consists on the representation of the texts within a vector space. It is typically performed as the final step before enabling the use of data in machine learning algorithms.

In this work, we consider the pre-processing as being the set of tasks performed in the preparation of the data for the vectorization process.

The preprocessing we adopted was adapted from Badjatiya *et al*. (2017), which cleaned up texts by replacing specific tokens with expressions representing them. For example, URLs have been replaced by '<url>' mentions to users by '<user>', 'hashtags' by '<hashtag>', numbers by '<number>' and some emoticons by their meanings.

These representations are interesting because they preserve the occurrence of contents that can be determinant in tasks of categorization of texts as the classification of hate speech. We set up the preprocessing to make substitutions according to the language of the dataset we were working on. Thus, in Portuguese language texts, mentions to users, for example, are replaced by '<user>', numbers by '<numbers>' etc.

After this step, the texts are tokenized with the NLTK[15] as stopwords of the language in question and the punctuation marks removed.

Execution Environment

All the experiments presented were executed in an architecture cluster CCET-UFS (2017) with 5 GPUs nodes, 22 CPU nodes and a master node for coordination of other. The configurations of nodes with and without GPUs are presented in the Table 10.

With 5 GPUs nodes, 22 CPU nodes and a master node for coordination of other.

In addition, the architecture proposed here was implemented in the Python 3.4 programming language, using Keras[16] as a high-level prototyping tool for neural networks and TensorFlow running on backend configured to use GPUs and scikit-learn[17] as an auxiliary tool to perform some machine learning tasks such as GBDT and cross-validation.

LSTM Model

The model addressed in this paper is adapted from Badjatiya *et al*. (2017). This is the state of the art work in terms of prediction, to the best of our knowledge it has the higher accuracy and F-measure in the area of hate speech classification and proved to be quite suitable for task involving tweets. Since our main dataset (as described in Section 5.5) of training is the one used by the above-mentioned authors, we prefer to adopt the same preprocessing method, however adapting it to consider the language of the text in question, since the database constructed in this work is in Portuguese.

In Fig. 8, we illustrate the architecture of the LSMT model used. The training flow of the comments, from the input layer to its actual classification, follows the order of the steps described below:

1. **Pre-processing and vectorization:** The data are preprocessed according to Section 5.2 and vectored according to the need of each scenario, but in all of them, the resulting vectors are created in order to respect the maximum size of all training data. Size is represented by the variable max_sentence_length.
2. **Input layer:** Once vectored, the data is processed in batchs of size batch_size. The function of this layer is to create a representation of embeddings. The values of their dimensions are calculated through a uniform distribution and the quantity of them is controlled through the parameter embedding_dim.
3. **LSTM layer:** The function of this layer is to learn a suitable representation for each comment, thus generating specific embeddings aimed at the domain of hate speech Badjatiya *et al*. (2017). These representations are then used to classify the model, as detailed below. The LSTM parameters can be checked in the Table 11.
4. **Densa layer:** In deep models, it is common to add a fully connected layer for sorting the output data of the neural layer that precedes it. In our model, the

[15]http://www.nltk.org/api/nltk.tokenize.html#nltk.tokenize.casual.TweetTokenizer

[16]https://keras.io/

[17]http://scikit-learn.org/stable/supervised_learning.html#supervised-learning

dense layer consists of a Multi-Layer Perceptron, which is responsible for receiving the units dimension vector of the LSTM layer and determining its class. The classification error in this layer is then propagated to the previous layers and the updating of their parameters performed according to the optimization algorithm chosen. The output of this layer is given by a softmax function, responsible for calculating the probability for each class and thus predicting the class for the comments received.

As we defined earlier, in our experiments we used LSTM and its combination with a GBDT. We saved the result of the classification using the neural network and the dense layer MLP (result identified with the word "LSTM" after the scenario number) and then, using the same trained model, extract the learned embeddings and submit them to GBDT, thus performing a new classification (result identified with the word "GBDT" after the scenario number). In other words, when we combine the LSTM model with the GBDT, we ignore the result of the neural network classification using the dense layer and perform a new training with the GBDT.

Parameters Default Value

The parameters's default values are defined in the Table 11.

The maximum size adopted for each vectorized sentence (input_length) has been defined in each scenario and calculated to respect the maximum size of the training and test comments.

Multiple Datasets

When we run experiments with supervised learning algorithms, we need to be careful about various aspects that influence their performance. One of them is the process of choosing test training data. During training, few data can cause underfitting, that is, the model will not have enough information and examples to generalize in prediction.

In the opposite, overfitting occurs when the model becomes over-specializes in the input data, making it specialized to recognize only the already seen examples, but being weak when needs to deal with the examples that were not present in the training dataset.

In order to execute the training and testing processes with the LSTM model discussed here, we used the datasets of Waseem and Hovy (2016) and Fortuna (2017) because they were the only ones publicly available and found. The Table 12 lists all the databases used in this paper. The "Name" column refers to the name used to reference the dataset in this work. The "Size" column displays quantity and comments present in each dataset, showing the quantity of positives and negatives.

Table 10: LCAD Cluster Nodes Settings

Node without GPU	Node with GPU
20 Cores in 2 sockets Intel Xeon Ten-Core E5-2660v2 de 2.2-GHz, with 25MB of cache Cache, 8 GT/s	20 Cores in 2 sockets Intel Xeon Ten-Core E5-2660v2 of 2.2-GHz, com 25MB of cache Cache, 8 GT/s
64-GB of memory DDR3 1866 MHz	64-GB of memory DDR3 1866 MHz
1 disk of 160-GB SSD	1 disk de 160-GB SSD
1 port Infiniband QDR 4×40 Gbps	1 port Infiniband QDR 4×40 Gbps
	2 cards NVIDIA Tesla K20

Table 11: Default configuration of LSTM model parameters

Parameter	Description	Valor
input_dim	Maximum expected numeric value in input layer	10,000
output_dim	Word embedding size to be generated	200
input_length	Size of the input sentence vector	Variable
units	Number of cells in the LSTM layer	100
weights	Input layer initialization weights	[]
dropout_rate1	Dropout rate of the input layer	0.25
dropout_rate2	LSTM layer dropout rate	0.50
optimizer	Function optimizer and loss	"rmsprop"
loss_fun	Error function	"categorical_crossentropy"
bach_size	mini bach size	128

Table 12: List of datasets used in this work

Name	Description	Size
discursos_votado	Dataset built in this work	491 neg + 533 pos = 1024
discursos_votados_en	Dataset "discursos_votados" translated into English	491 neg + 533 pos = 1024
NAACL_SRW_2016	Dataset de tweets created by Waseem and Hovy (2016)	1034 neg + 5047 pos = 16081*
NAACL_SRW_2016_pt	Dataset "NAACL_SRW_2016" translated to Portuguese	1034 neg + 5047 pos = 16081*
NAACL_SRW_2016_cleaned_pt	Dataset "NAACL_SRW_2016" pre-processed and translated, in that order	11034 neg + 5047 pos = 16081*
dataset_portugues	Dataset in Portuguese created by Fortuna (2017)	1977 neg + 547 pos = 2524*

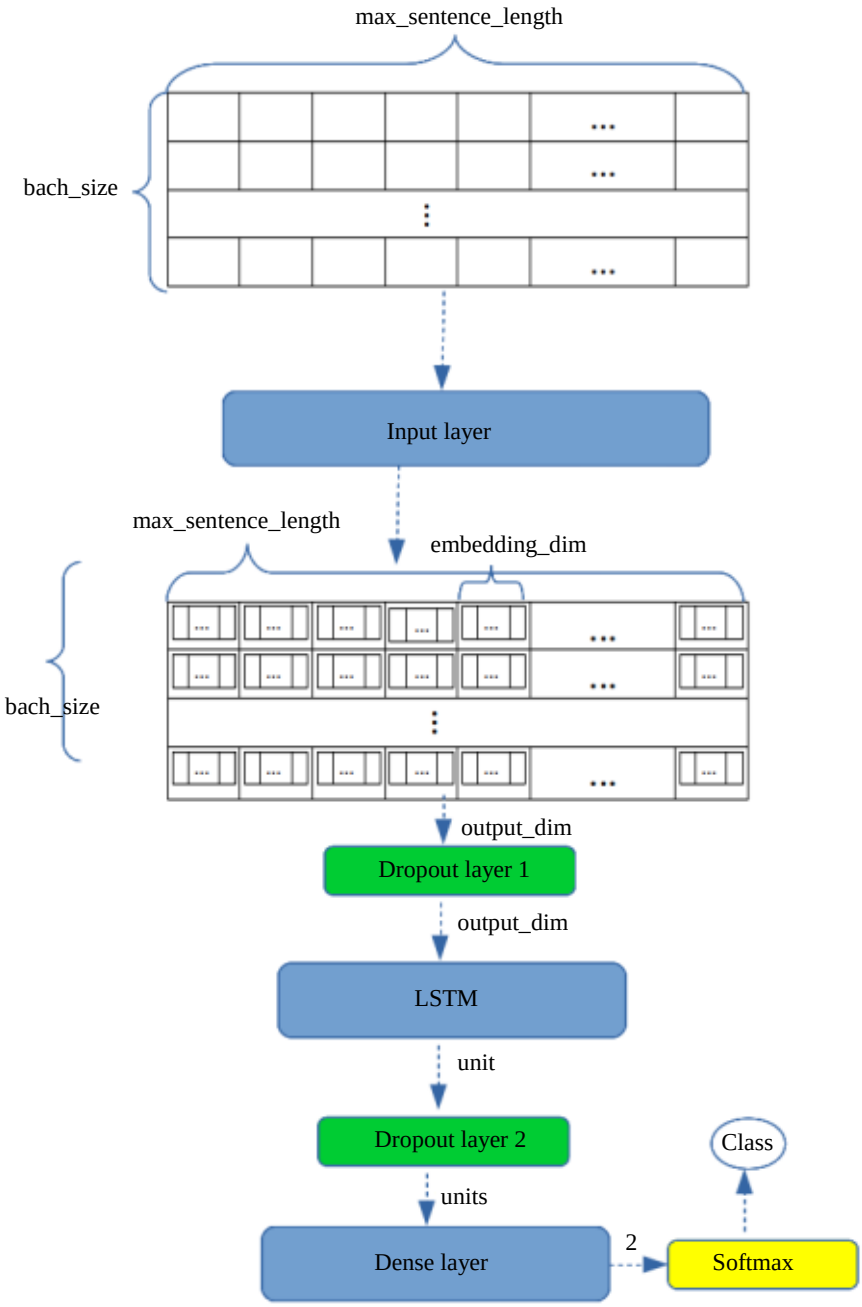


Fig. 8: LSTM model

NAACL_SRW_2016 is a set of tweets labeled from one of three distinct classes: "sexism", "racism" and "none." We assume that the labels corresponding to "none" refer to comments that do not contain hate speech, that is, using the expression adopted in this work, are clean comments. Comments labeled "racism" or "sexism" are therefore considered abusive comments because they contain some kind of hatespeech. In this way, we convert the base originally created with 3 types of different classes to a base of binary labels.

Originally, the NAACL_SRW_2016 has 16,914 tweets labeled, of which 3,383 are sexist, 1,972 are racist and 11,559 are neither. However, as tweets can only be made available through their IDs according to Twitter's privacy policy[18], we were able to download only 16,131 of them. We then detected that 25 of them were duplicates and with divergent labels and removed

[18]https://developer.twitter.com/en/developer-terms/agreement-and-policy

along with their duplicates, since redundant data can cause overfitting and divergent labels hampers the learning process of the model.

The dataset NAACL_SRW_2016_pt corresponds to the base "NAACL_SRW_2016 translated" into the Portuguese language with the help of the Google Translate API. The use of Automatic Translation Systems (STA) like this is not new in the task of textual categorization. Da Silva *et al*. (2018) used this approach to train a character-based CNN model in a translated dataset of feelings analysis. The results were similar to the original ones, i.e. results from the original English language.

The dataset dataset_portugues originally had 5,668 tweets annotated in several types and subtypes of hatespeech. Only 2,524 of them were available for download.

Training Methods

The scenarios described here were submitted to the training methods described below, according to the need of each scenario:

- **Method 1** - Cross validation: When the dataset is trained on itself, i.e., there is no independent test set, the template is cross-validated with 10 folds and caculated metrics by averaging their values in each fold. Therefore, the models trained by this method can present F-Measure value outside the range defined by the accuracy and Recall presented, since it was not calculated directly from these, but the average of their history
- **Method 2** - When training and test data are predefined: The model is trained with the complete training base and tested with the test data

In both methods, training occurs in mini batchs of size batch_size (Table 11).

Experimental Scenarios

We separate the experiments into scenarios so that their evolution becomes easier to understand. Each scenario corresponds to an attempt to improve the results of the previous scenario or a new approach to evaluate the performance of the preprocessing, vectorization or mixing of the datasets described in Section 12. Altogether, 24 scenarios have been defined.

In Table 13, we summarize the experiments performed in scenarios 1 to 4. Initially, we tested the performance of the LSTM + GBDT model in the task of classifying the hate speech present in NAACL_SRW_2016, considering their original labels (racism, sexism, or none) and using Training Method 1, which is run within Scenario 1, presented in Table 14, along with Scenario 2, described below. We show both the model result using the LSTM classifier and the GBDT. Network performance is displayed in terms of the metrics described in Section 5.1 and represents our baseline.

Our goal in Scenarios 1 to 4 was to validate the research hypothesis on the LSTM performance as a cross-lingual model using only the translated dataset, without data originally in Portuguese.

In Scenario 2, we trained the model with the NAACL_SRW_2016_pt dataset in order to validate its suitability for the task of predicting hate speech, also with ternary labels. The results are equivalent to the means of the values calculated along the cross validation, as previously explained. In parentheses, we show the standard deviations of the set of calculated metrics.

The metric values for each fold were macro weighted according to the number of labels for each class, as is commonly done in multi-class categorization.

Scenarios 3 and 4 (Table 14) consist of the same experiment executed in Scenarios 1 and 2, respectively. But this time, only binary labels are considered: we trained the model LSTM with the datasets NAACL_SRW_2016 and NAACL_SRW_2016_pt to classify the comments as abusive or clean.

Once the good performance of the trained LSTM with NAACL_SRW_2016_pt has been verified, we perform experiments in Scenarios 5 to 11 using the same network trained in different ways and tested with our dataset. The Scenarios are summarized in the Table 15 and their results listed in the Table 16. This time, we tried to validate the research hypothesis in a different way: considering data originally in Portuguese.

In Scenarios 5 and 6, the LSTM model was trained with NAACL_SRW_2016_pt and tested with "speeches", according to Training Method 2. Only the words present in the training dataset were considered in Scenario 5. In other words, the vocabulary of the model was the vocabulary of NAACL_SRW_2016_en after preprocessing and tokenization discussed in Section 5.2.

In Scenario 6, the vocabulary considered outside Glove100: only the words present in this model were added to the vector resulting from the vectorization process for each sentence. The hypothesis that justifies this approach is: using a vocabulary greater than Scenario 6, the smaller the number of words out of the vocabulary at the time of the vectorization and thus more information a cross-lingual model would have to generalize.

Table 13: Summary of the experiments performed in Scenarios 1 to 4

Scenario	Experiment
1	LSTM trained with NAACL_SRW_2016 using their own vocabulary for classification ternary
2	LSTM trained with NAACL_SRW_2016_pt using their own vocabulary for classification ternary
3	LSTM trained with NAACL_SRW_2016 using their own vocabulary for classification binary
4	LSTM trained with NAACL_SRW_2016_pt using their own vocabulary for classification binary

Table 14: Results of Scenarios 1 to 4

Scenario	Precision	Recall	F-measure
1 (LSTM)	0.818 (+/- 0.0095)	0.807 (+/- 0.0219)	0.807 (+/- 0.0176)
1 (GBDT)	0.913 (+/- 0.0097)	0.913 (+/- 0.0096)	0.913 (+/- 0.0096)
2 (LSTM)	0.813 (+/- 0.0065)	0.815 (+/- 0.0080)	0.812 (+/- 0.0070)
2 (GBDT)	0.918 (+/-) 0.0063)	0.918 (+/-) 0.0061)	0.918 (+/- 0.0063)
3 (LSTM)	0.739 (+/- 0.0457)	0.695 (+/- 0.0714)	0.712 (+/- 0.0178)
3 (GBDT)	0.867 (+/- 0.0104)	0.843 (+/- 0.0161)	0.855 (+/- 0.0097)
4 (LSTM)	0.732 (+/- 0.0159)	0.663 (+/- 0.0345)	0.695 (+/- 0.0173)
4 (GBDT)	0.873 (+/- 0.0125)	0.847 (+/- 0.0157)	0.860 (+/- 0.0097)

Table 15: Summary of experiments performed in Scenarios 5 to 11

Scenario	Experiment
5	LSTM trained with NAACL_SRW_2016_pt, your own vocabulary and tested with discursos_votados
6	LSTM trained with NAACL_SRW_2016_pt, GloVe vocabulary and tested with discursos_votados
7	LSTM trained with NAACL_SRW_2016_cleaned_pt your own vocabulary and tested with discursos_votados
8	LSTM trained with NAACL_SRW_2016_cleaned_pt, Glove vocabulary and tested with discursos_votados
9	LSTM trained with NAACL_SRW_2016_cleaned_pt + dataset_portugues, with the resulting vocabulary and tested with discursos_votados
10	LSTM trained with NAACL_SRW_2016_cleaned_pt + dataset_portugues, vocabulary GloVe and tested with discursos_votados
11	LSTM trained with NAACL_SRW_2016, own vocabulary and tested with discursos_votados_en

Table 16: Results of Scenarios 5 to 11

Scenario	Precision	Recall	F-measure
5 (LSTM)	0.606	0.274	0.377
5 (GBDT)	0.648	0.280	0.391
6 (LSTM)	0.720	0.126	0.214
6 (GBDT)	0.653	0.176	0.278
7 (LSTM)	0.627	0.283	0.390
7 (GBDT)	0.661	0.300	0.413
8 (LSTM)	0.703	0.156	0.255
8 (GBDT)	0.665	0.223	0.334
9 (LSTM)	0.659	0.171	0.271
9 (GBDT)	0.653	0.240	0.351
10 (LSTM)	0.679	0.167	0.268
10 (GBDT)	0.619	0.161	0.256
11 (LSTM)	0.558	0.281	0.374
11 (GBDT)	0.553	0.285	0.376

Table 17: Summary of experiments performed in Scenarios 12 to 15

Scenario	Experiment
12	LSTM trained with vectors TF-IDF of N-grams of characters of NAACL_SRW_2016_cleaned_pt and tested with discursos_votados
13	LSTM trained with a N-grams frequencies of characters of NAACL_SRW_2016_cleaned_pt and tested with discursos_votados
14	LSTM trained with the N-grams frequencies added to the index vetors of NAACL_SRW_2016_cleaned_pt and tested withdiscursos_votados
15	LSTM trained with the N-grams frequencies concatenated to the index vectors of NAACL_SRW_2016 cleaned_pt and tested with discursos_votados

Since NAACL_SRW_2016 is composed of tweets which contains informal expressions like hastags, mentions to users, URLs, slangs etc., their translation (NAACL_SRW_2016_en) contains many unknown English words.

For this reason, we created the NAACL_SRW_2016_clenaned_pt and use them in Scenario 7. This dataset consists of the preprocessed NAACL_SRW_2016 according to the method described in Section 5.2 and, then, translated. As discussed earlier, the preprocessing has the advantage of not deleting certain content from the text, transforming it into "well-behaved" expressions.

To be clear about the motivation behind creating NAACL_SRW_2016_clenaned_pt, note the comment below belonging to NAACL_SRW_2016 and labeled as sexist:

> Borrowed time #CuntAndArsehole cant wait for you to get_from_file blown away by the decent teams. #FirstElimination #BeatItDogs #MKR #KatAndAndre (sic)

The "#" symbol will be deleted in the preprocessor and the English words that follow will remain. As they are known Portuguese words, at the time of the vectorization, they will be ignored and replaced by a special token representing a word unknown in the vocabulary.

In the preprocessed phrase below, note the substitution of hashtags for the word "hashtag", which on the one hand causes its content (and possible semantic value) to be eliminated, but allows for inclusion of representative generic information:

> borrowed time hashtag cant wait for you to get_from_file blown away by the decent teams. hashtag hashtag hashtag hashtag

This time, all words are recognized and represented by specific values at the time of vectorization, returning a more significant representation than their translated version without preprocessing and enriching the resulting vector with neighborhood information that these expressions represent.

For example, in the calculated vocabulary index vector of the expression above, after the index of the word "descents", the index of the word "hashtag" will be added. In regards to the translated comment of the original English version, the hashtags maintained would be transformed into a special index that represents an unknown word, as we said previously.

After this experiment, in Scenario 8, we evaluated the performance of the trained LSTM with NAACL_SRW_2016_clenaned_pt using the GloVe100 vocabulary, thus increasing the training vocabulary in the same way as in Scenario 6. Then, we tried to join the previous dataset with the dataset_portugues and to train using the resulting vocabulary (Scenario 9) and the vocabulary GloVe100 (Scenario 10).

Our goal was to introduce in the training set abusive and clean discourses, originally written in Portuguese, to mitigate possible negative effects that the NAACL_SRW_2016 machine translation may cause in the training process and, thus, improve model test performance by using our dataset, since the comments are in Portuguese.

In Scenario 11, we tested the inverse of what we tested in Scenarios 5 to 10: we trained the LSTM with the NAACL_SRW_2016 and tested it with our dataset automatically translated to English. This time, the goal was to evaluate if the translations from Portuguese to English cause less noise than the translations from English to Portuguese, improving the performance of the trained model.

Up to Scenario 11 we vectored the comments through their calculated vocabulary index vectors, converting these vectors into embeddings of hate speech used in each scenario.

We also tried to vectorize the comments using two traditional techniques of vectorization in Machine Learning: N-grams frequency vectors and TFIDF vectors. Both are Bag-of-Words techniques that allow the calculation of the frequency of the top K N-grams that are present in each training and test example. The first represents the comments using the frequencies themselves, i.e., the occurrence counts or, depending on the case, the presence or absence of each sequence relevant to each comment. The second one constructs vectors in order to represent the importance of the N-gram for each sentence in relation to all the others.

In Scenarios 12 to 15 (summarized in Table 17), we vectored the comments using the TFIDF vector character counting methods discussed earlier. In all of them, we trained the LSTM model with the dataset NAACL_SRW_2016_cleaned_pt and tested it with discourses_voted. Their results are found in Table 18.

In Scenario 12, we trained with TFIDF vectors using the dataset training vocabulary itself, while in Scenario 13 we used frequency vectors. Next, in Scenario 14, we tried to add to the vectors of vocabulary indexes to the frequency vectors of each comment. In Scenario 15, we concatenate these two vectors for each training and test example. The Table ref tab: summaryCenario12-15 summarizes the experiments performed for each of these scenarios.

Table 18: Scenarios result from 12 to 15

Scenario	Precision	Recall	F-measure
12 (LSTM)	0.000	0.000	0.000
12 (GBDT)	0.000	0.000	0.000
13 (LSTM)	0.000	0.000	0.000
13 (GBDT)	0.417	0.000	0.000
14 (LSTM)	0.530	0.381	0.443
14 (GBDT)	0.563	0.443	0.496
15 (LSTM)	0.000	0.000	0.000
15 (GBDT)	0.000	0.000	0.000

Table 19: Summary of the experiments performed in Scenarios 16,17 and 18

Scenario	Experiment
16	LSTM trained with NAACL_SRW_2016_cleaned_pt, your own vocabulary and tested with dataset_portugues
17	LSTM trained with NAACL_SRW_2016_cleaned_pt, GloVe vocabulary and tested with dataset_portugues
18	LSTM trained with dataset_portugues, your own vocabulary and tested with discursos_votados

Table 20: Scenario results of 16, 17 and 18

Scenario	Precision	Recall	F-measure
16 (LSTM)	0.275	0.402	0.326
16 (GBDT)	0.274	0.349	0.307
17 (LSTM)	0.284	0.325	0.303
17 (GBDT)	0.279	0.369	0.318
18 (LSTM)	0.586	0.146	0.234
18 (GBDT)	0.596	0.191	0.290

Table 21: Summary of experiments performed in Scenarios from 19 to 24

Scenario	Experiment
19	LSTM trained with **NAACL_SRW_2016_cleaned_pt lemmatized, GloVe** vocabulary and tested with **dataset_portugues**
20	LSTM trained with o vetores de N-Grams concatenated with index vectors **NAACL_SRW_2016_cleaned_pt from lemmatized** and tested with discursos_votados.
21	LSTM trained with **NAACL_SRW_2016_cleaned_pt** lemmatized, your own vocabulary and tested with **dataset_portugues + discurso_votado lemmatized.**
22	LSTM trained with **NAACL_SRW_2016_cleaned_pt** lematizado, your own vocabulary and tested with **dataset_portugues + discurso_votado lemmatized.**
23	LSTM trained with **NAACL_SRW_2016_cleaned_pt lemmatized**, your own vocabulary and tested with **discursos_votados lemmatized.**
24	LSTM **bidirectional** trained with **NAACL_SRW_2016_cleaned_pt**, your own vocabulary and tested with **discursos_votados.**

Table 22: Scenarios result from 19 to 24

Scenario	Precision	Recall	F-measure
19 (LSTM)	0.219	0.322	0.261
19 (GBDT)	0.206	0.360	0.262
20 (LSTM)	0.000	0.000	0.000
20 (GBDT)	0.000	0.000	0.000
21 (LSTM)	0.000	0.000	0.000
21 (GBDT)	**0.676**	0.047	0.088
22 (LSTM)	0.346	0.231	0.277
22 (GBDT)	0.329	0.250	0.284
23 (LSTM)	0.562	0.161	0.251
23 (GBDT)	0.560	0.296	0.388
24 (LSTM)	0.703	0.049	0.091
24 (GBDT)	**0.586**	0.077	0.136

In Table 19, we describe the results of the dataset NAACL_SRW_2016_cleaned_pt used for LSTM training and tested with discursos_votados, without and with the vocabulary GloVe100, in Scenarios 16 and 17, respectively. We also trained the model with the dataset_portugues and we tested with discursos_votados (Scenario 18). The results can be checked in the Table 20. In these scenarios, we validate our research hypothesis by training or testing the model with another dataset in Portuguese that was not ours.

To simplify the vocabulary of the dataset NAACL_SRW_2016_cleaned_pt, we try to add the stemming step Martin and Jurafsky (2009) to its pre-processing. By hypothesis, some of the previous results could be improved if the number of words in common between the training set and the test set increased. As lemmatization reduces words to their most basic lexicographical units, it is important to validate our hypothesis. The experiments were run in Scenarios 19 through 23 (Table 21) and their results are listed in Table 22.

In scenario 24, we used a BiLSTM (respecting the same LSTM architecture discussed here) to evaluate how it behaved in training with the NAACL_SRW_2016_cleaned_pt and test discursos_votados as a cross-lingual model.

Result Analysis

Our goal with Scenarios 1 and 2 was to partially reproduce the experiment used in the State of the Art paper and validate the quality of the automatic translation of its dataset using the same LSTM model combined or not with the GBDT for ternary classification. The results using Scenario 1 are similar to the values reached by Badjatiya *et al*. (2017) when using the combined models: 0.913 in precision against 0.93 of the article. Using the initialized LSTM with random weights, our results were subtly better than theirs: 0.818, 0.807 and 0.807 against 0.805, 0.804, 0.804 accuracy, recall and F-measure, respectively.

The little difference in the result is well justified because the dataset used in this work does not have the same number of examples of its original version, since some tweets were unavailable at the time of their download and duplicate examples were removed, as explained before.

Scenario 2 used NAACL_SRW_2016_pt for the ternary sort task and achieved better results if compared to the same template trained in the original text in English: 0.918 versus 0.913 for the metrics adopted, considering only the result of the model combined with the decision tree. The result is promising because it validates the hypothesis that the use of the dataset translated for training the same model used with the English data guarantees comparable results.

The fact that the model performs well in both datasets in the classification into 3 distinct classes does not guarantee that it will achieve similar or even reasonable hit rates in the two-class classification. For this reason, in Scenarios 3 and 4, we performed the same experiments of Scenarios 1 and 2 changing them only for binary classification. The best result was in terms of accuracy: 0.867 and 0.873 in Scenarios 3 and 4, respectively.

Their results are lower than their versions run with the 3 classes. We believe that the difference occurs because of the patterns of clean or abusive discourses that are best recognized when grouped into their original classes (sexist, racist, or none), thus preserving the specific features that define them. Despite the difference, the values are still good, especially because one of the datasets is the result of machine translation.

It is worth mentioning that the use of GBDTs has drastically improved all the results of the LSTM network in the scenarios discussed above, demonstrating that its "boosting" technique is also effective in dealing with networks trained in datasets translated as in our case.

The LSTM model and its combination with GBDT produced good results, which validates our research hypothesis: the same architecture achieved good results both for the original comments in English and for its translated version, characterizing as cross-lingual. The next step is to determine the validity of the hypothesis when the same model is evaluated with comments in Portuguese, as is the case of our dataset.

After completing experiments with the main training datasets from Scenarios 5 to 11, we evaluated the performance of the same model validated in Scenarios 1 and 2, testing it with our dataset. As we can see, we achieved expressive results in several scenarios in terms of Precision, within which the best of the results was that of Scenario 6, with 0.720 in this metric, thus validating our research hypothesis again of the cross-lingual character of the LSMT architecture.

Another highlight was Scenario 8, whose vectoring process considered only the vocabulary words GloVe100, causing tokens not present in them to be discarded, but providing the advantage of being a much larger vocabulary and potentially containing more words present in the training and test data.

Note that the results of Scenarios 6 and 8 are similar: 0.720 and 0.703, respectively. In the case of Scenario 8, we used NAACL_SRW_2016_clenaned_pt in order to reduce the number of words out of the vocabulary at the time of the vectorization. This approach, however, did not produce significant differences to justify the use of the strategy, contrary to the hypothesis established previously.

The use of the GBDT did not always result in an improvement in the performance of the LSTM, as we can see in Scenarios 6, 8, 9, 10 and 11 whose precision

values declined with the use of the Decision Tree. One of the hypotheses for this is that the database was not large enough so the GBDT could be able to improve and stabilize the LSTM classification

Unlike precision, from which we obtained reasonable results, the coverage results in none of the Scenarios between 5 and 11 were good, all remaining below 0.5. The F-measure, since it is the harmonic mean between the precision and coverage values, was pushed to values also below 0.5, as a consequence of the coverage behavior in all scenarios.

Looking at the coverage Equation (9), we note that the ratio of their low values is usually high False Negative (FN) rates. The higher the value, the higher the denominator and the lower the final value of the fraction. Scenario 6, we can easily identify this behavior through its confusion matrix (Fig. 9) calculated from the values predicted by the LSTM network (without GBDT).

By convention, in the confusion matrix the lines represent true positive values (abusive comments) and true negatives (clean comments). Columns represent the values predicted by the network. Crossing rows and columns, we establish the relationships we need to calculate the metrics we use in this work. 466 comments were classified as clean when in fact they are abusive (False negatives).

However, as we argued earlier, we consider accuracy to be the most important metric for hate speech detection models. In this sense, the precision values of Scenarios 6 and 7 are indicative of the ability of the model to actually recognize abusive discourses.

In Scenarios 12 through 15, we tried out some classic machine learning features associated with the LSTM. Our goal was to evaluate if sequences of occurrences of character N-Grams (Scenarios 13, 14 and 15) trained by a recurring network could cause the learning of the sequences of these occurrences, thus associating the BoW model with a model whose sequence order input is important. In Scenario 12, we initially tested the behavior of the network using TFIDF vectors. None of the scenarios presented satisfactory results, as we can observe in Table 18.

We can conclude that vectors of occurrences of character sequences, alone, do not aggregate information enough to allow the kind of learning that LSMT requires. Note that in Scenario 14, there was some gain in information relative to others, however derisory. In it alone, we take into account the vectors of indexes of words, by their sum to the vectors of frequencies of N-Grams. Probably for this reason, the network was able to gain some learning using the portions of the input vector of the network that contained the vocabulary word indexes.

In Scenarios 16 and 17, we tried to test the LSTM model with the dataset_portugues to evaluate its performance by using it as test data. The results were even lower in terms of accuracy, which was below 0.3. Even the LSTM trained as the dataset_portugues and tested with our dataset (Scenario 18) had unsatisfactory results, getting below 0.6.

This result was expected, since the amount of training data is very small relative to the amount of parameters to be trained (even though it does not train the embeddings in the LSTM), which sum in this Scenario 80.602. When this type of problem occurs, we come across a curse of dimensionality Christopher (2016).

In Scenarios 19 to 23, we evaluated the performance of the LSTM model using in some cases the lemmatization technique and in Scenario 24 we applied the dataset NAACL_SRW_2016_cleaned_pt in a BiLSTM model. The best results achieved in terms of accuracy were in Scenarios 21 and 24. In the first, we tested the network by joining our dataset lemmatized as dataset_portugues using the vocabulary of the training data itself and reached the precision of 0.676. In the second, the LSTM network reached 0.703. In this scenario, the application of the GBDT resulted in the worse performance of the network. In Scenario 21, the network went from precision 0.0 to 0.676 only with the use of the decision tree.

In scenario 20, note that, again, the network failed to learn from the use of N-Grams sequences even with the typed input data, reinforcing the clue that this type of data representation is not suitable for recurrent models.

The result of Scenario 24 suggests that the BiLSTM model also behaves as a cross-lingual model, confirming our research hypothesis once again, as well as the traditional LSTM. Its accuracy is comparable to our best Scenario (Scenario 6), although the use of the GBDT associated with it did not cause performance improvement.

Scenarios 1 through 4 are promising: they demonstrate that the LSMT model addressed is robust in recognizing hate speech even in automatically translated textit datasets, confirming that it can be used both in Portuguese and in English, without any special configuration.

However, performance in the following scenarios were not as good as those 4, with some models and preprocessing techniques notorious for achieving high accuracy but low coverage overall. A fact that confirms that the models are correctly classifying what in fact considers hate speech.

The results also suggest that working on a commentary as is with multiple types of hate speech but treated as if they were all clean or abusive speeches can be challenging. The various patterns inherent in each type probably require more training and test data and preferably data with types or subtypes of speech compatible with each other.

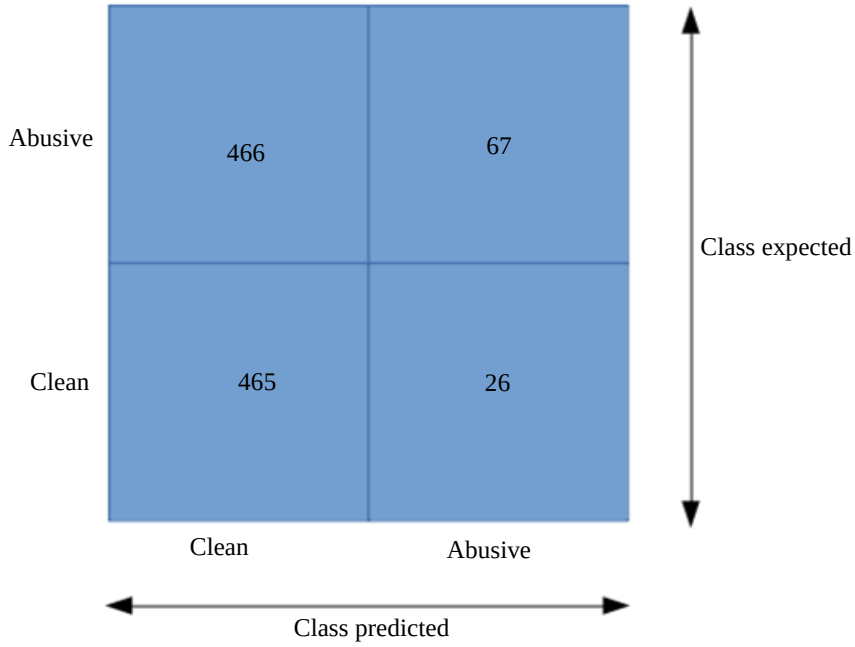


Fig. 9: Scenario 6 confusion matrix using the LSTM result

It is likely that if our database contained only sexist, racist, or clean discourses, some of the scenarios would achieve superior accuracy and coverage when trained with the NAACL_SRW_2016_pt database.

In summary, we verified the following items after the experiments:

- **Homogeneous bases:** Based on Scenarios 1, 2, 3 and 4: Training and testing bases with the same types of speech tend to achieve better results
- **GBDTs are robust:** Its boosting technique allows you to stabilize and improve rankings even in difficult scenarios such as those involving different language textures, most of the time
- **External vocabulary:** The use of external vocabulary, despite the computational cost when it is very large, is decisive for vectorizing the input data, mainly in decreasing the amount of unrecognized words in the vectorization
- **Sequence vectors of words:** In isolation, they do not represent good features for recurring models such as LSTM

Conclusion

In this work, various models for hate speech classification were analysed from collection of user comments. An english database was employed to explore different approaches of data preprocessing and vectoring.

Several cross-lingual models were trained to detect hate speech as a general work objective. Due to the lack of labeled datasets in Portuguese language, we created a dataset from volunteers contributions. The experiments were divided into scenarios to evaluate the performance of model for each input data provided, especially for hate speech dataset. The precision achieved is 0.720 in hate speech assessment.

The major contribution of this work is the proposal of techniques to automate and aid hate speech detection from content available in social media. To fight discriminating comments, State of Art approaches were used to demonstrate their low effectiveness in Portuguese language domain.

In order to do this, we explored several forms of pre- processing and vectoring of the datasets used and those submitted to the LSTM model and its variant, BiLSMT. By the detection of which approaches represented by the 24 scenarios created were the most promising within the scope of our work, they serve as a reference for other researchers.

In summary, our contributions were: (i) training of cross-lingual models through the use of an English database for automatic classification of hate speech in Portuguese, despite the lack of this language datasets (ii) determination of the most promising pre-processing and vectorization techniques using dataset in English, serving as a reference to researchers in area detection of hate speech interested in cross-lingual models.

As future work, we consider the research of features of semantic nature as a bias for the classification of hate speech. Some articles found use semantic information to iteratively find more sentences similar to those already

found based on words enriched by WordNet information, for example.

In addition, since labelled hate speech corpus is extremely rare, working with unsupervised or semi-supervised learning as in Xu *et al*. (2017) seems quite adequate for pre-identification of abusive comments. Another very promising area is the semantic frames Barreira *et al*. (2017). They are ideal for problems such as identifying hate speech, both for not requiring huge databases and for easily identifying new patterns of discrimination and offense, a feature that is appreciated since variation in language and hate speech varies constantly in the digital world.

Acknowledgment

The authors thank CAPES and FAPITEC-SE for the financial support [Edital CAPES/FAPITEC/SE No 11/2016 - PROEF, Processo 88887.160994/2017-00]. The authors also thank FAPITEC-SE for granting a graduate scholarship to Flávio Santos and CNPq for granting a productivity scholarship to Hendrik Macedo [DT-II, Processo 310446/2014-7]. Finally, the authors thank the research group Ludii.co/DCOMP/UFS.

Author's Contributions

Thiago D. Bispo: Bibliographic review, development, execution and validation of experiments; manuscript writing.

Hendrik T. Macedo: Principal advisor of work and scope delimitation; manuscript writing.

Flávio de O. Santos: Assistance in theory and practice related to experiments; implementation and validation of codes used for training.

Rafael P. da Silva and Adolfo Guimarães: Assistance in theory and practice related to experiments.

Leonardo N. Matos, Bruno O.P. Prado and Gilton J.F. da Silva: Assistance in theory and practice related to experiments; manuscript writing.

Ethics

This paper is original with unpublished material. The corresponding author confirms that this manuscript has not been published elsewere and that no ethical issues are involved.

References

Alpaydin, E., 2014. Introduction to machine learning. MIT Press.

Badjatiya, P., S. Gupta, M. Gupta and V. Varma, 2017. Deep learning for hate speech detection in tweets. Proceedings of the 26th International Conference on World Wide Web Companion, Apr 03-07, Perth, Australia, pp: 759-760. DOI: 10.1145/3041021.3054223

Barreira, R., V. Pinheiro and V. Furtado, 2017. Framefor-uma base de conhecimento de frames semânticos para perícias de informática (framefora knowledge base of semantic frames for digital forensics) [in portuguese]. Proceedings of the 11th Brazilian Symposium in Information and Human Language Technology, (HLT' 17), pp: 171-180.

Breiman, L., 2001. Random forests. Machine Learn., 45: 5-32.

Cavalcante Segundo, A.D.H., 2016. Questão de Opinião? Lumen Juris.

CCET-UFS, 2017. Laboratório de computação alto desempenho. Laboratório de Computação Alto Desempenho, Universidade Federal de Sergipe.

Christopher, M.B., 2016. Pattern Recognition and Machine Learning. 1st Edn., Springer, New York, ISBN-10: 1493938436, pp: 738.

Da Silva, R.P., F.A.O. Santos, F.B. do Nascimento and H.T. Macedo, 2018. Cross-Language Approach for Sentiment Classification in Brazilian Portuguese with Convnets. In: Information Technology - New Generations, Latifi, S. (Ed.), Springer International Publishing, Cham, ISBN-13: 978-3-319-77027-7, pp: 311-316.

Djuric, N., J. Zhou, R. Morris, M. Grbovic and V. Radosavljevic, 2015. Hate speech detection with comment embeddings. Proceedings of the 24th International Conference on World Wide Web, May 18-22, ACM, Florence, Italy, pp: 29-30. DOI: 10.1145/2740908.2742760

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

Friedman, J., T. Hastie and R. Tibshirani, 2001. The Elements of Statistical Learning. 1st Edn., Springer, New York, ISBN-13: 978-0-387-84858-7, pp: 748.

Friedman, J.H., 2002. Stochastic gradient boosting. Comput. Stat. Data Anal., 38: 367-378. DOI: 10.1016/S0167-9473(01)00065-2

Gao, L. and R. Huang, 2017. Detecting online hate speech using context aware models. Arxiv preprint arXiv: 1710.07395.

Goodfellow, I., Y. Bengio and A. Courville, 2016. Deep learning. MIT Press.

Hao, P.Y., J.H. Chiang and Y.K. Tu, 2007. Hierarchically SVM classification based on support vector clustering method and its application to document categorization. Expert Syst. Applic., 33: 627-635. DOI: 10.1016/j.eswa.2006.06.009

Hartmann, N., E. Fonseca, C. Shulby, M. Treviso and J. Rodrigues, 2017. Portuguese word embeddings: Evaluating on word analogies and natural language tasks. arXiv preprint arXiv:1708.06025.

Hochreiter, S. and J. Schmidhuber, 1997. Long short term memory. Neural Comput., 9: 1735-1780.

Jaouedi, N., N. Boujnah and M.S. Bouhlel, 2019. Deep learning approach for human action recognition using gated recurrent unit neural networks and motion analysis. J. Comp. Sci., 15: 1040-1049. DOI: 10.3844/jcssp.2019.1040.1049

Kuhn, M., 2008. Building predictive models in r using the caret package. J. Stat. Software, 28: 1-26.

Liu, S., N. Yang, M. Li and M. Zhou, 2014. A recursive recurrent neural network for statistical machine translation. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, (ACL' 14), Association for Computational Linguistics, Baltimore, Maryland, pp: 1491-1500.

Marsland, S., 2014. Machine Learning: An Algorithmic Perspective. CRC Press, ISBN-13: 9781466583283, pp: 457.

Martin, J.H. and D. Jurafsky, 2009. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. 1st Edn., Upper Saddle River, ISBN-10: 0131873210, pp: 988.

Mikolov, T., K. Chen, G. Corrado and J. Dean, 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Moura, M.A., 2016. O Discurso do Ódio em Redes Sociais. Lura Editorial (Lura Editoração Eletrônica LTDA-ME).

Nobata, C., J. Tetreault, A. Thomas, Y. Mehdad and Y. Chang, 2016. Abusive language detection in online user content. Proceedings of the 25th International Conference on World Wide Web, Apr. 11-15, International World Wide Web Conferences Steering Committee, Canada, Montréal, Québec, Canada, pp: 145-153. DOI: 10.1145/2872427.2883062

Park, J.H. and P. Fung, 2017. One-step and two-step classification for abusive language detection on twitter. arXiv preprint arXiv:1706.01206

Pennington, J., R. Socher and C. Manning, 2014. Glove: Global vectors for word representation. Proceedings of the Conference on Empirical Methods in Natural Language Processing, (NLP' 14), Association for Computational Linguistics, Doha, Qatar, pp: 1532-1543.

Praseetha, V.M. and S. Vadivel, 2018. Deep learning models for speech emotion recognition. J. Comput. Sci., 14: 1577-1587. DOI: 10.3844/jcssp.2018.1577.1587

Schmidt, A. and M. Wiegand, 2017. A survey on hate speech detection using natural language processing. Proceedings of the 5th International Workshop on Natural Language Processing for Social Media, (PCM' 17), Association for Computational Linguistics, Valencia, Spain, pp: 1-10.

Silva, L.A., M. Mondal, D. Correa, F. Benevenuto and I. Weber, 2016. Analyzing the targets of hate in online social media. Proceedings of the 10th International Conference on Web and Social Media, May 17-20, AAAI Press, Cologne, Germany, pp: 687-690.

Singhal, P. and P. Bhattacharyya, 2016. Sentiment analysis and deep learning: a survey.

Sutton, C.D., 2005. Classification and regression trees, bagging and boosting. Handbook Stat., 24: 303- 329. DOI: 10.1016/S0169-7161(04)24011-1

Waseem, Z. and D. Hovy, 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. Proceedings of the NAACL Student Research Workshop, (SRW' 16), Association for Computational Linguistics, San Diego, California, pp: 88-93.

Xu, Z., J. Li, B. Liu, J. Bi and R. Li *et al*., 2017. Semi-supervised learning in large scale text categorization. J. Shanghai Jiaotong Univ., 22: 291-302. DOI: 10.1007/s12204-017-1835-3