UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

FACULTAD DE INGENIERÍA

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

IMPROVING THE AUTOMATIC IDENTIFICATION OF AGGRESSIVE COMMENTS ON SOCIAL MEDIA BY INCORPORATING AUTHOR AND MESSAGE CONTEXT

BY:

MARCO EMANUEL CASAVANTES MORENO

THESIS PRESENTED AS REQUIREMENT TO OBTAIN THE MASTER'S DEGREE IN COMPUTER ENGINEERING

CHIHUAHUA, CHIH., MÉXICO **FEBRUARY OF 2020**

Improving the automatic identification of aggressive comments on social media by incorporating author and message context. Thesis presented by Marco Emanuel Casavantes Moreno as partial requirement to obtain the Master's Degree in Computer Engineering, has been approved and accepted by:

[illegible]

M.I. Javier González Cantú
Director of the Faculty of Engineering

[illegible]

Dr. Alejandro Villalobos Aragón
Secretary of Research and Postgraduate

[illegible]

M.S.I. Karina Requena Yáñez
Academic Coordinator

[illegible]

Dr. Luis Carlos González Gurrola
Thesis director

February of 2020

Date

Committee:

Dr. Luis Carlos González Gurrola, director
Dr. Manuel Montes y Gómez - INAOE, codirector
Dra. Graciela María de Jesús Ramírez Alonso, examiner
M.I. Jesús Roberto López Santillán, examiner

UNIVERSIDAD AUTÓNOMA DE
CHIHUAHUA

25 de febrero de 2020

I.S.C. MARCO EMANUEL CASAVANTES MORENO
Presente

En atención a su solicitud relativa al trabajo de tesis para obtener el grado de Maestro en Ingeniería en Computación, nos es grato transcribirle el tema aprobado por esta Dirección, propuesto y dirigido por el director **Dr. Luis Carlos González Gurrola** para que lo desarrolle como tesis, con el título: **IMPROVING THE AUTOMATIC IDENTIFICATION OF AGRESIVE COMMENTS ON SOCIAL MEDIA BY INCORPORATING AUTHOR AND MESSAGE CONTEXT.**

Index

UNIVERSIDAD AUTÓNOMA DE
CHIHUAHUA

Solicitamos a Usted tomar nota de que el título del trabajo se imprima en lugar visible de los ejemplares de las tesis.

A T E N T A M E N T E
"Naturam subiecit aliis"

EL DIRECTOR

FACULTAD DE INGENIERÍA U.A.CH.
DIRECCIÓN

[illegible]

M.I. JAVIER GONZÁLEZ CANTÚ

EL SECRETARIO DE INVESTIGACIÓN Y POSGRADO

[illegible]

DR. ALEJANDRO VILLALOBOS ARAGÓN

Dedication

This thesis is dedicated to my family, for giving me their unconditional support and motivation to reach my goals, particularly through the process of pursuing my master's degree.

Acknowledgments

I owe a debt of gratitude to my university and to CONACYT for giving me the opportunity to complete this work.

I am grateful for being able to work under the guidance of my teachers, particularly my supervisors Dr. Luis Carlos González Gurrola and Dr. Manuel Montes y Gómez, whose passion for research inspired me to strive for knowledge and have fun while learning.

Abstract

This research was conducted with the purpose of developing better approaches to tackle the problems that involve cyberbullying and hate speech in social media. People can deal with negative experiences in social networks, such as being a target of cyberbullying or exposing themselves to hateful and vulgar content. It is crucial to address the importance of early identification of users who promote hate speech, as this could allow important outreach programs to prevent consequences in real life, such as self-harm or suicide; however, traditional approaches to detect these behaviors only use the short text messages provided in datasets by task organizers and these texts suffer from lack of context. The objective of this research is to design a classification method based on author and text metadata to identify hostile comments on social networks. In order to build new classification methods for this task, it was necessary to extend the available collections of Twitter® tagged data, where in addition to having the text of the tweet and its class/label, it was possible to retrieve context information in the form of tweet and author features. As a result, we found that there are statistically significant differences between the classification reports of the methods that use metadata with respect to the conventional approaches that lack additional information to the text in the majority of datasets tested.

Table of Contents

List of Figures

List of Tables

1

Introduction

In the 90's, availability of online documents started to grow along with renewed interest to develop automated text categorization tools, in other words, a machine learning approach that could build a classifier by "learning" from labeled documents to replace previous methods based on manual efforts of domain experts [1]. Some of the applications of automatic text classification include information retrieval systems, document filtering, document organization, and more recently it found its way into handling data from social media platforms. Technology has changed the way in which people communicate with each other, giving rise to new services such as social networks like Facebook® or Twitter®, where informal writing styles are used to share ideas. Such social networks, though, present several challenges to keep their sites free of malicious content. Hate speech is a problem affecting the interactions between online groups. The intolerance and aggressiveness of certain users provokes a negative impact on the experience of other consumers or people interested in being part of the communities and their conversations. The task of classifying offensive content online is difficult since a common problem for impoliteness, rudeness, and swearing research is that all three phenomena are impossible to define universally because all are culturally and personally determined [2], even more so, the fact of not being face to face in the communication channel and even preserve anonymity, encourages these individuals to express themselves offensively. On the other hand, the volume of messages that are sent daily, the growth of online communities, and the respective ease of access to these social networks, makes the moderation of communication channels a difficult task to be dealt with by conventional means, and as people increasingly communicate online, the need for high quality automated abusive language classifiers becomes much more profound [3].

1.1 Antecedents

When investigating what has been developed to mitigate this problem during literature review, it was found that in [4] the Lexical Syntactic Feature (LSF) approach was proposed to identify offensive content on social networks (taking into account the specific style of writing, structure and abusive content of the person), and further predict the potential of a user to send offensive content. This method managed to perform better than those used in its time but conclude that considering the context would help detect offensive language that does not contain vulgar words.

In [3] a method was developed based on machine learning to detect hate speech in the comments of online users from two domains (“Yahoo!® Finance and News”) that surpasses a cutting-edge deep learning approach; they experiment with several characteristics for this task, such as different syntactic characteristics, as well as different types of embeddings (vector representations of words and comments), and find them very powerful when combined with the standard features of Natural Language Processing (NLP). Character n-grams perform well in their noisy data sets (texts with differences in their superficial representation, such as misspellings and abbreviations, and their original intention) and determine that a future work area includes the use of the context of the comment as additional features.

In 2017, the authors of [5] created their own dataset of Hate Speech and Offensive language from Twitter, and performed one of the most complete studies of the subject by its time. They implemented stemming, and created n-grams weighted by their *tf-idf*, constructed Penn Part-of-Speech tag n-grams to capture information about the syntactic structure, used modified Flesch-Kincaid Grade Level and Flesch Reading Ease scores to capture the quality of each text, and a sentiment lexicon to assign sentiment scores to each tweet. They also include binary and count indicators for hashtags, mentions, retweets, and URLs, among other features.

The framework developed in [6] investigates the effectiveness of developing an abstract layer of linguistic features based around the use of ’othering’ language. This kind of language represents a speech act that aims to protect resources for the in-group, and includes terms and phrases that separate the ingroup (e.g ’we’, ’us’) from the outgroup (e.g. ’them’, ’those’), suggesting action or separation based on perceived symbolic and realistic threats (e.g. 'send them', 'get out'). Their hypothesis was that this additional layer would provide better context for a classifier beyond words alone.

After carrying out this review of previous related work, it is clear that, as text is the only feature available in the associated datasets along with a class label, practically all efforts to tackle this classification task are focused solely on feature extraction based on text.

1.2 Problem statement

The volume of text-based exchanges in social media have made human editorial approaches unfeasible [7], and recent decisions and rulings by regulatory authorities explicitly mention automatic systems as tools to help mitigate the spread of mischievous content, proving their high social relevance [8]. Furthermore, administrators of social media platforms could prevent abusive behavior and harmful experiences. It is crucial to address the importance of early identification of users that promote hate speech, as this could enable important outreach programs, to prevent an escalation from speech to action [9]. Moreover, considering the high levels of aggressiveness and hostile behavior of certain users towards particular groups or individuals, more serious real-life issues, like self-harm or suicide, could actually be prevented.

Many share tasks events are held each year in different places around the world to tackle this challenge and plenty of research is published to test new algorithms and approaches. To mention a few, "GermEval Shared Task on the Identification of Offensive Language" is intended to foster research on the identification of offensive content in German language. GermEval is a series of shared task evaluation campaigns that have been run informally by self-organized groups of interested researchers and were endorsed by special interest groups within the German Society for Computational Linguistics (GSCL) [10]. The study conducted by [11] focuses on detecting vulgar and pornographic obscene speech in Arabic social media without the need for manually creating word lists. They publicly released their own datasets along with the lexicons they created, motivated by the limited previous work for Arabic.

The authors of [8] presents the 1st edition of HASOC Hate Speech and Offensive Content Identication in Indo-European Languages, namely: German, English, and Hindi. Their objectives are to motivate research for these languages and to find out the quality of hate speech detection technology in other languages.

However, the issue is that spotting offensive messages and hate speech is challenging because systems cannot rely on the text content [8]. A significant amount of research papers [4, 5, 9, 11, 12, 13] point out that a recurrent obstacle for this tasks is the difficulty to judge each comment due to lack of context, as in social media this is often limited. Text is the only feature available in almost all hate speech/aggressive /offensive datasets, but some research documents suggest the possibility of using different features based on user metadata [14].

The work proposed in [15] looks at the effectiveness of the 'profile description' field on Twitter to carry out the task of user classification. Their results show that such metadata can be an effective feature for any classification task.

In another study [16], the properties of the content of the message, user and its use of the social network are extracted, in addition to describing the attributes that characterize Twitter members that exhibit behaviors of abuse and aggression. It is shown that its methodology for the analysis, labeling and classification of data can scale up to millions of tweets, while its machine learning model (created with a Random Forest classifier), can distinguish between normal, aggressive and cyberbullying users with high precision (more than 90% of Area Under the Curve). This means that different features need to be explored besides text.

1.3 Research questions

The information given in the previous section motivated the proposal of the next research questions:

- What are the most successful representations used for automatic detection of aggressive comments?
- Is contextual information important to improve aggressive content detection?
- What kind of profiling provides greater advantages for the task of identifying aggressive comments?

1.4 General objective

To design a classification method considering author and text metadata to identify hostile comments on social networks.

1.4.1 Specific objectives

1. To extend datasets used in aggressive detection tasks, retrieving metadata of available samples.
2. To propose a classification method that combines features based on metadata with traditional approaches of text representations.
3. To determine the relevance of contextual information in the aggressive detection task, by comparing the results of classifiers using only text representations against classifiers using text and features based on metadata.

1.5 Justification

This research proposal is relevant because the volume of information that these texts (messages, comments, and publications on social networks) represent exceeds the capacity for human analysis in a reasonable time. Some aspects to consider are:

1. The average adult reads approximately 250 words per minute (wpm) [17].
2. The maximum time of focus is, on average, 45-50 minutes, with rest intervals of 10 minutes [18].
3. Only in the U.S. approximately 105 million tweets are generated per day [19, 20], each with an average of 50 words [21].
4. About 15,000 bullying-related tweets are posted every day [22, 23].
5. More than 1 in 3 young people have experienced cyber threats online [24].
6. Fewer than 1 in 5 cyber bullying incidents are reported to law enforcement [24].

This data tells us that approximately 420,000 hours of focused reading would be required to cover all tweets generated on a single day; this makes obvious the need for an automatic tool that helps social networks in this task, relieve the workload and detect situations involving hate speech and cyberbullying so that these matters can be spotted in time and prevent real-life issues.

1.6 Conceptual contribution

Analysis of the inclusion of metadata for the task of automatic detection of offensive comments to:

- Improve and increase the knowledge that exists about contemporary aggressiveness detection methods through the proposals made in this investigation.
- Develop theoretical relationships between contextual attributes and online interactions.
- Identify the dependency or independency of metadata and the task.

1.7 Empirical contribution

The evaluation and examination of the effects of metadata of messages and authors as variables to the relationship between text comments and their label/class.

1.8 Hypothesis

Offensive language detection is a complex task, automatic systems cannot rely solely on features extracted from short texts that lack contextual information, for that reason, the hypothesis of this work is: taking into account message and author metadata improves automatic detection of aggressive content on social media.

1.9 Research method

For this project, an experimental method of research was used; algorithms (independent variables) were tested to detect aggressive comments (dependent variable) and data was collected and used to compare the results between these algorithms.

2

What offensive language is and how it could be modeled

Having established how aggressive behavior takes place in social media, propagates with ease and poses a challenge for moderators and authorities, it is important to define what makes a text offensive and what kind of "branches" exist within its concept.

2.1 Offensive language, Hate Speech, Cyberbullying and offensive text messages.

2.1.1 Offensive language

In this research we will use the definition proposed in [2], referring to this kind of language as vulgar, pornographic and hateful.

2.1.2 Types of offensive language

As stated in [4], vulgar language refers to gross expressions, which include explicit and offensive references to sexual or bodily functions. Pornographic language is the representation of explicit sexual matter for the purpose of sexual arousal and erotic satisfaction. Hateful language includes any out-of-law communication that belittles a person or a group based on some characteristics such as race, color, ethnicity, gender, sexual orientation, nationality and religion.

Table 2.1: Examples of offensive language.

Vulgar	**Pornographic**	**Hateful**
"I hate when strippers tell you they're dancers hoe ballet dancers don't pop their pussy to Beethoven you a stripper"	"Going back to school sucks more dick than the hoes who attend it"	"He's a beaner smh you can tell he's a mexican"
"I named my penis the truth because bitches can't handle it"	"If I don't get my dick sucked at your party by a bad bitch I'm far gone set it off"	"lmfaoooo I hate black people"
"Some of these hoes deserve a kick in their vagina lucky I ain't tryna loose my shoe"	"I shoot at the pussy I bust in the pussy I'm cuming too soooooooooooon"	"I really just want to kill some towel head terrorists already can high school be over now please"

2.1.3 Hate Speech

It is defined by [5] as a language used to express hatred towards a target group or to be derogatory, to humiliate or to insult its members.

Table 2.2: Examples of Hate Speech.

Hate Speech (HS)
"Spics are half breed trash, no filthy native should be allowed to speak to any european"
"The #south of the US is white trash"
"The blacks in #california are typical ni**ers"

2.1.4 Cyberbullying

It is defined as an aggressive, intentional act carried out by a group or individual, using electronic forms of contact, repeatedly and over time against a victim who cannot easily defend himself [13], while in [25] it is defined to include being: called offensive names, purposefully embarrassed, stalked, sexually harassed, physically threatened and harassed in a sustained manner.

2.1.5 Offensive text messages

Following the criteria to identify hate speech proposed in [9], a text message is offensive if it:

- Uses a sexist or racial slur
- Attacks a minority
- Seeks to silence a minority
- Criticizes a minority (without a well-founded argument)
- Promotes, but does not directly use, hate speech or violent crime
- Criticizes a minority and uses a straw man argument
- Blatantly misrepresents truth or seeks to distort views on a minority with unfounded claims
- Shows support of problematic hash tags
- Negatively stereotypes a minority
- Defends xenophobia or sexism

2.2 Text classification

The aggressive detection task can be seen, at its core, as the assignment of text documents to one or more predefined categories based on their content, in other words, a text categorization task [26]. Every automatic text categorization task involves two major components:

1. Feature extraction through text representations.

2. Machine learning methods of classification.

2.3 Text representations

In order to apply machine learning techniques to solve NLP problems, words must be coded so a computer can understand them, that is to say, vocabulary of documents must be represented in a computer-ready language. The segments below describes two of the most employed methods to achieve this: *Bag-of-Words* and *Word Embeddings*.

2.3.1 *Bag-of-Words*

Most machine learning applications in the text domain work with the bag-of-words representation (BoW). This model treats each word present in a collection of documents as a feature, and since each file only contains a small subset of the whole vocabulary, BoW is an extremely sparse representation. The value assigned to individual features can be either positive (if the word exists within the document) or zero (if the word is absent). The positive values can be normalized term frequencies or simple binary indicators. For example, consider the next two documents:

- the weenie dog chases a cat
- my cat does not like dry food

A BoW representation of these sentences, filled with binary indicators, would look like table 2.3, where each column refers to a term and each row is a document.

Table 2.3: Example of a *Bag-of-Words*.

	the	**weenie**	**dog**	**chases**	**a**	**cat**	**my**	**does**	**not**	**like**	**dry**	**food**
Doc1	1	1	1	1	1	1	0	0	0	0	0	0
Doc2	0	0	0	0	0	1	1	1	1	1	1	1

Alternatively, a BoW can also consider character n-grams as features:

There may be some applications (where a binary input is strictly required, or when presence is more important than frequency) for which binary representations are good enough due to its

Table 2.4: Example of a Bag of Character 3-grams.

	the	**wee**	**een**	**eni**	**nie**	**dog**	**cha**	**has**	**ase**	**ses**	**cat**	**...**
Doc1	1	1	1	1	1	1	1	1	1	1	1	...
Doc2	0	0	0	0	0	0	0	0	0	0	1	...

simplicity. However, if frequency is indeed relevant for the task at hand, the use of normalized frequency of terms is a better way to fill the values of a BoW. This variant is referred to as the *tf-idf* model, where *tf* stands for the *term frequency* and *idf* stands for the *inverse document frequency*. Consider a document collection containing *n* documents in *d* dimensions. If $X = (x_1 \ldots x_d)$ is the d-dimensional representation of a document after the term extraction phase, then x_i represents the unnormalized frequency of said document, where all the values of x_i are nonnegative and most are zero.

The first step to normalize term frequencies is to compute the inverse document frequency of each term. The inverse document frequency id_i of the *i*th term is a decreasing function of the number of documents n_i in which it occurs:

$$id_i = \log{(n/n_i)} \tag{2.1}$$

The term frequency is normalized by multiplying it with the inverse document frequency:

$$x_i \Leftarrow x_i \cdot id_i \tag{2.2}$$

One problem with *idf* normalization is that it might increase the frequency of misspellings and errors that weren't handled in the preprocessing stage.

In summary, the universe of words (or terms) corresponds to the dimensions (or features) in this model, turning them into a sparse multidimensional representation, where the ordering of the terms is not used [27].

2.3.2 Word and Document Embeddings

Word ordering conveys semantics that cannot be inferred from the bag-of-words representation. For example, consider the following pair of sentences:

- The cat chased the mouse
- The mouse chased the cat

Clearly, the two sentences are very different but they are identical from the point of view of the bag-of-words representation. For longer segments of text, term frequency usually conveys sufficient evidence to robustly handle simple machine learning decisions like binary classification. This is one of the reasons that sequence information is rarely used in simpler settings like classification. On the other hand, more sophisticated applications with fine-grained nuances require a greater degree of linguistic intelligence. A common approach is to convert text sequences to multidimensional embeddings because of the wide availability of machine learning solutions for multidimensional data. However, the goal is to incorporate the sequential structure of the data within the embedding. Such embeddings can only be created with the use of sequencing information because of its semantic nature [27]. The simplest approach is to use a 2-gram embedding:

- For each pair of terms t_i and t_j the probability $P(t_j \mid t_i)$ that term t_j occurs just after t_i is computed.
- A matrix S is created in which S_{ij} is equal to $[P(t_i \mid t_j) + P(t_j \mid t_i)]/2$.
- Values of S_{ij} below a certain threshold are removed.
- The diagonal entries are set to be equal to the sum of the remaining entries in that row. This is done in order to ensure that the matrix is positive semi-definite.
- The top-*k* eigenvectors of this matrix can be used to generate a word embedding.

The linguistic power in the embedding depends almost completely on the type of word-to-word similarity function that is leveraged [27].

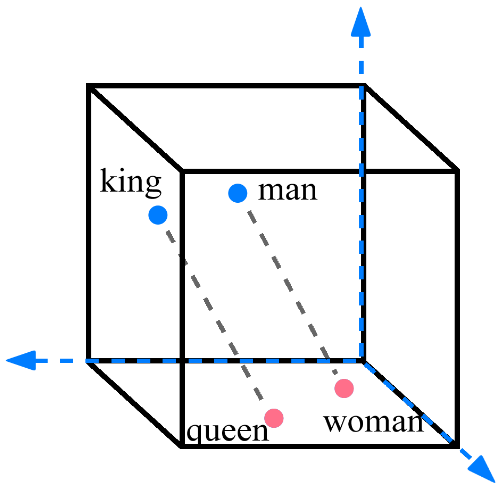


Figure 2.1: Example of word embeddings on a three-dimensional space.

The main idea behind this technique is that words that are similar in context (at least according to the text from which the embeddings algorithm trained with) appear closer to each other in a multidimensional space. Based on this, one can use the position of the words in this space to compute the similarity and relation that the text has with its surroundings.

2.4 Machine Learning Algorithms

Machine Learning is about making computers modify or adapt their actions (such as making predictions), so that these actions get more accurate, where accuracy is measured by how well the chosen actions reflect the correct ones. It is only over the past decade or so that the inherent multidisciplinarity of machine learning has been recognized. It merges ideas from neuroscience and biology, statistics, mathematics, and physics, to make computers learn [28]. Machine Learning systems can be classified according to the amount and type of supervision they get during training. There are four major categories: supervised learning, unsupervised learning, semi supervised learning and Reinforcement Learning [29]. When we feed the training data and the desired solutions or labels to an algorithm, we are talking about supervised learning, and a typical task in this category is classification. The classification problem consists of taking input vectors and deciding which of N classes they belong to, based on training from exemplars of each class. The most important point about the classification problem is that it is discrete, each example belongs to precisely one class, and the set of classes covers the whole possible output space [28].

For this research, we chose Linear Support Vector Machines and Linear Regression as baseline classifiers. These algorithms are considered part of the traditional approaches for most of the NLP tasks, they are well established, reliable and still competitive to this day. Because of its transparency, computational affordability and flexibility at handling different types of inputs, Random Forest was selected as the blender of the information present in our proposed models.

2.4.1 Linear Support Vector Machines

A Support Vector Machine (SVM) is one of the most popular models in modern machine learning due to its versatility and power. They were introduced by Vapnik in 1992 and have taken off radically since then, principally because they often provide significantly better classification performance than other machine learning algorithms on reasonably sized datasets. This method works by finding the support vectors, the most useful data points in each class in a dataset that lie closest to the classification line, a line that separates the classes in the best way possible maximizing the margin (largest radius around the classification line) before we hit a data point; this leads to an interesting feature of these algorithms: after training this model we can throw away all data except for the support vectors, and use them for classification [28].

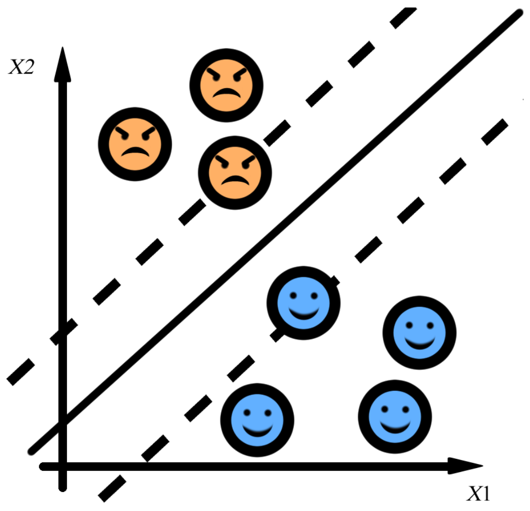


Figure 2.2: Example of a Linear Support Vector Machine.

In this algorithm, we can determine our classifier line by using the standard equation of the straight line (equation 2.3):

$$y = w \cdot x + b \tag{2.3}$$

In equation 2.3, *w* is the weight vector, *x* is the particular input vector and *b* is the bias weight. As demonstration, we can use the classifier shown in figure 2.2 by saying that any *x* value that gives a positive value for *y* is above the line and therefore an example of the "aggressive" orange class, and any x that gives a negative value becomes part of the "neutral" blue class. A few distance constraints need to be added to take account of the margin. If we consider *M* to be the perpendicular distance between a dashed line and the classification line, we need to check if the absolute value of *y* is less than *M*:

$$\text{Assign orange class to } x \text{ if } y = w \cdot x + b \geq M \tag{2.4}$$

$$\text{Assign blue class to } x \text{ if } y = w \cdot x + b \leq -M \tag{2.5}$$

However, this technique alone is not appropriate for datasets with outliers, since these kind of data points can make a classification problem non-linearly separable. In order to generalize and be useful for most real world cases, it needs to allow for some mistakes. This would be called a Soft Margin Classifier, as it has to look for the widest margin with the fewest classification mistakes, also named margin violations [29]. In a mathematical way, the function that we want to minimize is:

$$L(w, \epsilon) = w \times w + \lambda \sum_{i=1}^{R} \epsilon_i \tag{2.6}$$

In function 2.6, *R* is the number of misclassified points, and each ϵ_i is the distance to the correct boundary line for the missing point [28]. We can see that a new parameter is included, λ (which is also know in Scikit-Learn's SVM classes as the C hyperparameter). A small λ means that we prioritize a large margin over a few errors, a large value of λ represents the opposite.

2.4.2 Logistic Regression

Even though it may seem as a contradiction to use the term regression in the name of a classifier, Logistic Regression (LR) is similar to linear regression, with the exception that instead of predicting a continuous value, it simply predicts whether something is true or false, in other words, this algorithm uses a linear regression equation that includes a function called “logistic/sigmoid function”, this function produces an “S” shaped curve that is able to tell the probability of class assignment.

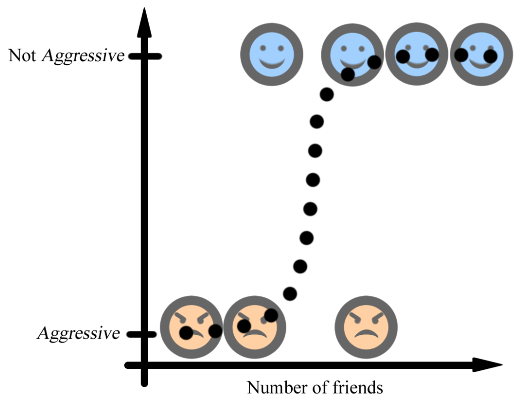


Figure 2.3: Example of sigmoid function in Logistic Regression.

The sigmoid function is defined [30] as:

$$f(t) = \frac{1}{1 + e^{-}t} \tag{2.7}$$

Now, we can consider *t* as a linear function in a univariate regression model [31]:

$$t = \beta_0 + \beta_1 x \tag{2.8}$$

Therefore, the Logistic Equation becomes:

$$p(x) = \frac{1}{1 + e^{-}(\beta_0 + \beta_1 x)} \tag{2.9}$$

The choice of the model parameters is a problem that involves finding a hypothesis that best explains our data. The “S” curve is fit to the data using a process called “maximum likelihood”. Basically, all the data points are used to calculate the likelihood of the data given the line generated by the sigmoid function. This curve shifts positions until a line with maximum likelihood is selected.

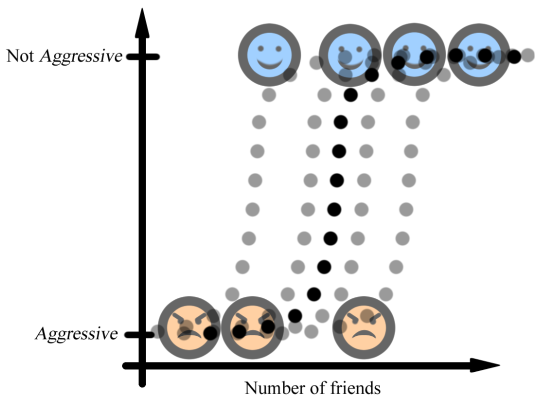


Figure 2.4: Sigmoid curve tested in different positions to find maximum likelihood.

Overall, this method helps to shrink real valued continuous inputs into a range of *(0,1)* being useful while dealing with probabilities and producing discrete binary outputs [32].

2.4.3 Random Forest

Random Forest (RF) is an ensemble of Decision Trees (DT). The spirit of this algorithm relies on the idea that if one decision tree is good, several randomly generated might be better. A decision tree is built from decision nodes (they correspond to features) and leaf nodes (correspond to class labels).

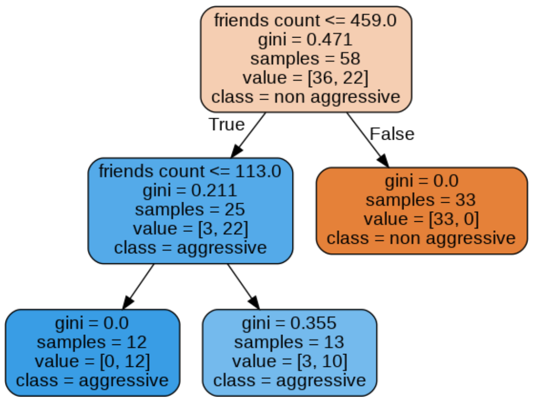


Figure 2.5: Decision Tree example.

Once we know a way to produce a tree we can create a forest, where each tree contributes to a final prediction in a majority vote for classification. The Random Forest Algorithm [33] is as follows:

- We start with N samples in our dataset (e.g. N posts in a social platform)
- Each sample has M features (e.g. date of creation, username, number of times shared, etc.)
- To create a tree, we take n random samples from the dataset and at each node, we select a subset m of M (where the size of m is proportional to $\sqrt{M}$) to find the best predictor.
- Repeat the procedure for next trees until you reach desired size of a forest

The name of the classifier comes from each tree being trained only on a random subsets of samples and features. An interesting aspect of these classifiers is that looking at a single Decision Tree, important features are likely to appear closer to the root of the tree, while unimportant features will often appear closer to the leaves (or not at all). It is therefore possible to get an estimate of a feature's importance by computing the average depth at which it appears across all trees in the forest [29].

2.5 Ensemble methods

An ensemble consists of putting together a group of predictors (classifiers or regressors) to aggregate their predictions, based on "the wisdom of the crowd", an idea that states that a collective opinion of a set of individuals is better than a single expert [29]. As an example, we know, as previously described, that a Random Forest is an ensemble of Decision Trees. Among the most popular ensemble methods are *bagging, boosting* and *stacking*.

2.5.1 *Bagging*

This technique consists of training a group of classifiers, where each one is fit using a random subset of the training set. These subsets are sampled with replacement [29].

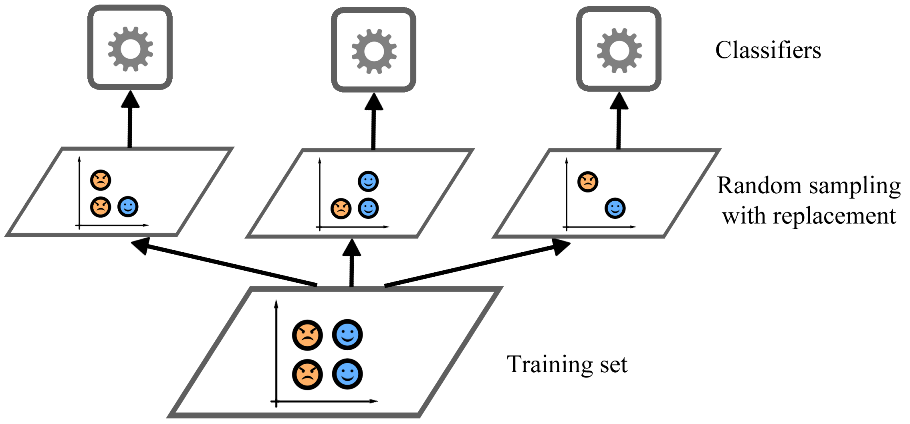


Figure 2.6: Bagging diagram.

2.5.2 *Boosting*

It refers to any ensemble method that combines a number of weak learners into a strong one, training predictors in order, so that the next classifier in this sequence can try to correct the weaknesses of its predecessor. The weakness of a model can be calculated either by adjusting weights of data points (*Adaboost*) or focusing on the difference between the prediction and the ground truth (*Gradient Boosting*) [34].

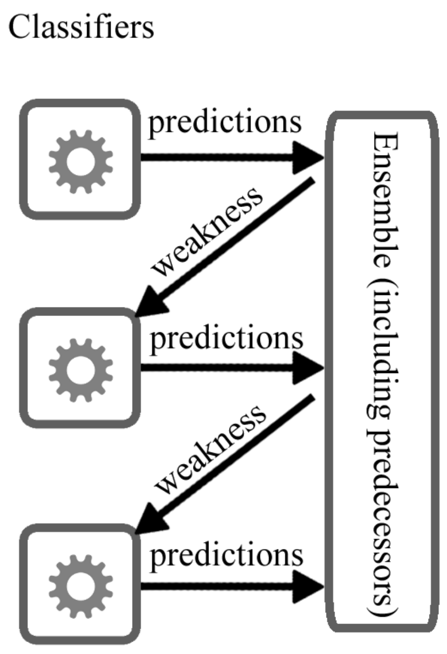


Figure 2.7: Boosting diagram.

2.5.3 *Stacking*

This technique takes the predictions made by a set of classifiers or regressors on a new instance as inputs for a final predictor. This means that, instead of making a decision as a result of majority of votes, the final predictor (also called blender) trains using the original target values and the predictions of its predecessors as features [29].

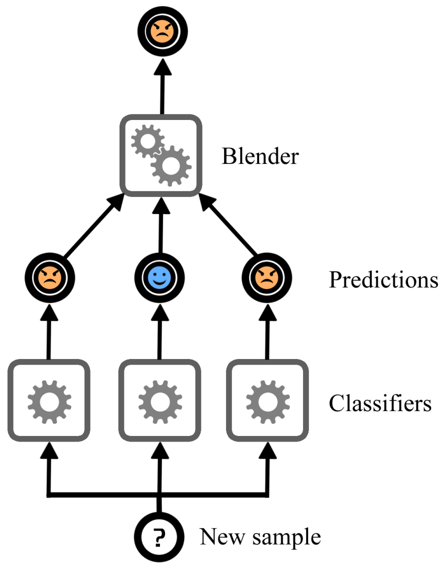


Figure 2.8: Stacking diagram.

In this research, a variant of stacking was used to develop classification models, the difference lies in that each classifier is trained using the same indexes of data points, but fed with different features. This implementation is further explained in section 3.

2.6 Automatic selection of hyperparameters for classifiers

A model hyperparameter is a characteristic of a model that is external to the model and whose value cannot be estimated from data. The value of the hyperparameter has to be set before the learning process begins. Grid-search is a technique used to find the optimal hyperparameters of a model which results in the most 'accurate' predictions [35]; by getting a set of valid values for each hyperparameter that is desired to tune, this strategy runs a series of loops, iterating through all possible combinations of hyperparameters and saving the configuration that achieves the best classification score.

2.7 Automatic selection of features

The motivation behind feature selection algorithms is to automatically select a subset of features that is most relevant to the problem. In this research, Sequential Feature Selector was used to look for the best combinations of features; this decision comes from the fact that, other feature selection strategies, such as Information Gain or Chi Squared Tests, are useful at discarding the individual features that are the most likely to be independent of class and therefore irrelevant for classification [36], as opposed to Sequential Feature Selector, that removes or adds one feature at the time based on the classifier performance until a feature subset of the desired size is reached [37].

The Sequential Feature Selector has four modalities:

- Sequential Forward Selection (SFS), starts with an empty set and begins to include one feature in every iteration.
- Sequential Backward Selection (SBS), starts with the given set of features and begins to exclude one of them in each iteration.
- Sequential Forward Floating Selection (SFFS).
- Sequential Backward Floating Selection (SBFS).

The floating variants have an additional exclusion or inclusion step to remove features once they were included (or excluded), so that a larger number of feature subset combinations can be sampled.

3

New datasets and baseline approaches

In this chapter, the steps to fulfill the objectives of the current proposal are presented in four parts as follows: the first one includes the details of preexisting annotated datasets and the analysis of the approaches proposed in their related work, the second part focuses on explaining the process of retrieving tweets to extend the datasets with metadata, the third part describes the experimental setting to establish baselines, and the last part covers the development of classification models using additional post and author features.

3.1 The acquisition of datasets containing offensive, hate speech and aggressive messages

To begin with, we identify which collections are available to download. We got access to annotated datasets that are used in related work, so a new approach can be developed with the purpose of proving this work's hypothesis. In this research, we focused on preexisting datasets that are collections of online posts gathered from Twitter (table 3.1), as the access to metadata is fairly easy in this platform. The next subsections provide more information about each dataset.

3.1.1 Davidson dataset

Created by Thomas Davidson in 2017, in this work [5] they use crowd-sourcing to label a sample of tweets having hate speech keywords into three categories: those containing hate speech, only

Table 3.1: Selected datasets.

Dataset	**Year of release**	**Labels**	**Language**
Davidson	2017	Hate Speech, Offensive Language, None	English
MEX-A3T	2018	Aggressive and Non-aggressive	Spanish
hatEval Eng-task	2018	Hateful and Not Hateful	English
hatEval Span-task	2018	Hateful and Not Hateful	Spanish
HASOC	2019	Hate-Offensive, None	English

offensive language, and those with neither. They trained a multi-class classifier to distinguish between the three different categories of tweets. Analysis of their predictions and errors shows that racist and homophobic tweets are more likely to be classified as hate speech, but sexist tweets are generally classified as offensive, and tweets without explicit hate words are also more difficult to sort out. In that research paper, they conclude that future work should take into account the context in the use of hate speech.

3.1.2 MEX-A3T Aggressive Detection Track dataset

Motivated by the lack of a Mexican corpus for aggressive detection, the team MEX-A3T created a dataset of tweets with hashtags related to topics of politics, sexism, homophobia and discrimination [38]. Each tweet of the corpus was labeled as aggressive or non-aggressive. They claim to have chosen Twitter as their source of data because the platform's openness and anonymity allows people to express offenses and aggressions among their opinions. The highest ranked team for this task in the 2018 edition developed a system for aggressive detection, EvoMSA [39] based on an ensemble approach with aggressive lexicons, and a classifier based on genetic programming to make the final predictions.

3.1.3 hatEval Subtask A datasets

Subtask A consists of a Hate Speech Detection task against Immigrants and Women in English and Spanish. For the Spanish part, the training and development set contains 5,000 tweets, (3,209 for the target women and 1,991 for immigrants), while for English it includes 10,000 tweets (5,000 for each target). They used the following approaches to collect tweets: monitoring potential victims of hate accounts, downloading the history of identified haters, and filtering Twitter streams with keywords [40]. For English, the highest ranked team trained a SVM model with RBF kernel using only the provided data, taking advantage of sentence embeddings from Google's Universal Sentence Encoder [41] as features. For Spanish, the highest ranked teams took advantage of linear kernel Support Vector Machines using a variety of text representations.

3.1.4 HASOC English Subtask A dataset

The creators of this collection identified topics for which many hate posts can be expected, then the data was sampled from Twitter and partially from Facebook using different hashtags and keywords [8]. For this dataset, the authors doesn't provide context or meta-data claiming that the inclusion of the latter might make the tasks somewhat unrealistic, and distribution of this data may pose legal issues; however this seems contradictory since distribution of text from tweets is also not allowed, as stated in the Developers Policy by Twitter [42]. Subtask A requires systems to classify tweets into two classes: Hate and Offensive (HOF) and Non- Hate and offensive. The highest ranked team in this challenge proposed a system based on an ordered neurons LSTM with an attention model [43]. The attention layer is used to assign weights to each word in a sentence, so the words that are biased towards hate speech and offensive language outstand.

3.2 The creation of extended versions of available datasets

As we mentioned earlier, we want to determine if contextual information is useful to identify offensive comments online, however, the datasets mentioned before do not include any other kind of information aside text messages from tweets and their corresponding label. To make up for this limitation, we setup a group of tools to look for more data associated with the samples provided in the datasets.

3.2.1 Metadata

Most studies perform data collection and subsequent publication of annotated datasets containing small portions of text. Granted, this constant focus on text, plus making it the only feature available to work with, has made possible the development of new NLP strategies. Take for instance ELMo [44], an evolution of Word Embeddings to create deep contextualized word representations. However, social media platforms allow their interactions to contain more information aside from the text written in messages. This additional info is called Metadata, it is defined as data that provides information about other data [45]. The next subsections describe what kind of data can be collected from users and tweets.

3.2.2 Tweet object

Also known as “status updates”, these objects represents tweets, each object has a list of fundamental properties. Table 3.2 displays the tweet attributes and descriptions [46] that were considered relevant for this task.

Table 3.2: Metadata of Tweet object.

Attribute	**Type**	**Description**
Retweet count	Integer	“Number of times this Tweet has been retweeted”.
Favorite count	Integer	“Indicates approximately how many times this Tweet has been liked by Twitter users”.
Date of creation	Date	“UTC time when this Tweet was created”. Used to extract the hour of the day.
Reply status	Boolean	Indicates if the Tweet is a reply to another Tweet
Quote status	Boolean	“Indicates whether this is a Quoted Tweet”.

3.2.3 User object

The User object contains Twitter User account metadata that describes the Twitter User referenced [47]. Table 3.3 displays the user attributes/metadata considered relevant for this task.

Table 3.3: Metadata of User object.

Attribute	**Type**	**Description**
Username	String	"The user who posted this Tweet".
Verified	Boolean	"...indicates that the user has a verified account".
Followers count	Integer	"The number of followers this account currently has".
Friends count	Integer	"The number of users this account is following".
Listed count	Integer	"The number of public lists that this user is a member of".
Favorites count	Integer	"The number of Tweets this user has liked in the account's lifetime".
Statuses count	Integer	"The number of Tweets (including retweets) issued by the user".
Default profile	Boolean	"...indicates that the user has not altered the theme or background of their user profile". Profile customization attribute.
Default profile image	Boolean	"...indicates that the user has not uploaded their own profile image and a default image is used instead". Profile customization attribute.
Created_at	Date	"The UTC datetime that the user account was created on Twitter". Determines how old the account is (calculating the number of days since the account was created).

3.2.4 Extending the datasets

By using the Standard Twitter API platform [48] together with additional libraries in Python such as GetOldTweets3 [49] and Twython [50] it is possible to search for every tweet in the collections selected (Table 3.1); if a message is still available online, it is retrieved as a tweet object, which includes properties of the post as well as information of the author of the tweet.

The process to extend a dataset is simple: first all the text messages are loaded with the proper encoding (to preserve the format of the original message), then a range of indexes is requested (since it is easier to retrieve data in batches), and then for every instance a query is made using the text to search for that tweet. If the tweet can be successfully recovered, a new file that contains the features discussed in tables 3.2 and 3.3 is created, after this, a manual concatenation is prepared to ensure correct matches between samples and data. These additional attributes are expected to better distinguish between labels and improve classification scores. While looking for tweets we took into account similarity between original tweet and query results (by retrieving at most 10 tweets for each query, comparing length of strings and character placement) and setting an "until-date" restriction for every dataset (e.g., if HASOC dataset was released on July 2019, we could only retrieve tweets issued until that date for this collection). A summarized version of this process is shown as pseudocode in algorithm 3.1. The extended parts of the datasets are available to the research community upon request. To protect the identity of users involved in these collections, screen names will be anonymized, and each request will require a signed privacy agreement. In case this poses a legal issue, we can facilitate the process that we followed to retrieve metadata.

Algorithm 3.1 Retrieve tweets metadata.

```
Input: d: dataset
Output: Metadata_array // Array of dimensions (tweets × features)
 1: until_date = date // different for each dataset,
 2: tweet_range = 10, max_ratio = 0, max_id = 0;
 3: For each tweet in d do
 4:   message = preprocessed text of tweet to resemble original format.
 5:   tweet_criteria = Set query search using message, until_date and tweet_range;
 6:   For i in range(tweet_range) do
 7:     query_result = query_using_tweet_criteria[i];
 8:     string_ratio = Ratio of sequence matcher between query_result and message;
 9:     If string_ratio > max_ratio then
10:       max_ratio = string_ratio;
11:       max_id = i;
12:     End if
13:   End for
14:   best_retrieved_tweet = query_using_tweet_criteria[max_id];
15:   // Metadata of best_retrieved_tweet is saved as single row into Metadata_array;
16: End for
17: Return Metadata_array
```

It is worth mentioning that, due to the nature of the tasks that led to the creation of these datasets, some tweets couldn't be recovered, possibly due to deletion of posts or suspended accounts. Table 3.4 shows the amount of tweets recovered per dataset.

3.3 The establishment of baselines for automatic detection of aggressive tweets using machine learning algorithms

We begin with an experimental setup and define initial classification scores for the subsets that were created in the previous step, being guided by related work and common approaches. We

Table 3.4: Recovered tweets for each dataset.

Dataset	**Year of release**	**Tweets**	**Recovered Tweets**	**Percentage Recovered**
Davidson	2017	24,783	12,200	49.23%
MEX-A3T	2018	7,700	5,139	66.74%
hatEval Eng	2018	10,000	5,311	53.11%
hatEval Span	2018	5,000	3,554	71.08%
HASOC	2019	7,005	5,925	84.58%

used Linear Support Vector Machine and Logistic Regression (a regression classifier), but different machine learning techniques can be utilized for this.

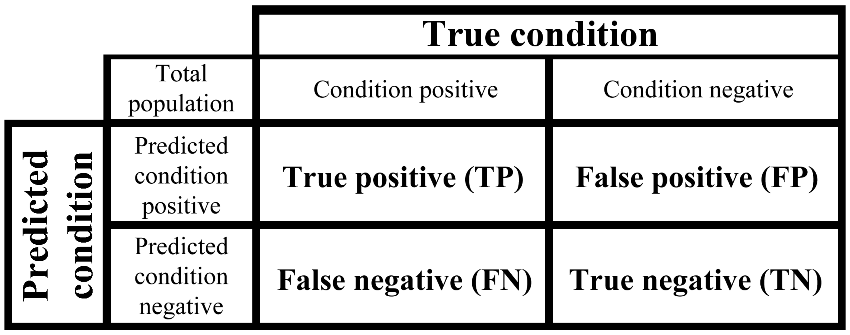
To evaluate the performance of the classifiers, the main classification metrics are reported, as defined in [51]:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

$$F1\text{-}score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} \tag{3.4}$$



		True condition	
	Total population	Condition positive	Condition negative
Predicted condition	Predicted condition positive	**True positive (TP)**	**False positive (FP)**
	Predicted condition negative	**False negative (FN)**	**True negative (TN)**

Figure 3.1: The four outcomes of a 2x2 confusion matrix.

Additionally, we also include F2-score, which weights recall higher than precision. This makes the F2-score more suitable in applications where it's more important to classify correctly as many positive samples as possible, rather than maximizing the number of correct classifications [52].

$$F2\text{-}score = (1 + 2^2) \cdot \frac{Precision \cdot Recall}{(2^2 \cdot Precision) + Recall} \tag{3.5}$$

This process is necessary as further statistical tests and comparisons will be made between the baselines and new results. Table 3.5 contains the hyperparameters used for *TfidfVectorizer* (an algorithm used to create a Bag of n-grams) in order to get optimal classification results for each dataset using this text representation as sole input for the classifiers.

Table 3.5: Hyperparameters of *TfidfVectorizer* used for each dataset

Dataset	**analyzer**	**ngram_range**	**use_idf**	**norm**	**min_df**
Davidson	word	(1,1)	False	L2	1
MEX-A3T	char	(3,4)	False	L2	1
hatEval Eng	word	(1,3)	False	L2	2
hatEval Span	word	(1,3)	False	L2	2
HASOC	word	(1,2)	False	L2	2

We conducted our experiments applying basic pre-processing steps on text data:

- All words were made lowercase.

- Emojis were converted into their text representation.
  (e.g., ":face_with_tears_of_joy:")
- Tweets were stripped from non-alphanumeric characters excluding some relevant symbols (#, @ and _).
- Every URL (occurrence of the sequence "http") was replaced with "weblink" to evenly represent references to external sources.

3.4 Model development

Once the baseline's configuration is established, different experimentation is conducted to perform classification, adding the extracted features from the new data retrieved for each tweet. The initial experimental setup began with the development of two classification models:

- A baseline, where metadata is absent.
- A model where metadata is included.

At first, these two approaches were enough to perform a comparison of classification scores, but we later found out that we could include two additional models, one to make comparisons more fair in terms of framework configuration, and one to try to enhance the use of features based on metadata. The classification models evolved as follows:

3.4.1 Baseline model (B/L)

This model represents the startup point of experiments regarding classifiers. Using the hyperparameters of table 3.5, we created a Bag of n-grams per dataset, serving as input for a Logistic Regression or Linear Support Vector Machine classifier. After being fit with the transformed text data, the classifier is asked to make label predictions on the test set, following a k-fold cross validation scheme. These are considered classic or traditional approaches [14], so the metadata remains untouched.

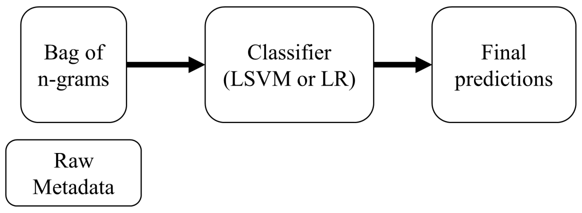


Figure 3.2: Diagram of B/L model

3.4.2 Addition of metadata (MD)

Since the data extracted from the text (e.g., Bag of Words and Characters n-grams) and the features of the tweets and authors are different in nature, we require techniques that can handle this set of attributes. In this research, several approaches were tested to combine the text and metadata features, including concatenating metadata to BoWs through One-Hot encoding, rounding values and feature scaling. In the end, an ensemble classifier, Random Forests, was used based on the methods mentioned in [16]. After the LR or LSVM classifier makes predictions in the form of label probabilities, these confidence levels are concatenated to their respective metadata to form a new feature vector, which is then fed to a RF classifier. An additional step to look for the best combination of attributes includes using an automatic feature selection algorithm. Seeing that tree-based classifiers don't require feature scaling or centering [29], the Random Forests can deal with raw metadata linked together to predictions of the text classifier as a stacking method.

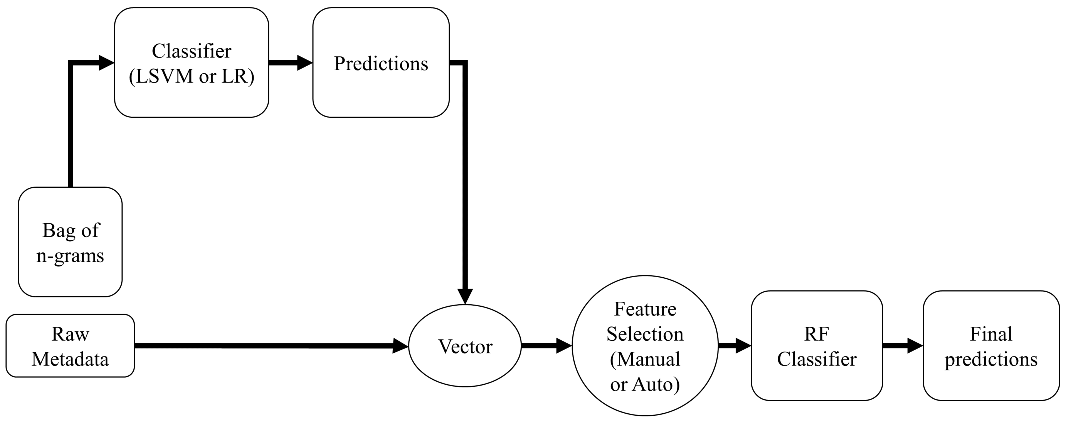


Figure 3.3: Diagram of MD model

3.4.3 Baseline predictions as inputs of Random Forest (B/L to RF)

When we noticed that the MD model was getting different classification scores from those of the B/L model, a new issue came up: how could we prove that better scores were the outcome of introducing metadata and not because we combined different classifiers?. This model sits in between B/L model and MD model, as it takes the predictions made by the first LR or LSVM classifier (using only the Bag of n-grams) as inputs for the Random Forest. This way, if there's any enhancement while comparing results between this model and MD model, we are able to attribute this improvement to the addition of metadata and not the change in final classifier.

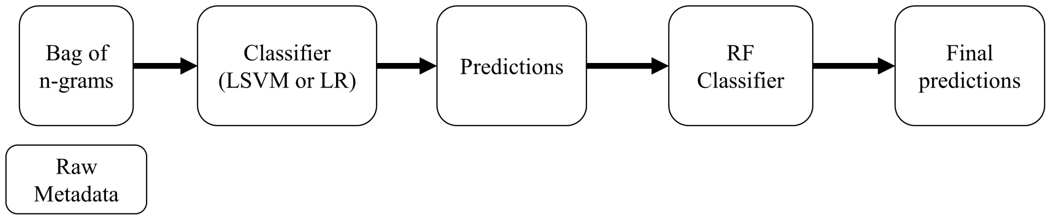


Figure 3.4: Diagram of B/L to RF model

3.4.4 Addition of metadata + Binning (MD + Bins)

After establishing that a RF classifier would make the final predictions for the MD model, we wanted to repurpose some of the ideas explored while designing this method. As an addon to the MD model, metadata can be discretized (i.e., continuous data transformed into intervals) to be similar to word frequency matrices in terms of values; this is useful as these new representations of metadata can be concatenated to Bags of n-grams, and become the input to LR / LSVM classifiers. We used sklearn's KBinsDiscretizer [52] to perform the discretization of metadata. The idea behind this last model is that a different classifier aside of RF could also train using the new feature vector.

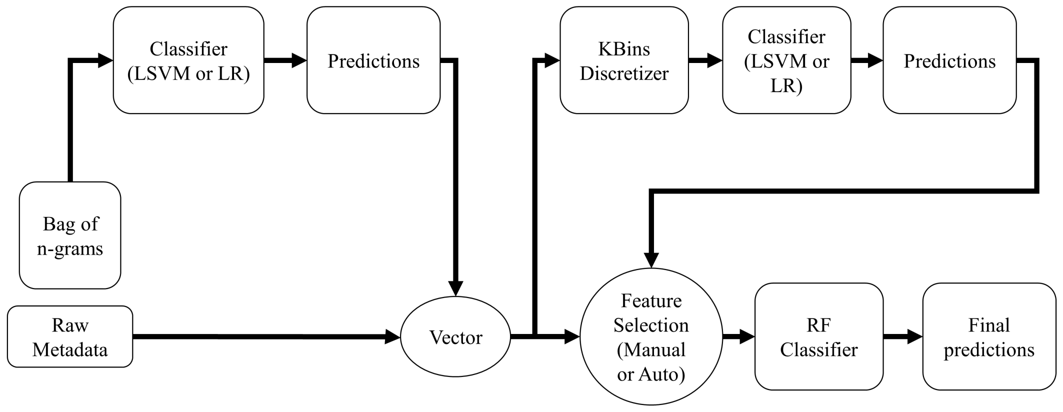


Figure 3.5: Diagram of MD + Bins model

4

Do authors contextual information help detect aggressiveness?

In this chapter the results obtained using the baselines and the classification methods that includes metadata are presented. The hyperparameters for the baseline classifiers were set as follows:

- LogisticRegression(C=1, random_state=0, solver='liblinear', max_iter=1000, multi_class='auto')
- LinearSVC(C=1, max_iter=1000) along with CalibratedClassifierCV

We used CalibratedClassifierCV with LinearSVC to get label probabilities, also interpreted as confidence levels. For the feature selection part in models with metadata, the parameter “k_features” was set as 'best' and “scoring” as 'f1_macro'.

We performed all modeling regarding the creation of term frequency feature matrices, feature selection, classifiers, cross validation and *GridSearch* using *scikit-learn* [53]. Table shows the hyperparameters explored by a *GridSearch* (GS) on all models involving a RandomForest as final classifier, the best set of hyperparameters for each model is specified in the next sections.

A Wilcoxon signed-rank test was performed to determine whether the difference between classification reports is statistically significant or not, that is, if paired results for all folds evaluated in a dataset belongs to different distributions between classifiers. For each table displaying classification reports, a highlighted cell for models using metadata indicates that that particular score is statistically significant against scores from baseline models.

Table 4.1: Hyperparameters explored for each RF Classifier in the *GridSearch*

Parameter	Range
n_estimators	[10, 50, 100, 150, 200]
max_features	['auto', None]
max_depth	[3, 4, 5, 6]
criterion	['gini', 'entropy']

To complement the classification reports with the most highlighted results per dataset, a table detailing the importance of features according to the final classifier of models with metadata is provided in section 4.2.

4.1 Results

4.1.1 Davidson dataset

Table 4.2 displays the performance of models using LR as B/L classifier, where MD + Bins model achieves the best results for all measures but Precision, benefiting from the inclusion of discretized metadata.

Table 4.2: Classification report in Davidson DS using LR as B/L [Mean (%) ± STDEV]

Model	Accuracy	Precision	Recall	F1-Score	F2-Score
LR as B/L	88.84 ± 0.58	76.15 ± 5.67	58.22 ± 1.23	60.58 ± 1.62	58.89 ± 1.32
B/L to RF	89.84 ± 0.76	72.95 ± 3.47	67.18 ± 1.95	68.41 ± 2.14	67.47 ± 1.97
MD	89.93 ± 0.93	73.23 ± 3.58	68.38 ± 2.04	69.80 ± 2.52	68.80 ± 2.21
MD + Bins	90.07 ± 0.83	73.83 ± 3.45	69.00 ± 2.26	70.41 ± 2.63	69.42 ± 2.39

Table 4.3 display the performance of models using LSVM as B/L classifier. In this case, the Bins addon is absent as it didn't improve scores of MD model.

Table 4.3: Classification report in Davidson DS using LSVM as B/L [Mean (%) ± STDEV]

Model	Accuracy	Precision	Recall	F1-Score	F2-Score
LSVM as B/L	90.30 ± 0.90	75.67 ± 4.26	64.55 ± 2.30	66.65 ± 2.80	65.12 ± 2.45
B/L to RF	90.05 ± 0.83	73.24 ± 3.00	70.54 ± 2.89	71.34 ± 2.80	70.78 ± 2.87
MD	90.14 ± 0.79	74.00 ± 2.35	71.07 ± 1.94	72.34 ± 2.06	71.54 ± 1.98

4.1.2 hatEval English dataset

Table 4.4 shows the performance of models using LR as B/L classifier. Similarly to table 4.2 in Davidson DS, MD + Bins model achieves the best results in all measures except in Precision.

Table 4.4: Classification report in hatEval Eng DS using LR as B/L [Mean (%) ± STDEV]

Model	Accuracy	Precision	Recall	F1-Score	F2-Score
LR as B/L	78.94 ± 1.02	78.87 ± 1.26	74.95 ± 1.17	76.04 ± 1.20	75.21 ± 1.19
B/L to RF	78.83 ± 0.86	77.46 ± 0.93	77.04 ± 1.00	77.18 ± 0.91	77.08 ± 0.96
MD	80.03 ± 1.04	78.74 ± 1.15	78.19 ± 1.09	78.42 ± 1.09	78.27 ± 1.09
MD + Bins	80.05 ± 1.02	78.76 ± 1.14	78.23 ± 1.06	78.45 ± 1.06	78.31 ± 1.05

Again, the Bins addon is not used in table 4.5 as it didn't increase the performance of the MD model.

Table 4.5: Classification report in hatEval Eng DS using LSVM as B/L [Mean (%) ± STDEV]

Model	Accuracy	Precision	Recall	F1-Score	F2-Score
LSVM as B/L	79.57 ± 1.16	78.77 ± 1.35	76.56 ± 1.29	77.33 ± 1.31	76.79 ± 1.30
B/L to RF	79.36 ± 1.08	78.05 ± 1.15	77.28 ± 1.34	77.60 ± 1.26	77.39 ± 1.32
MD	80.11 ± 1.21	78.82 ± 1.29	78.27 ± 1.44	78.51 ± 1.37	78.35 ± 1.42

4.1.3 hatEval Spanish dataset

As shown in tables 4.6 and 4.7, the MD + Bins model achieves the best scores for all measures.

Table 4.6: Classification report in hatEval Span DS using LR as B/L [Mean (%) ± STDEV]

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**F2-Score**
LR as B/L	78.47 ± 1.55	79.21 ± 1.59	75.25 ± 1.85	76.14 ± 1.90	75.39 ± 1.91
B/L to RF	78.96 ± 1.93	78.10 ± 2.01	77.91 ± 2.28	77.95 ± 2.11	77.91 ± 2.20
MD	79.72 ± 1.93	78.83 ± 2.02	78.96 ± 2.07	78.87 ± 2.02	78.91 ± 2.04
MD + Bins	80.25 ± 1.66	79.48 ± 1.80	79.06 ± 1.72	79.23 ± 1.73	79.12 ± 1.72

It is also worth noting that, while using LSVM as B/L classifier, both MD models show to be more stable with lower standard deviations.

Table 4.7: Classification report in hatEval Span DS using LSVM as B/L [Mean (%) ± STDEV]

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**F2-Score**
LSVM as B/L	80.79 ± 2.16	80.32 ± 2.31	79.08 ± 2.32	79.54 ± 2.32	79.23 ± 2.33
B/L to RF	80.60 ± 2.03	79.92 ± 2.23	79.32 ± 2.06	79.56 ± 2.09	79.40 ± 2.06
MD	80.84 ± 1.79	80.16 ± 1.92	79.55 ± 1.85	79.80 ± 1.86	79.64 ± 1.85
MD + Bins	81.03 ± 1.81	80.33 ± 1.96	79.80 ± 1.85	80.02 ± 1.87	79.88 ± 1.85

4.1.4 HASOC dataset

For this dataset, adding metadata didn't help to improve scores of LR and LSVM classifiers (tables 4.8 and 4.9). The Bins addon was discarded for both baselines. We believe that perhaps this happened because, judging by the difference in topics, the English dataset was sampled from at least three different regions (USA, UK and India); if this is the case, then the dissimilarity of cultures could have prevented the existance of social media patterns among the tweets based on metadata. However, this is a conjecture, as the specifics of how the dataset was sampled are not given in [8].

Table 4.8: Classification report in HASOC DS using LR as B/L [Mean (%) ± STDEV]

Model	Accuracy	Precision	Recall	F1-Score	F2-Score
LR as B/L	69.31 ± 0.99	69.20 ± 2.71	57.83 ± 1.03	55.94 ± 1.31	56.22 ± 1.10
B/L to RF	68.84 ± 1.28	65.07 ± 1.54	63.76 ± 1.53	64.13 ± 1.55	63.85 ± 1.55
MD	68.76 ± 1.36	65.02 ± 1.59	63.95 ± 1.51	64.29 ± 1.54	64.04 ± 1.53

Table 4.9: Classification report in HASOC DS using LSVM as B/L [Mean (%) ± STDEV]

Model	Accuracy	Precision	Recall	F1-Score	F2-Score
LSVM as B/L	70.04 ± 0.74	68.01 ± 1.68	60.44 ± 0.66	60.16 ± 0.78	59.73 ± 0.69
B/L to RF	67.47 ± 1.23	63.63 ± 1.40	62.94 ± 1.36	63.18 ± 1.37	63.01 ± 1.36
MD	67.31 ± 1.00	63.48 ± 1.21	62.88 ± 1.32	63.09 ± 1.28	62.94 ± 1.31

4.1.5 MEX-A3T dataset

Table 4.10 shows the performance of models using LR as B/L classifier, where MD model achieves the best results for Recall and F-scores.

Table 4.10: Classification report in MEX-A3T DS using LR as B/L [Mean (%) ± STDEV]

Model	Accuracy	Precision	Recall	F1-Score	F2-Score
LR as B/L	79.65 ± 1.05	79.29 ± 1.46	74.81 ± 1.31	76.09 ± 1.30	75.13 ± 1.34
B/L to RF	79.12 ± 1.23	77.27 ± 1.31	76.59 ± 1.67	76.84 ± 1.49	76.67 ± 1.61
MD	79.51 ± 1.07	77.68 ± 1.17	77.12 ± 1.33	77.33 ± 1.21	77.19 ± 1.28

For the LSVM as B/L classifier variant, MD model barely outperforms B/L to R/F model (table 4.11). The Bins addon wasn't useful for any baselines.

Table 4.11: Classification report in MEX-A3T DS using LSVM as B/L [Mean (%) ± STDEV]

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**F2-Score**
LSVM as B/L	80.50 ± 0.68	79.36 ± 0.99	76.88 ± 0.74	77.77 ± 0.71	77.16 ± 0.73
B/L to RF	79.88 ± 0.95	77.99 ± 1.04	78.01 ± 0.93	77.98 ± 0.95	77.99 ± 0.93
MD	80.05 ± 1.09	78.19 ± 1.23	78.10 ± 1.03	78.12 ± 1.09	78.10 ± 1.05

4.2 Feature importance

As we mentioned in subsection 2.4.3, Random Forests classifiers have the capability to rate the importance of a feature based on the position of said feature in the trees of the forest. This allowed us to know the importance of the attributes used as inputs for the R/F classifiers at the end of our models.

For simplification purposes, in Metadata models, the first classifier (the one that recieves a Bag of n-grams as input) is called "Classifier 1", and, when present, the second classifier (that sees discretized metadata and prior label probabilities) is called "Classifier 2". Table 4.12 displays the name of the features and the abbreviations that we used in the feature importance reports shown in the next subsections.

Table 4.12: Abbreviation of features used in importance reports

Feature	**Abbreviation**
Classifier 1 probability for Aggressive label	CLF1 AG
Classifier 1 probability for HS label	CLF1 HS
Classifier 1 probability for NOT label	CLF1 N
Classifier 2 probability for HS label	CLF2 HS
Classifier 2 probability for NOT label	CLF2 N
Classifier 2 probability for OF label	CLF2 OF
User Account Age (in days)	DAYS
User Account Follower count	FLWR_C
User Account Friend count	FRND_C
Hour of the day in which the tweet was posted	HOUR
User Account Listed count	LIST_C
Tweet Quote Status	QUOTE
Tweet Reply Status	REPLY
Tweet Retweet count	RTW_C
User Account Status count	STAT_C
Tweet Favorite count	TFAV_C
User Account Default Image	UDIMG
User Account Default Profile	UDPROF
User Account Favorites count	UFAV_C

4.2.1 Davidson dataset

Using LR as B/L, the most important features (below Classifiers label probabilities), as shown in table 4.13 are:

- User Account Favorites count (appears in 9 folds).
- User Account Age (appears in 6 folds).

- Hour of the day (appears in 6 folds).
- User Account Status count (appears in 5 folds)
- User Account Follower count (appears in 4 folds).

Since an automatic feature selection method was employed on this scenario, the attributes could be considered or not for a fold depending on the SFS algorithm. Hence, the feature importance is not only hinted by the values given by the RF classifier but also by the number of times they appear among all folds. As a side note, four out of these top five features are part of the user's information (only the hour of the day pertains to the tweet's metadata).

Table 4.13: Feature importance in MD + Bins model (LR as B/L in Davidson DS) [Percent (%)]

Feature	**Fold 1**	**Fold 2**	**Fold 3**	**Fold 4**	**Fold 5**	**Fold 6**	**Fold 7**	**Fold 8**	**Fold 9**	**Fold 10**
CLF1 N	81.06	82.60	80.86	82.19	82.16	80.81	83.09	82.83	82.41	81.65
CLF1 HS	15.28	15.90	14.23	16.41	16.40	15.22	16.69	16.52	17.02	15.80
CLF2 N	2.20					0.50				
CLF2 HS	0.94	1.02	0.34	0.94	0.93					2.05
UFAV_C	0.19	0.14	0.12	0.08	0.13	0.15		0.21	0.12	0.14
DAYS	0.15	0.10	0.12	0.12	0.08			0.10		
FRND_C	0.09				0.06					
HOUR	0.07	0.11	0.07		0.05				0.11	0.14
TFAV_C	0.02	0.05	0.05			0.03		0.10	0.03	0.07
RTW_C	0.01	0.01	0.00	0.03	0.03				0.01	
STAT_C				0.13	0.15		0.21	0.13	0.19	
FLWR_C				0.11				0.09	0.06	0.14
LIST_C		0.05	0.02					0.02	0.04	
CLF2 OF			4.18			3.27				

4.2.2 hatEval English dataset

For this case, the most important features (below Classifiers label probabilities) were User Account related metadata:

- User Account Follower count.
- User Account Favorites count.
- User Account Status count.
- User Account Friend count

There's an additional aspect of this: these four features are also considered relevant when carrying out the same analysis but with LSVM as B/L. The importance percentage of these attributes goes from almost 3% to above 1%. Tweet metadata importance is rated below 0.5%. The complete importance values are displayed in table 4.14.

Table 4.14: Feature importance in MD + Bins model (LR as B/L in hatEval Eng DS) [Percent (%)]

Feature	**Fold 1**	**Fold 2**	**Fold 3**	**Fold 4**	**Fold 5**	**Fold 6**	**Fold 7**	**Fold 8**	**Fold 9**	**Fold 10**
CLF1 HS	48.38	49.41	48.78	49.57	50.17	49.40	49.88	49.19	49.26	49.23
CLF1 N	38.15	38.86	38.25	39.58	39.68	38.58	39.63	38.94	38.95	39.02
CLF2 N	2.94	1.88	2.47	1.42	1.00	1.91	0.93	2.16	1.81	1.87
CLF2 HS	2.75	2.04	2.37	1.53	1.24	1.71	1.05	2.08	2.14	1.81
FLWR_C	2.10	2.32	1.95	2.31	2.16	2.61	2.25	2.19	1.99	2.00
UFAV_C	1.90	1.64	2.19	1.63	1.74	1.90	1.74	1.68	1.95	1.74
STAT_C	1.57	1.75	1.85	1.84	1.62	1.69	1.80	1.60	1.69	2.29
FRND_C	1.41	1.39	1.52	1.30	1.43	1.36	1.67	1.24	1.48	1.24
TFAV_C	0.44	0.41	0.37	0.48	0.56	0.47	0.51	0.48	0.43	0.38
REPLY	0.15	0.09	0.11	0.19	0.19	0.14	0.23	0.14	0.16	0.16
QUOTE	0.14	0.19	0.06	0.10	0.07	0.16	0.16	0.17	0.09	0.15
UDIMG	0.08	0.04	0.07	0.06	0.12	0.06	0.15	0.12	0.03	0.10

4.2.3 hatEval Spanish Dataset

Table 4.15 shows that the most important features in this dataset using LR as B/L (below Classifiers label probabilities) are:

- User Account Follower count.
- User Account Listed count.
- Tweet Quote Status.
- User Account Default Profile.
- Tweet Reply Status.

These features also appear in the analysis when LSVM is used as B/L, however their importance is much lower (all of them are below 0.5%).

Table 4.15: Feature importance in MD + Bins model (LR as B/L in hatEval Span DS) [Percent (%)]

Feature	**Fold 1**	**Fold 2**	**Fold 3**	**Fold 4**	**Fold 5**	**Fold 6**	**Fold 7**	**Fold 8**	**Fold 9**	**Fold 10**
CLF1 N	44.81	45.14	48.32	44.22	47.59	47.83	43.99	48.96	44.00	43.94
CLF1 HS	37.27	37.60	36.56	38.63	37.92	36.45	38.20	36.39	40.58	39.42
CLF2 HS	9.20	9.58	8.88	9.40	8.54	9.57	8.04	7.32	9.06	8.43
CLF2 N	3.17	2.47	1.48	2.55	1.60	1.87	4.60	1.56	1.67	3.33
FLWR_C	2.85	2.42	2.16	2.59	2.27	1.87	2.13	2.49	2.19	2.43
LIST_C	1.82	1.58	1.69	1.68	1.37	1.28	1.78	1.94	1.49	1.74
QUOTE	0.41	0.62	0.34	0.23	0.29	0.41	0.34	0.73	0.53	0.26
UDPROF	0.29	0.38	0.39	0.23	0.25	0.44	0.55	0.41	0.25	0.23
REPLY	0.18	0.17	0.18	0.38	0.14	0.19	0.25	0.14	0.16	0.17
UDIMG	0.00	0.04	0.00	0.08	0.01	0.08	0.12	0.06	0.07	0.04

4.2.4 MEX-A3T dataset

For this instance, the most important features (below Classifiers label probabilities) according to table 4.16 are:

- User Account Follower count.
- User Account Status count.
- User Account Favorites count.
- Tweet Reply Status.
- Hour of the day.
- User Account Default Profile.
- Tweet Favorite count.

In this dataset, despite not benefiting from the Bins addon, we can appreciate more balance between the user and tweet metadata included; the RF classifier considered seven metadata attributes with importance values between 2% and 0.5%.

4.2.5 Additional aspects

- The spanish datasets are the only ones where RF considers the User Account Default Profile attribute as helpful, indicating that perhaps profile customization options play a different role depending on the language and manners associated with it.
- User Account Follower count is the only metadata feature to be considered on all datasets where the Metadata model displays statistically significant results. User Account Status count comes behind, appearing in three feature importance reports.
- Since the Username feature was added to the Bag of n-grams representation, its importance could not be measured by the RF classifiers, however, it showed to improve classification scores when included in the Davidson and hatEval datasets. This is reasonable, considering that in the datasets mentioned before, many examples of a label (e.g., Hateful, Offensive) were tied to only a few users, with multiple instances for each one of them.

Table 4.16: Feature importance in MD + Bins model (LR as B/L in MEX-A3T DS) [Percent (%)]

Feature	**Fold 1**	**Fold 2**	**Fold 3**	**Fold 4**	**Fold 5**	**Fold 6**	**Fold 7**	**Fold 8**	**Fold 9**	**Fold 10**
CLF1 N	49.62	48.80	49.43	47.01	49.62	48.97	48.10	48.66	47.84	46.28
CLF1 AG	43.54	43.97	43.50	45.87	43.11	43.43	44.98	43.88	44.38	46.57
FLWR_C	1.43	1.51	1.28	1.38	1.48	1.41	1.38	1.58	1.73	1.21
STAT_C	1.28	1.24	1.20	1.41	1.17	1.44	1.21	1.26	1.37	1.33
UFAV_C	1.05	0.92	0.78	0.89	0.96	0.94	0.94	0.91	1.08	1.05
REPLY	1.01	1.12	1.17	1.00	1.09	1.21	1.05	1.30	1.14	1.26
HOUR	0.69	0.88	0.82	0.79	0.82	0.80	0.79	0.69	0.86	0.72
UDPROF	0.51	0.65	0.75	0.75	0.88	0.79	0.67	0.75	0.66	0.68
TFAV_C	0.39	0.43	0.56	0.42	0.46	0.47	0.43	0.52	0.46	0.47
QUOTE	0.21	0.10	0.19	0.16	0.15	0.27	0.15	0.20	0.12	0.12
RTW_C	0.18	0.25	0.22	0.21	0.19	0.17	0.23	0.16	0.25	0.19
UDIMG	0.08	0.14	0.10	0.11	0.08	0.09	0.08	0.07	0.11	0.13

4.3 New and fixed mistakes

As part of the analysis performed on the results, we thought it would be interesting to display a few examples of when the MD models are able to fix mistakes made by B/L classifiers, as well as new mistakes that didn't appear initially.

Table 4.17: Examples of fixed and new mistakes by MD model (LSVM as B/L in Davidson DS)

Ground truth	**B/L**	**MD**	**Text**
2 (None)	Prediction: 0 Confidence: 91%	Prediction: 2 Confidence: 87%	"@user_mention marshall law whatever you **coon**"
"	Prediction: 0 Confidence: 49%	Prediction: 2 Confidence: 85%	"lil kim really made herself **trash** smh"
"	Prediction: 0 Confidence: 61%	Prediction: 2 Confidence: 71%	"this store is so **redneck**"
"	Prediction: 2 Confidence: 74%	Prediction: 1 Confidence: 51%	"all i did was said her weed was **trash**"
1 (Offensive)	Prediction: 1 Confidence: 70%	Prediction: 2 Confidence: 71%	"@user_mention like hov said we all **ghetto** b lol"
"	Prediction: 1 Confidence: 68%	Prediction: 2 Confidence: 56%	"i know these **niglets** heard me screaming"

Table 4.18: Examples of fixed and new mistakes by MD model (LSVM as B/L in hatEval Eng DS)

Ground truth	**B/L**	**MD**	**Text**
1 (Hateful)	Prediction: 0 Confidence: 75%	Prediction: 1 Confidence: 94%	"**Women** are children they require men in their lives to set boundaries and teach them lessons"
"	Prediction: 0 Confidence: 57%	Prediction: 1 Confidence: 99%	"**Women** are beautiful to me when they are silent why do they have to speak up and ruin things"
"	Prediction: 0 Confidence: 84%	Prediction: 1 Confidence: 94%	"It's a good thing I always wear a glove on my left hand because if I ever had to touch hands with a **woman** my IQ would totally drop to 0 lol"
0 (Not Hateful)	Prediction: 0 Confidence: 57%	Prediction: 1 Confidence: 85%	"New mexican president to create **border** force to stop **illegal immigrants drugs** from central america weblink surprise"
"	Prediction: 0 Confidence: 50%	Prediction: 1 Confidence: 98%	"After arriving in the EU **refugees** make **protest** and want money welcome or not weblink"
"	Prediction: 0 Confidence: 68%	Prediction: 1 Confidence: 85%	"200 **black suspects refugees** stop the german police to bring one person out of the country after police have to let him go under pressure he escaped in the underground weblink"

Table 4.19: Examples of fixed and new mistakes by MD model (LSVM as B/L in hatEval Span DS)

Ground truth	**B/L**	**MD**	**Text**
1 (Hateful)	Prediction: 0 Confidence: 56%	Prediction: 1 Confidence: 82%	"jajajajaja **zorra** sos vos mi amor por teclado somos todas malas eh pero en la cara nunca nada vos ridícula face_blowing_a_kiss"
"	Prediction: 0 Confidence: 61%	Prediction: 1 Confidence: 73%	"Pero si son **árabes** chaval si ven a sus padres **degollar animales** to los días weblink"
"	Prediction: 0 Confidence: 69%	Prediction: 1 Confidence: 52%	"¿Cuántas veces se ríe una **mujer** con un chiste? tres veces: cuando se lo cuentan, cuando se lo explican y cuando lo entiende"
0 (Not Hateful)	Prediction: 0 Confidence: 54%	Prediction: 1 Confidence: 74%	"Horrible nadie ni nada te obliga a **violar** a weblink de los refugiados del aquarius me obligaron a **violar a menores y animales** durante meses weblink vía @user_mention"
"	Prediction: 0 Confidence: 60%	Prediction: 1 Confidence: 67%	"@user_mention @user_mention @user_mention @user_mention @user_mention **joder** tío cállate que soy un **puto** adolescente de **mierda** queriendo llamar la **puta** atención"
"	Prediction: 0 Confidence: 52%	Prediction: 1 Confidence: 58%	"Esto de ser **mexicano indocumentado** no es divertido"

5

Conclusions and future work

To summarize, in this research we wanted to explore the inclusion of other features aside from those extracted from text messages in aggressive detection tasks. We retrieved available metadata for five datasets sampled from Twitter, creating new extended versions of these collections in hopes of using this additional information to bring context to text comments. We developed four classification models, and performed comparisons of classification scores between methods with and without metadata-based features to determine if they are useful for the task.

5.1 Conclusions

The new data added to the extended datasets includes Tweet and User attributes, providing information that describes users, interactions between them and the impact of messages inside the social platform. We consider that it is important to analyze all the information at our disposal if we want better automatic systems to detect Hate Speech and offensive posts online.

We created extended versions of five datasets from Twitter to conduct our investigation:

1. Davidson.
2. MEX-A3T Aggressive Detection Track.
3. hatEval Development-English-task A set.
4. hatEval Development-Spanish-task A set.

5. HASOC-English-Subtask A.

Showing up next are the answers to the research questions raised in section 1.3, obtained after carrying out our investigation:

What are the most successful representations used for automatic detection of aggressive comments?

Despite being simple text representations, Bag of Words and Bag of Character n-grams are still competitive approaches at aggressive detection tasks.

Is contextual information important to improve aggressive content detection?

As shown in previous feature importance tables, the vast majority of contribution percentage to decision making by the final classifier goes to classifiers confidence levels. Our proposed classification methods, the models that include metadata, won´t work independently of text representations, as metadata is not well suited to be the sole input of a classifier. Instead, the Metadata models are intended to be addons, going hand in hand with text and even adapt to it by using automatic selection of features.

What kind of profiling provides greater advantages for the task of identifying aggressive comments?

Although this research started using author profiling based on age, gender, occupation, location and unsupervised clustering, these traits didn't help to improve classification scores. Greater advantages can be seen by using profiling based on user, message and network features. Adding these features shows that some context can be given to text-only representations and obtain better results, mainly in Recall, F1 and F2 scores.

Additional analysis of results indicates that subtle aggressive messages that were misclassified by baseline models are classified correctly by incorporating metadata. Some new misclassified instances by models with metadata are created due to the use of words associated with trigger topics (politics, religion, swearing), but overall, corrected mistakes by models with metadata outnumber their new mistakes.

Although far from solving the issue of automatic detection of hateful content, new models that consider metadata may be a step in the right direction to tackle this problem.

5.2 Future work

1. To perform more tests as new datasets become available. This will allow us to further study the importance of metadata along with their relevance, as social interactions online are dynamic processes and adaptability of aggressive detection systems is required.

2. To try more classifiers as baselines for the proposed models, specifically those based on deep learning, as these methods are outperforming classical approaches.

3. To search for more potential features that could bring context to texts for this tasks. As stated before, it is more difficult to make good decisions the less information we have. It would be interesting to keep exploring about what kind of data we could use to our advantage to build better classifiers.

References

[1] F. Sebastiani, “A tutorial on automated text categorisation,” *Proceedings of ASAI-99, 1st Argentinian Symposium ...*, pp. 1–25, 1999. [Online]. Available: http://web.iiit.ac.in/~jawahar/PRA-03/textCat.pdf

[2] “The pragmatics of swearing,” *Journal of Politeness Research. Language, Behaviour, Culture*, vol. 4, no. 2, pp. 267–289, 2008. [Online]. Available: https://www.degruyter.com/view/j/jplr.2008.4.issue-2/jplr.2008.013/jplr.2008.013.xml

[3] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang, “Abusive Language Detection in Online User Content,” *Proceedings of the 25th International Conference on World Wide Web - WWW ’16*, pp. 145–153, 2016. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2872427.2883062

[4] Y. Chen, “Detecting Offensive Language in Social Medias for Protection of Adolescent Online Safety,” no. December, pp. 1–34, 2011.

[5] T. Davidson, D. Warmsley, M. Macy, and I. Weber, “Automated hate speech detection and the problem of offensive language,” *Proceedings of the 11th International Conference on Web and Social Media, ICWSM 2017*, no. Icwsm, pp. 512–515, 2017.

[6] W. Alorainy, P. Burnap, H. Liu, and M. Williams, “The Enemy Among Us: Detecting Hate Speech with Threats Based ’Othering’ Language Embeddings,” no. January 2019, 2018. [Online]. Available: http://arxiv.org/abs/1801.07495

[7] A. Arango, J. Pérez, and B. Poblete, “Hate speech detection is not as easy as you may think: A closer look at model validation,” *SIGIR 2019 - Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 45–53, 2019.

[8] S. Modha, T. Mandl, P. Majumder, and D. Patel, "Overview of the HASOC track at FIRE 2019: Hate speech and offensive content identification in indo-european languages," *CEUR Workshop Proceedings*, vol. 2517, pp. 167–190, 2019.

[9] Z. Waseem and D. Hovy, "Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter," *Proceedings of the NAACL Student Research Workshop*, pp. 88–93, 2016. [Online]. Available: http://aclweb.org/anthology/N16-2013

[10] J. M. Struß, M. Siegel, M. Wiegand, and M. Klenner, "Overview of GermEval Task 2 , 2019 Shared Task on the Identification of Offensive Language," no. Konvens, pp. 352–363, 2019.

[11] H. Mubarak, K. Darwish, and W. Magdy, "Abusive Language Detection on Arabic Social Media," pp. 52–56, 2017.

[12] S. Malmasi and M. Zampieri, "Challenges in discriminating profanity from hate speech," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 30, no. 2, pp. 187–202, 2018.

[13] C. V. Hee, E. Lefever, B. Verhoeven, J. Mennes, B. Desmet, G. D. Pauw, and W. Daelemans, "Detection and Fine-Grained Classification of Cyberbullying Events," *International Conference Recent Advances in Natural Language Processing (RANLP)*, pp. 672–680, 2015.

[14] E. Shushkevich and J. Cardiff, "Automatic misogyny detection in social media: A survey," *Computacion y Sistemas*, vol. 23, no. 4, pp. 1159–1164, 2019.

[15] C. Nagpal and K. Singhal, "Twitter User Classification using Ambient Metadata," *arXiv preprint arXiv:1407.8499*, 2014. [Online]. Available: http://arxiv.org/abs/1407.8499

[16] D. Chatzakou, N. Kourtellis, J. Blackburn, E. De Cristofaro, G. Stringhini, and A. Vakali, "Mean Birds: Detecting Aggression and Bullying on Twitter," 2017. [Online]. Available: http://arxiv.org/abs/1702.06877

[17] E. Petersen, "$\rightarrow$ La velocidad de lectura promedio en palabras por minuto | Geniolandia," 2018, (Accessed on 2018-05-13). [Online]. Available: https://www.geniolandia.com/13181761/la-velocidad-de-lectura-promedio-en-palabras-por-minuto

[18] C. Purcell, "Cecilia Purcell | Psicologia Infanto-Juvenil Para Todos." 2009, (Accessed on 2018-05-13). [Online]. Available: http://www.ceciliapurcell.cl/articulo01.html

[19] "Twitter Usage Statistics - Internet Live Stats," (Accessed on 2018-08-22). [Online]. Available: http://www.internetlivestats.com/twitter-statistics/

[20] "• Twitter by the Numbers (2018): Stats, Demographics & Fun Facts," (Accessed on 2018-08-22). [Online]. Available: https://www.omnicoreagency.com/twitter-statistics/

[21] "Frequently Asked Questions | 280daily," (Accessed on 2018-08-22). [Online]. Available: https://280daily.com/faq

[22] "Bullying On Twitter: Researchers Find 15,000 Bully-Related Tweets Sent Daily (STUDY) | HuffPost," (Accessed on 2019-12-02). [Online]. Available: https://www.huffpost.com/entry/bullying-on-twitter_n_1732952

[23] J. M. Xu, K. S. Jun, X. Zhu, and A. Bellmore, "Learning from bullying traces in social media," *NAACL HLT 2012 - 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, pp. 656–666, 2012.

[24] "Cyber Bullying Statistics - Bullying Statistics," (Accessed on 2019-12-02). [Online]. Available: http://www.bullyingstatistics.org/content/cyber-bullying-statistics.html

[25] E. Wulczyn, N. Thain, and L. Dixon, "Ex Machina: Personal Attacks Seen at Scale," pp. 1–9, 2016. [Online]. Available: http://arxiv.org/abs/1610.08914

[26] K. Aas and L. Eikvil, "Text categorisation: A survey." 1999.

[27] C. C. Aggarwal, *Machine Learning for Text: An Introduction*, 2018.

[28] S. Marsland, *Machine Learning: An Algorithmic Perspective*, 2nd ed. Chapman and Hall/CRC, 2014.

[29] A. Géron, *Hands-On Machine Learning with Scikit-Learn & TensorFlow*, 2017.

[30] T. Mitchell, *Machine Learning*, ser. McGraw-Hill International Editions. McGraw-Hill, 1997. [Online]. Available: https://books.google.com.mx/books?id=EoYBngEACAAJ

[31] "Univariate Linear Regression - MuPAD," (Accessed on 2020-02-18). [Online]. Available: https://www.mathworks.com/help/symbolic/mupad_ug/univariate-linear-regression.html

[32] "LOGISTIC REGRESSION CLASSIFIER - Towards Data Science," (Accessed on 2019-12-02). [Online]. Available: https://towardsdatascience.com/logistic-regression-classifier-8583e0c3cf9

[33] "Decision Trees - Introduction to Machine Learning," (Accessed on 2020-02-13). [Online]. Available: https://tomaszgolan.github.io/introduction_to_machine_learning/markdown/introduction_to_machine_learning_02_dt/introduction_to_machine_learning_02_dt/

[34] "Boosting Algorithms Explained - Towards Data Science," (Accessed on 2020-02-14). [Online]. Available: https://towardsdatascience.com/boosting-algorithms-explained-d38f56ef3f30

[35] "Grid Search for model tuning - Towards Data Science," (Accessed on 2019-12-02). [Online]. Available: https://towardsdatascience.com/grid-search-for-model-tuning-3319b259367e

[36] "sklearn.feature_selection.chi2 — scikit-learn 0.21.3 documentation," (Accessed on 2019-12-02). [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.chi2.html

[37] "Sequential Feature Selector - mlxtend," (Accessed on 2019-12-02). [Online]. Available: http://rasbt.github.io/mlxtend/user_guide/feature_selection/SequentialFeatureSelector/

[38] M. Álvarez-Carmona, E. Guzmán-Falcón, M. Montes-y Gómez, H. J. Escalante, L. Villaseñor-Pineda, V. Reyes-Meza, and A. Rico-Sulayes, "Overview of MEX-A3T at IberEval 2018: Authorship and aggressiveness analysis in Mexican Spanish tweets," *CEUR Workshop Proceedings*, vol. 2150, pp. 74–96, 2018.

[39] M. Graff, S. Miranda-jim, E. S. Tellez, and N. S. Claudia, "INGEOTEC at MEX-A3T : Author profiling and aggressiveness analysis in Twitter using µ TC and EvoMSA."

[40] N. Bauwelinck, G. Jacobs, V. Hoste, and E. Lefever, “LT3 at SemEval-2019 Task 5: Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter (hatEval),” pp. 436–440, 2019.

[41] D. Cer, Y. Yang, S. yi Kong, N. Hua, N. Limtiaco, R. St. John, N. Constant, M. Guajardo-Céspedes, S. Yuan, C. Tar, Y. H. Sung, B. Strope, and R. Kurzweil, “Universal sentence encoder for English,” *EMNLP 2018 - Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Proceedings*, pp. 169–174, 2018.

[42] “Developer Policy – Twitter Developers,” (Accessed on 2020-02-03). [Online]. Available: https://developer.twitter.com/en/developer-terms/policy

[43] B. Wang, Y. Ding, S. Liu, and X. Zhou, “YNU wb at HASOC 2019: Ordered neurons LSTM with attention for identifying hate speech and offensive language,” *CEUR Workshop Proceedings*, vol. 2517, pp. 191–198, 2019.

[44] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contextualized word representations,” *CoRR*, vol. abs/1802.05365, 2018. [Online]. Available: http://arxiv.org/abs/1802.05365

[45] “Metadata | Definition of Metadata by Merriam-Webster,” (Accessed on 2020-01-30). [Online]. Available: https://www.merriam-webster.com/dictionary/metadata

[46] “Tweet object — Twitter Developers,” (Accessed on 2019-12-02). [Online]. Available: https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/tweet-object

[47] “User object — Twitter Developers,” (Accessed on 2019-12-02). [Online]. Available: https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/user-object

[48] “Tweets – Twitter Developers,” (Accessed on 2019-10-02). [Online]. Available: https://developer.twitter.com/en/products/tweets

[49] “GetOldTweets3 · PyPI,” (Accessed on 2019-10-02). [Online]. Available: https://pypi.org/project/GetOldTweets3/

[50] "Twython — Twython 3.6.0 documentation," (Accessed on 2019-10-02). [Online]. Available: https://twython.readthedocs.io/en/latest/

[51] D. Olson and D. Delen, *Advanced Data Mining Techniques*, 01 2008.

[52] "sklearn.preprocessing.KBinsDiscretizer — scikit-learn 0.21.3 documentation," (Accessed on 2019-12-02). [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.KBinsDiscretizer.html

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

Glossary

Table 5.1: Acronyms / Abbreviations

API	Application Programming Interface
B/L	Baseline
BoW	Bag-of-Words
CV	Cross Validation
DE	Document Embeddings
DS	Dataset
DT	Decision Trees
Eng	English
FN	False negative
FP	False positive
GS	Grid Search
GSCL	German Society for Computational Linguistics
HASOC	Hate Speech and Offensive Content Identification in Indo-European Languages
HOF	Hate-Offensive
HS	Hate Speech
LSF	Lexical Syntactic Feature
LSVM	Linear Support Vector Machine
LR	Logistic Regression
MD	Metadata

Table 5.2: Continuation of Table 5.1

NB	Naïve Bayes
NLP	Natural Language Processing
OF	Offensive
RBF	Radial basis function
RF	Random Forest
SBFS	Sequential Backward Floating Selection
SBS	Sequential Backward Selection
SFFS	Sequential Forward Floating Selection
SFS	Sequential Forward Selection
Span	Spanish
STDEV	Standard Deviation
SVM	Support Vector Machine
tf-idf	term frequency–inverse document frequency
TN	True negative
TP	True positive
UK	United Kingdom
USA	United States of America
U.S.	United States
WE	Word Embeddings

Appendix A

A.1 UACh at MEX-A3T 2019: Preliminary Results on Detecting Aggressive Tweets by Adding Author Information Via an Unsupervised Strategy.

Research paper attached at the end.

A.2 UACh-INAOE at HASOC 2019: Detecting Aggressive Tweets by Incorporating Authors' Traits as Descriptors

Research paper attached at the end.

A.3 Configuration of models

Configuration of Stacking	Best set of parameters	Extra features added
LR to RF model: RF using only predictions of Logistic Regression on text.	(n_estimators=10, max_features = 'auto', max_depth = 4, criterion = 'entropy')	None
MD model: RF using predictions of LR on text and raw metadata.	(n_estimators=100, max_features = None, max_depth = 4, criterion = 'gini')	All available metadata, feature selection was made by Sequential Backwards Floating Selection (SBFS)
MD + Bins model: RF using predictions of LR on text, raw metadata and LR predictions on discretized metadata.	Same as MD model #3 for RF. Binning: (Bins = 5, strategy = 'quantile')	

Figure A.1: Configuration of models in Davidson DS using LR as B/L.

Configuration of Stacking	Best set of parameters	Extra features added
LSVM to RF model: RF using only predictions of LSVM on text.	(n_estimators=10, max_features = None, max_depth = 3, criterion = 'entropy')	None
MD model: RF using predictions of LSVM on text and raw metadata.	(n_estimators=100, max_features = None, max_depth = 4, criterion = 'gini')	Default Image Friends count Quote status Reply status Username

Figure A.2: Configuration of models in Davidson DS using LSVM as B/L.

Configuration of Stacking	**Best set of parameters**	**Extra features added**
LR to RF model: RF using only predictions of Logistic Regression on text.	(n_estimators=10, max_features = ‘auto’, max_depth = 3, criterion = ‘gini’)	None
MD model: RF using predictions of LR on text and raw metadata.	(n_estimators=200, max_features = None, max_depth = 6, criterion = ‘gini’)	Default Image, Followers count, Friends count, Quote status, Reply status, Tweet Favorite count, User Favorite count, User Statuses count, Username
MD + Bins model: RF using predictions of LR on text, raw metadata and LR predictions on discretized metadata.	Same as MD model for RF. Binning: (Bins = 3, strategy = ‘kmeans’)	

Figure A.3: Configuration of models in hatEval English DS using LR as B/L.

Configuration of Stacking	Best set of parameters	Extra features added
LSVM to RF model: RF using only predictions of LSVM on text.	(n_estimators=100, max_features = None, max_depth = 6, criterion = ‘gini’)	None
MD model: RF using predictions of LSVM on text and raw metadata.	(n_estimators=100, max_features = ‘auto’, max_depth = 4, criterion = ‘entropy’)	Default Image Friends count Quote status Reply status Tweet Favorite count User Favorite count User Statuses count Username Feature selection was made by SBFS.

Figure A.4: Configuration of models in hatEval English DS using LSVM as B/L.

Configuration of Stacking	**Best set of parameters**	**Extra features added**
LR to RF model: RF using only predictions of Logistic Regression on text.	(n_estimators=10, max_features = 'auto', max_depth = 4, criterion = 'entropy')	None
MD model: RF using predictions of LR on text and raw metadata.	(n_estimators=10, max_features = 'auto', max_depth = 6, criterion = 'entropy')	Default Image Default Profile Followers count Listed count Quote status Reply status Username
MD + Bins model: RF using predictions of LR on text, raw metadata and LSVM predictions on discretized metadata.	Same as MD model for RF. Binning: (Bins = 2, strategy = 'uniform')	

Figure A.5: Configuration of models in hatEval Spanish DS using LR as B/L.

Configuration of Stacking	**Best set of parameters**	**Extra features added**
LSVM to RF model: RF using only predictions of LSVM on text.	(n_estimators=150, max_features = 'auto', max_depth = 3, criterion = 'gini')	None
MD model: RF using predictions of LSVM on text and raw metadata.	(n_estimators=150, max_features = 'auto', max_depth = 6, criterion = 'gini')	All available metadata
MD + Bins model: RF using predictions of LR on text, raw metadata and LR predictions on discretized metadata.	Same as MD model for RF. Binning: (Bins = 3, strategy = 'kmeans')	

Figure A.6: Configuration of models in hatEval Spanish DS using LSVM as B/L.

Configuration of Stacking	Best set of parameters	Extra features added
LR to RF model: RF using only predictions of LR on text.	(n_estimators=200, max_features = 'auto', max_depth = 4, criterion = 'gini')	None
MD model: RF using predictions of LR on text and raw metadata.	(n_estimators=200, max_features = 'auto', max_depth = 3, criterion = 'gini')	All available metadata, feature selection was made by SBFS

Figure A.7: Configuration of models in HASOC DS using LR as B/L.

Configuration of Stacking	Best set of parameters	Extra features added
LSVM to RF model: RF using only predictions of LSVM on text.	(n_estimators=150, max_features = 'auto', max_depth = 6, criterion = 'entropy')	None
MD model: RF using predictions of LSVM on text and raw metadata.	(n_estimators=10, max_features = 'auto', max_depth = 3, criterion = 'entropy')	All available metadata, feature selection was made by SBFS

Figure A.8: Configuration of models in HASOC DS using LSVM as B/L.

Configuration of Stacking	Best set of parameters	Extra features added
LR to RF model: RF using only predictions of LR on text.	(n_estimators=200, max_features = 'auto', max_depth = 5, criterion = 'gini')	None
MD model: RF using predictions of LR on text and raw metadata.	(n_estimators=200, max_features = 'auto', max_depth = 5, criterion = 'gini')	All available metadata except Username, Listed count and Friends count

Figure A.9: Configuration of models in MEX-A3T DS using LR as B/L.

Configuration of Stacking	Best set of parameters	Extra features added
LSVM to RF model: RF using only predictions of LSVM on text.	(n_estimators=150, max_features = 'auto', max_depth = 4, criterion = 'gini')	None
MD model: RF using predictions of LSVM on text and raw metadata.	(n_estimators=150, max_features = 'auto', max_depth = 4, criterion = 'gini')	All available metadata, feature selection was made by SBFS

Figure A.10: Configuration of models in MEX-A3T DS using LSVM as B/L.

UACh at MEX-A3T 2019: Preliminary Results on Detecting Aggressive Tweets by Adding Author Information Via an Unsupervised Strategy

Marco Casavantes, Roberto López, and Luis Carlos González

Universidad Autónoma de Chihuahua. Facultad de Ingeniería. Chihuahua, Chih., México
{p271673,jrlopez,lcgonzalez}@uach.mx

Abstract. In this paper we describe our participation for the Aggressiveness Detection Track in the second edition of MEX-A3T. We evaluate different strategies for text classification, including classifiers such as Support Vector Machines and a Multilayer Perceptron trained on n-grams (words and characters) and word embeddings. We also study the inclusion of features to try to give context to the text messages and explore if people verbally attack differently depending on their traits and overall environment. Preliminary results show that our strategy is competitive to detect aggression in tweets, ranking in 2nd place with respect to the participants of 2018 and 2019.

Keywords: Spanish text classification · Aggressiveness Detection · Multilayer Perceptron.

1 Introduction

Technology has changed the way in which people communicate with each other, giving rise to new services such as social networks, where a style of informal communication is used. Such social networks, though, present several challenges to maintain communication channels open to the free sharing of ideas. The intolerance and aggressiveness of certain users affects the experience of other consumers or people interested in being part of the communities and their conversations. The fact of not being face to face in the communication channel and even preserve anonymity, encourages these individuals to express themselves offensively. However, the volume of messages that are sent daily, the growth of online communities, and the respective ease of access to these social networks, make the moderation of communication channels a difficult task to be dealt with by conventional means, and as people increasingly communicate online, the need for high quality automated abusive language classifiers becomes much more profound[1].

One of the goals of the second edition of MEX-A3T[2] is to tackle this problem and further improve the research of this important NLP task, the detection of aggressive tweets in Mexican Spanish. In this work we evaluate strategies proposed before, such as the use of lexical features through TF-IDF representations, and different approaches to add features in order to try to give context to each text. Surprisingly, even tackling the task with such a basic approach our proposal is able to offer competitive results, just slightly behind the top performer of this competition in 2018 and 2019, INGEOTEC. Furthermore, we also investigate how to incorporate author's traits by using unsupervised methods and attempting to include this information as possible features, based on the hypothesis that there are different ways of aggression depending on the author's context.

2 Proposed Method

2.1 Data Pre-processing

After loading the train and test sets, we strip the tweets from non-alphanumeric characters and only keep some relevant Spanish characters (á,é,í,ó,ú,ñ,and ü), all words are then made lowercase and subsequently we noticed that in both sets there exists many different terms to express laughter (mainly due to how many times "ja" is repeated when the word "jaja" appears and because of typos) so that led us to replace every word containing "jaja" to "risa" (laugh), with the purpose of decreasing the number of terms that represent this emotion.
It is worth mentioning that we also created and conducted experiments on a version of the datasets where emojis were converted to text and hashtags were separated by words (e.g., ":)" would turn into "smiling face", and "#FelizMiércoles" would be "feliz miércoles"), however most hashtags were wrongly separated and the performance of the classifiers decreased by incorporating these steps and were therefore discarded.

2.2 Features

We conducted our research using the following features:
Lexical: We use word n-grams (n=1, 2) and char n-grams (n=3, 4) as features, this collection of terms is weighted with its term frequency-inverse document frequency (TF-IDF).
Document Embeddings: The objective was to represent the tweets through Word Embeddings[3] and try different classifiers with these new features, each text message was converted to a vector of size = 300 (mean of the vectors of each word). The model of words in Spanish was computed with fastText[4] and downloaded from [5] .
User Occupation and Location predictions: Although we attempted several strategies to obtain unsupervised author profiles for each document [6], we

ended up using the output of the system developed by [7] as predictions of occupation and location values to explore the possibility of differences in vocabulary that exists according to the profile of the author of the message.
Grouping tweets by theme: An implementation of Self Organizing Maps (SOM) as a clustering strategy called MiniSom[8] was used with aims to find groups in the collection of texts based on underlying or non-explicit features, the clustering was done including all words and also ignoring swear words (to reduce the noise and focus on thematic terms), after training the network we were able to compute the coordinates assigned to a tweet on the map and use these as new features.
Perspicuity score / Inflesz scale: Based on [9], we adapted the idea of capturing the quality of each tweet by using a modified Flesch Reading Ease score (since this test only applies to text written in English), called Perspicuity score and its equivalence to the Inflesz scale, following the equation described in [10] where the number of sentences is also fixed at one.

All the extra categorical features mentioned above were concatenated following a One Hot Encoding scheme.

3 Experiments and Results

The datasets were provided by MEX-A3T Team. Table 1 shows the distribution of training and test partitions for Spanish tweets.

Table 1. Data distribution for Spanish tweets corpus

Class	Training	Test
Aggressive	2727	N/A
Non-aggressive	4973	N/A
Total	7700	3156

We separated the training set in 67% for training and 33% for validation to evaluate our experiments with different combinations of features discussed in section 2.2. We started our research by recreating the baselines described in the overview of the first edition of MEX-A3T[11], particularly focusing on the character trigrams baseline, as it holds the best performance in comparison to the BoW baseline.
We trained Linear Support Vector Machines and a Multilayer Perceptron as classifiers for this task, and we decided to use the perceptron as the final system to submit our predictions since it exhibited the best results in the validation stage, as shown in Table 2 where we obtained the F1-score macro and the F-Measure over the aggressive class. We performed all modeling regarding the creation of tf-idf feature matrices and SVM classifiers using scikit-learn[12], and for the Multilayer Perceptron, we used the implementation described in [13],

there was only an instance were this Perceptron couldn't be trained with Word Embeddings, so we tried another configuration on the MLPClassifier from scikit-learn getting low scores similar to the ones obtained using LinearSVM, and therefore casting aside this approach.

3.1 Results

As stated before, the Multilayer Perceptron was chosen as final system, however, because of time and memory constraints we had to train this model using only character n-grams of range [3,4] for this task even though later results shows better performance by using n-grams of range [3,5]. Table 3 list the top five final rankings for the aggressiveness detection task for 2019, more details of all results of the contest are shown at [2]. It is interesting to observe that even when our system relied on such a basic approach, it is able to compete face-to-face against INGEOTEC, a model based on an ensemble of classifiers, which specially tailors discriminative features for aggressive detection via a Genetic Programming strategy.

3.2 Analysis

To breakdown our results, we started by getting the 10 most valuable n-grams at character level separated by length, as shown in Table 4. With respect to the aggressive class, our final configuration had more false positives than false negatives, meaning that it was easier for an aggressive tweet to be missclassified as non-aggressive than the other way around. Despite running several experiments and adding new features trying to give context to the tweets, in hopes of improving classification in this task, unfortunately these strategies showed, at best, almost unnoticeable changes in the results, and hinder of classification at worst. After manual inspection, we observed that this could have happened because:

- Occupation and Location predictions did not group the messages in a balanced way, in fact, most tweets would fall under only one out of eight available categories for occupation and six categories for location.
- SOM Coordinates would not enhance the classification scores before as the clusters were capturing word repetition instead of thematic aspects for each tweet. Later experiments (after submission of results) showed that this behaviour was caused because the clustering was made with n-grams; training the SOM with word embeddings created with the train set of this task (without external resources) solved this issue and did a better job at grouping the tweets by subjects.
- There was no relevant pattern by applying a perspicuity score to each tweet, as there were multiple cases of similar scores assigned to both aggressive and non-aggressive messages.

Table 2. Detailed classification with F1-scores in the validation stage.

Added features	**Classifier**	**Char n-gram range**	**F1-score macro**	**F1-score** (aggressive_class)
None	LinearSVM	[3,3]	0.76	0.68
None	MLP	[3,3]	0.77	0.66
None	LinearSVM	[3,4]	0.77	0.69
None	MLP	[3,4]	0.79	0.69
None	LinearSVM	[3,5]	0.77	0.70
None	MLP	[3,5]	0.79	0.70
Word Embeddings	LinearSVM	N/A	0.59	0.39
Word Embeddings	MLPClassifier	N/A	0.56	0.34
Occupation (O)	LinearSVM	[3,3]	0.76	0.69
Occupation (O)	MLP	[3,3]	0.78	0.67
Occupation (O)	LinearSVM	[3,4]	0.77	0.69
Occupation (O)	MLP	[3,4]	0.76	0.68
Location (L)	LinearSVM	[3,3]	0.77	0.68
Location (L)	MLP	[3,3]	0.77	0.65
Location (L)	LinearSVM	[3,4]	0.77	0.70
Location (L)	MLP	[3,4]	0.76	0.67
Perspicuity (P)	LinearSVM	[3,3]	0.76	0.68
Perspicuity (P)	MLP	[3,3]	0.77	0.66
Perspicuity (P)	LinearSVM	[3,4]	0.77	0.70
Perspicuity (P)	MLP	[3,4]	0.76	0.67
SOM Coordinates (S)	LinearSVM	[3,3]	0.76	0.68
SOM Coordinates (S)	MLP	[3,3]	0.78	0.66
SOM Coordinates (S)	LinearSVM	[3,4]	0.76	0.69
SOM Coordinates (S)	MLP	[3,4]	0.79	0.69
O + L + P + S	LinearSVM	[3,3]	0.76	0.69
O + L + P + S	MLP	[3,3]	0.78	0.67
O + L + P + S	LinearSVM	[3,4]	0.77	0.69
O + L + P + S	MLP	[3,4]	0.77	0.69

Table 3. Final scores of the aggressiveness detection task.

Rank	**Team**	**F1-score** (aggressive_class)	**F1-score** (non-aggressive_class)	**Accuracy**
1	INGEOTEC	0.4796	0.8131	0.7250
2	Casavantes (Our approach)	0.4790	0.8164	0.7285
3	GLP (run 2)	0.4749	0.7949	0.7050
4	GLP (run 4)	0.4635	0.7774	0.6854
5	mineriaUNAM (run 2)	0.4549	0.8016	0.7075

Table 4. Best n-grams at character level in training set

Length	n-gram	Frequency in aggressive class	Frequency in non aggressive class
3 chars	'os '	3074	3207
	' de'	2571	3879
	'as '	2205	3252
	'que'	1991	3540
	' qu'	1965	3667
4 chars	' de '	1860	2798
	'que '	1768	3262
	' que'	1649	2954
	' put'	1589	1517
	' la '	1062	2195

4 Conclusions and Future Work

In this paper, we describe our strategy to classify aggressive and non-aggressive tweets in Mexican Spanish. In our best performing system, we use only lexical features and our results show a better performance than most results of all participants. This outcome, and the fact that the F-measure for the aggressive class is still low compared to the score on the non-aggressive class, motivates the idea of future work focusing on feature analysis for aggressiveness detection and explore which representations are truly relevant, including word embeddings, Bag of Words and Characters of different n-gram ranges, see if these complement each other and if so, how to combine them. We analyzed our clustering strategies, and after changing the way they were trained we could observe slight improvement in classification results, motivating us to keep experimenting on ways to try to add context to the text messages. We also believe in the potential that neural networks display for this task, and that more research on how to build and train them properly will certainly improve the current situation of this task.
As future work, we look forward to develop new strategies based on deep neural networks, such as Recurrent Neural Networks, which are tools aimed to work with sequential data similar in nature to time series.

References

1. Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, pages 145–153, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee.
2. Mario Ezra Aragón, Miguel Á Álvarez-Carmona, Manuel Montes-y Gómez, Hugo Jair Escalante, Luis Villaseñor-Pineda, and Daniela Moctezuma. Overview of MEX-A3T at IberLEF 2019: Authorship and aggressiveness analysis in Mexican Spanish tweets. In *Notebook Papers of 1st SEPLN Workshop on Iberian Languages Evaluation Forum (IberLEF), Bilbao, Spain, September*, 2019.

3. Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1188–II–1196. JMLR.org, 2014.
4. Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.
5. Github - mquezada/starsconf2018-word-embeddings: Material para el taller "representaciones vectoriales de palabras basadas en redes neuronales" de la starsconf 2018. https://github.com/mquezada/starsconf2018-word-embeddings. (Accessed on 06/02/2019).
6. Roberto López Santillán, L.C. González-Gurrola, and Graciela Ramírez-Alonso. Custom document embeddings via the centroids method: Gender classification in an author profiling task. In Linda Cappellato, Nicola Ferro, Jian-Yun Nie, and Laure Soulier, editors, *CLEF 2018 Evaluation Labs and Workshop – Working Notes Papers, 10-14 September, Avignon, France*. CEUR-WS.org, September 2018.
7. Rosa Marıa Ortega-Mendoza and A Pastor López-Monroy. The winning approach for author profiling of mexican users in twitter at mex. a3t@ ibereval-2018.
8. Github - justglowing/minisom: Minisom is a minimalistic implementation of the self organizing maps. https://github.com/JustGlowing/minisom. (Accessed on 06/03/2019).
9. Thomas Davidson, Dana Warmsley, Michael W. Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. *CoRR*, abs/1703.04009, 2017.
10. Escala inflesz — legible. https://legible.es/blog/escala-inflesz/. (Accessed on 06/02/2019).
11. Miguel Álvarez-Carmona, Estefanía Guzmán-Falcón, Manuel Montes-y Gómez, Hugo Jair Escalante, Luis Villaseñor-Pineda, Verónica Reyes-Meza, and Antonio Rico-Sulayes. Overview of MEX-A3T at IberEval 2018: Authorship and aggressiveness analysis in Mexican Spanish tweets. *CEUR Workshop Proceedings*, 2150:74–96, 2018.
12. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
13. Github - afshinrahimi/sparsemultilayerperceptron: Lasagne / theano based multilayer perceptron mlp which accepts both sparse and dense matrices and is very easy to use with scikit-learn api similarity. https://github.com/afshinrahimi/sparsemultilayerperceptron. (Accessed on 06/03/2019).

UACh-INAOE at HASOC 2019: Detecting Aggressive Tweets by Incorporating Authors' Traits as Descriptors

Marco Casavantes[1], Roberto López[1]
, Luis Carlos González-Gurrola[1], and Manuel Montes-y-Gómez[2]

[1] Facultad de Ingeniería, Universidad Autónoma de Chihuahua (UACh), Mexico
{p271673,jrlopez,lcgonzalez}@uach.mx
[2] Laboratorio de Tecnologías del Lenguaje, Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), Mexico
mmontesg@inaoep.mx

Abstract. In this paper, we describe our participation for the Aggressiveness Detection Track in English texts for HASOC 2019. We evaluate different strategies for text classification, including classifiers such as Logistic Regression and Support Vector Machines trained on n-grams (words and characters) and word embeddings for clustering techniques. We also study the incorporation of contextual characteristics to explore whether people verbally attack differently depending on their traits and environment.

Keywords: English text classification · Aggressiveness Detection · Twitter.

1 Introduction

As people increasingly communicate online through social media, they may deal with negative experiences such as being targets of cyberbullying or expose themselves to hateful and vulgar content. These problems have become more relevant in the past few years, as they pose several challenges to preserve the freedom of speech and sharing of ideas over these communication channels. The growth in the volume of the messages that are posted on social media on a daily basis demands more efficient means to detect and moderate the spread of offensive content and hate speech. Furthermore, administrators of social media platforms could prevent abusive behavior and harmful experiences. It is crucial to address the importance of early identification of users that promote hate speech, as this could enable important outreach programs, to prevent an escalation from speech to action [11]. Moreover, considering the high levels of aggressiveness and hostile behaviour of certain users towards particular groups or individuals, more serious real-life issues, like self-harm or suicide, could actually be prevented.

In the last years, several shared tasks have been organized with the purpose of attracting attention to these problems [14, 12, 7, 13]. Take for instance the second edition of MEX-A3T [4]. In that event, our participation focused on detecting aggressive tweets in a Mexican Spanish dataset, by incorporating traits of authors (e.g., occupation, location). Therefore, by participating in HASOC [9] (Sub-task A for English), we aimed to test our approach on a different collection of tweets, tweaking our system to face this new challenge.

In this study, we evaluate common strategies such as lexical feature engineering through *term frequency* representations (e.g., bag of words through *tfidf*), along with different approaches with the aim to enhance features by adding context to each document. Furthermore, we also advanced our research by including the authors' traits, and using the outcome of unsupervised methods as potential useful features.

The hypothesis behind our approach is that offensive messages could be better recognized by analyzing not only the message but the user profile. The rest of this document is organized as follows: in section 2 we describe our approach; in section 3, the results attained are detailed and analyzed; finally, in section 4 we state our conclusions and delineate some future work.

2 Proposed Method

Similar to our participation in MEX-A3T 2019 [5], we aim to enrich the classification of aggressive tweets by including a possible theme to which each tweet belongs, being this the main experiment that attempt to support our hypothesis. This section gives a complete description of the changes and adaptation of features that we propose in our approach.

2.1 Data Pre-processing

Once the text files were loaded using UTF-8 encoding, we conducted our experiments in a custom version of the dataset where:

- All words are made lowercase.
- Emojis are converted into their text representation.
  (e.g., ":face_with_tears_of_joy:")
- Tweets are stripped from non-alphanumeric characters excluding some relevant symbols (#, @ and _).
- Every URL (occurrence of the sequence "http") was replaced with "weblink" to evenly represent references to external sources.

2.2 Features

We conducted our research using the following features:
Lexical: We use both word n-grams (n=1, 2) and char n-grams (n=2, 3, 4),

however this collection of terms was only weighted with its term frequency.
Document Embeddings: Using only the text available in both the train and test set, we employed a representation of the tweets through Word Embeddings [8] to feed different clustering strategies.
Grouping tweets by theme: We use different clustering methods (an implementation of Self Organizing Maps [1], K-Means and Affinity Propagation) to generate new features based on thematic terms in each tweet.

- The SOM allowed us to locate each tweet on a two-dimensional plane, taking the coordinates as new features.
- Using K-Means and Affinity Propagation we calculate, for every sample, the distance between itself and the rest of clusters.

Flesch Reading Ease and Flesch Kincaid Grade scores: Based on [6], we wanted to capture the quality of each tweet by getting the Flesch Reading Ease and Kincaid Grade scores using textstat [3]. In our experiments the number of sentences is also fixed at one.
Named Entity Recognition (NER) counters: Upon manual inspection of frequent tokens (Table 4), we observed that a big part of the dataset included references to people like Donald Trump (current president of USA), Boris Johnson (current Prime Minister of the United Kingdom), Mahendra Singh Dhoni (indian international cricketer) and organizations like ICC (International Cricket Council). Based on this information we decided to incorporate counters of how many persons, organizations and locations were mentioned in each text using polyglot [2].

3 Experiments and Results

The datasets were provided by the HASOC-2019 organization team. Table 1 shows the distribution of training and test partitions for English tweets.

Table 1. Data distribution for English tweets corpus used in HASOC-2019.

Class	Training	Test
Non Hate-Offensive (NOT)	3591	N/A
Hate and Offensive (HOF)	2261	N/A
Total	5852	1153

We started our research by recreating our baselines used in MEX-A3T 2019, this time focusing on the word unigrams and bigrams baseline, as it holds the best performance in this task in comparison to the character n-grams baseline. In order to generalize our results for the test set, we evaluated our experiments using two different configurations, a single stratified train-validation split and a 5-Fold Cross Validation.

We trained Linear Support Vector Machines and a Logistic Regression classifier for this task, and we decided to use both of them to submit our predictions:

- **Run 1** consists of a LinearSVM trained with the best 800 features from a Bag of Words of range=(1,2) considering the term frequency of all the tokens. The feature selection was done by a chi-squared statistics test on a 70-30% train-validation split.
- **Run 2** is the same as Run 1, but in this case the top 1250 features were selected from a stratified 5-fold cross validation on the train set, specifying a 20% split for the validation set.
- **Run 3** is the result of creating an ensemble of two Logistic Regression classifiers, one trained with a Bag of Words and the other one with a Bag of Character n-grams. The predictions were assigned by choosing the model with the highest probability for each tweet.

Table 2 shows the macro and weighted F1-score that we obtained over the two classes. We performed all modeling regarding the creation of term frequency feature matrices, classifiers, cross validation and Kmeans/Affinity Propagation clustering using scikit-learn[10].

3.1 Results for HASOC 2019

As stated before, a Linear Support Vector Machine was chosen as our system's classifier adding Named Entity Recognition counters for runs 1 and 2, and a Logistic Regression classifier ensemble was used to submit run 3. Table 3 lists the results of our three submissions for the English Hate Speech and Offensive Content Identification Sub-task A for HASOC 2019, more information of all results of the contest is available at [9].

3.2 Analysis

We analyzed our participation in HASOC'19 in two ways. The first analysis focuses on observing what are the 10 most frequent n-grams (excluding stopwords) at word level (separated by length) in the Hate-Offensive class, these are shown in Table 4. We also exhibit in Table 5 the best word n-grams per class according to the Logistic Regression classifier (LRC) trained with the whole training set. In our final configuration, it was easier for an offensive tweet to be missclassified as non-aggressive, and despite running several experiments, most of our attempts to improve classification in this task by adding new features trying to give context to the tweets unfortunately affected the results negatively. After inspection, we observed that this could have happened because:

- The clustering techniques that we used didn't add anything new since the tweets were kind of grouped from the beginning, as some main topics can be spotted (e.g., Trump, Dhoni/ICC and "DoctorsFightBack" protest related tweets).

Table 2. Detailed classification with F1-scores in the validation stage.

Run	**Features**	**Setup**	**Macro F1-score**	**Weighted F1-score**
	Complete BoW	LinearSVM on single split	0.6315	0.6571
	Top 800 features from BoW	“	0.6452	0.6717
Run 1	NER + top features	“	0.6468	0.6731
	Complete BoW	LinearSVM on 5-FoldCV	0.6242	0.6527
	Top 1,250 features from BoW	“	0.6323	0.6645
Run 2	NER + top features	“	0.6352	0.6671
	Flesch Reading Ease Score	“	0.6181	0.6468
	Flesch Kincaid Grade Score	“	0.6205	0.6489
	NER counters	“	0.6250	0.6518
	K-Means Clustering	“	0.6237	0.6521
	Affinity Propagation	“	0.6213	0.6501
	SOM Coordinates	“	0.6185	0.6475
Run 3	Bag of Words and Chars.	Ensemble on single split	0.6174	0.6515
Run 3	Bag of Words and Chars.	Ensemble on 5-FoldCV	0.6227	0.6547

Table 3. Final scores of the 2019 Hate Speech and Offensive Content Identification Sub-task A in English.

Rank	**Team**	**Macro F1-score**	**Weighted F1-score**
1/79	YNU_wb	0.7882	0.8395
20/79	UACh-INAOE_english_1_run_3	0.7075	0.7828
29/79	UACh-INAOE_english_1_run_2	0.6765	0.7490
30/79	UACh-INAOE_english_1_run_1	0.6753	0.7491

– Since there were multiple cases of similar quality scores assigned to both not offensive and offensive messages, the classifiers could not pick a relevant pattern.

Table 4. Most frequent n-grams at word level in training set

Length	**N-gram**	**Freq. in HOF class**	**Freq. in NOT class**
Unigram	'fucktrump'	515	628
	'trumpisatraitor'	386	484
	'realdonaldtrump'	383	347
	'trump'	280	323
	'icc'	270	543
Bigram	'fucktrump weblink'	130	236
	'world cup'	65	127
	'trumpisatraitor weblink'	58	80
	'borisjohnsonshouldnotbepm weblink'	50	73
	'resisttrump fucktrump'	36	129

Table 5. Best word n-grams per class in training set

Class	**N-gram**	**LRC Weight**	**-**	**Class**	**N-gram**	**LRC Weight**
HOF	'fuck'	-4.51		NOT	'dhonikeepstheglove'	3.67
	'fucking'	-2.90			'doctorsfightback'	3.16
	'dickhead'	-1.88			'dhoni'	1.62
	'youre'	-1.85			'shameonicc'	1.50
	'gandinaaliabuse'	-1.83			'doctors'	1.32
	'traitor'	-1.78			'borisjohnsonshouldnotbepm'	1.13
	'shit'	-1.60			'new'	1.08
	'you'	-1.55			'happy'	1.06
	'president'	-1.52			'happy johnmccainday'	1.05
	'hes'	-1.51			'real'	0.98

The second analysis addresses the performance of our proposal, regarding F1-score and contrasted against the rest of the competitors. Fig. 1 presents two box plots for the complete distribution of competitors in terms of Macro F1 and Weighted F1. This analysis suggests that the outcome achieved by our proposal is competitive, practically been located within the first quartile for all participants.

4 Conclusions and Future Work

In this paper, we describe our strategy to classify offensive and non-offensive tweets in a relatively new English collection of tweets. Regarding our experi-

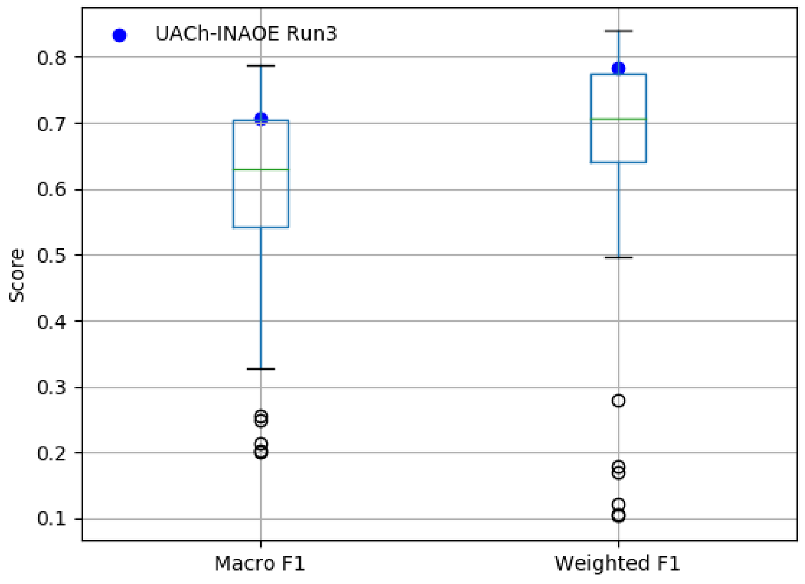


Fig. 1. Box plots of the results for Sub-task A for English.

ments for this task we can conclude that, in our best performing system, term frequency matrices of words and character n-grams complement each other in an ensemble of Logistic Regression classifiers. After seeing that the NER counters were basically the only useful features in the validation stage and the fact that we could not improve our classification scores with our current approach on providing context to tweets motivates the idea of future work focusing on finding new features to help us in our goal to see if it's possible to differentiate an offensive text from a non offensive one based on the message's underlying properties and the author's attributes.

References

[1] Github - justglowing/minisom: Minisom is a minimalistic implementation of the self organizing maps. https://github.com/JustGlowing/minisom. (Accessed on 06/03/2019).

[2] polyglot· PyPI. https://pypi.org/project/polyglot/. (Accessed on 09/09/2019).

[3] textstat· PyPI. https://pypi.org/project/textstat/. (Accessed on 09/09/2019).

[4] Aragón, M. E., Álvarez-Carmona, M. Á., Montes-y Gómez, M., Escalante, H. J., Villaseñor-Pineda, L., and Moctezuma, D. (2019). Overview of MEX-A3T at IberLEF 2019: Authorship and aggressiveness analysis in Mexican Spanish tweets. In *Notebook Papers of 1st SEPLN Workshop on Iberian Languages Evaluation Forum (IberLEF), Bilbao, Spain, September*.

[5] Casavantes, M., López, R., and González, L. C. (2019). UACh at MEX-A3T 2019 : Preliminary Results on Detecting Aggressive Tweets by Adding Author Information Via an Unsupervised Strategy.

[6] Davidson, T., Warmsley, D., Macy, M. W., and Weber, I. (2017). Automated hate speech detection and the problem of offensive language. *CoRR*, abs/1703.04009.

[7] Kumar, R., Ojha, A. K., Malmasi, S., and Zampieri, M. (2018). Benchmarking aggression identification in social media. In *Proceedings of TRAC*.

[8] Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1188–II–1196. JMLR.org.

[9] Mandl, T., Modha, S., Patel, D., Dave, M., Mandlia, C., and Patel, A. (2019). Overview of the HASOC track at FIRE 2019: Hate Speech and Offensive Content Identification in Indo-European Languages. In *Proceedings of the 11th annual meeting of the Forum for Information Retrieval Evaluation*.

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

[11] Waseem, Z. and Hovy, D. (2016). Hateful symbols or hateful people? predictive features for hate speech detection on twitter. pages 88–93.

[12] Wiegand, M., Siegel, M., and Ruppenhofer, J. (2018). Overview of the germeval 2018 shared task on the identification of offensive language.

[13] Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N., and Kumar, R. (2019a). Predicting the Type and Target of Offensive Posts in Social Media. In *Proceedings of NAACL*.

[14] Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N., and Kumar, R. (2019b). Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86.