Cyberbullying Detection Task: The EBSI-LIA-UNAM system (ELU) at COLING'18 TRAC-1

Ignacio Arroyo-Fernández
Universidad Nacional Autónoma de México

Dominic Forest
Université de Montréal EBSI

Juan-Manuel Torres-Moreno
LIA - UAPV (France) & Polytechnique Montréal

Mauricio Carrasco-Ruiz
Universidad Nacional Autónoma de México

Thomas Legeleux
Université de Montréal EBSI

Karen Joannette
Université de Montréal EBSI

Abstract

The phenomenon of cyberbullying has growing in worrying proportions with the development of social networks. Forums and chat rooms are spaces where serious damage can now be done to others, while the tools for avoiding on-line spills are still limited. This study aims to assess the ability that both classical and state-of-the-art vector space modeling methods provide to well known learning machines to identify aggression levels in social network cyberbullying (i.e. social network posts manually labeled as Overtly Aggressive, Covertly Aggressive and Non-aggressive). To this end, an exploratory stage was performed first in order to find relevant settings to test, i.e. by using training and development samples, we trained multiple learning machines using multiple vector space modeling methods and discarded the less informative configurations. Finally, we selected the two best settings and their voting combination to form three competing systems. These systems were submitted to the competition of the TRACK-1 task of the Workshop on Trolling, Aggression and Cyberbullying. Our voting combination system resulted second place in predicting Aggression levels on a test set of untagged social network posts.

1 Introduction

The introduction of the Internet and its democratization in the public sphere has fostered the emergence of many sociological phenomena. It opens the possibility of forming friendly relations and information sharing from online networking platforms. These platforms, which are often the subject of strong ownership by young users, introduce a paradigm shift in interpersonal relationships since they are now interactive spaces where it is hard to put regulation rules in place. Thus, each time more we see the appearing of aggressive behaviors that were confined to the physical space until recently. These aggressions can take different forms: insults, intimidation, humiliation, exclusion, etc. All them have for common point to take place in a space where it is possible to cause serious moral damage without suffering the consequences, in particular because of the possibility of evolving in a completely anonymous way. This phenomenon, referred to as cyberbullying, or Internet harassment, is defined as *"the use of information and communication technologies to repeatedly, intentionally, and aggressively engage in behavior with respect to an individual or a group with the intention of causing harm to others"* – Belsey (2013).

Several solutions have been proposed, often based on a behavioral and pedagogical approach (ignore the attacker, confront him or even denounce him). However, it becomes necessary to think about relevant algorithmic strategies to better protect Internet users from cyberbullying. Such strategies would allow the establishment of an automated system to detect cyberbullying in social media. This assist moderators to identify the most serious cases, and thus to contribute to a safer virtual space for both the youngest and the adults.

In this paper we propose an approach based on multiple classical text mining pipelines. The aim of this is to explore the actual difficulties linguistic and extralinguistic phenomena may imply to effectively

Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying, pages 140–149
Santa Fe, USA, August 25, 2018.

identify aggression levels of cyberbullying in social network posts manually labeled as Overtly Aggressive (OAG), Covertly Aggressive (CAG) and Non-aggressive (NAG). Our exploratory approach has two main stages. First, we tested multiple vector space modeling methods along with multiple well known learning machines (classifiers) for predicting each of aggression levels individually. The vector space modeling techniques included TF–IDF vectors, Latent Semantic Analysis (LSA, of varied dimensionalities) of TF–IDF vectors. Both TF–IDF and LSA were computed for character and word $n-$gram features. In addition, we used word embedding-based representations of posts. Learning machines that were evaluated along with the aforementioned vector space modeling methods include Naïve Bayes, Linear Perceptron, Support Vector Machine (SVM) and Passive-Aggressive classifier. Once this first stage gives us several preliminary results, we selected the two configurations giving the best accuracies as competing systems for the first two runs. As a third run, we combined these two systems, which incorporates a random class generator based on class frequencies of the training and development sets. This class generator draws one of the three classes in the case of disagreement of the combined systems. This approach resulted in 2nd place of the Facebook competition dataset with 0.6315 of weighted F1 score. In the Social Media dataset, our best system resulted in 4th place with 0.5716 of weighted F1 score.

This paper is organized as follows: the section 2 shows the state of-the-art, section 3 describes the methodology and the COLING'18 TRAC-1 dataset used, section 4 shows the results and finally the section 5 conclude our paper.

2 State of the Art

There already exist software designed to combat the phenomenon of cyberbullying, e.g. Bsecure[1], CyberPatrol[2], Eblaster[3] or IamBigBrother[4]. The main drawback of these systems is that they are based on keyword filtering, which is a limitation because no statistical features of texts are taken into account. Further, these keyword filtering methods require manual maintenance.

To overcome the limitations of keyword filtering systems, (Yin et al., 2009) is one of the former attempts to detect cyberbullying by using statistical features: word frequency, analysis of *feelings* (use of pronouns in the second person, insults, etc.) and context. (Dinakar et al., 2011) built a system that can detect bullying elements in commentaries of YouTube videos. These were classified according to different representative categories (sexuality, intelligence, race and physical attributes). The classification revealed weaknesses and an increase in false positive cases. Researchers emphasized the importance of using common sense to understand users' goals, emotions, and relationships, thereby disambiguating and contextualizing language.

In (Berry and Kogan, 2010) the authors were also interested in a word search method based on a bag of words (BoW) system incorporating sentiment and contextual analysis. They build a decision tree that predicted intimidating messages with an accuracy rate of 93%. The researchers also have developed the Chatcoder software to detect malicious activities on-line (Kontostathis et al., 2012)[5].

In another study it was tested a system that allows users of a website to control the messages posted on their web pages: it customized vocabulary filtering criteria using a machine learning method that automatically labeled the contents. This approach had limitations because it was unable to measure relationships between terms beyond a certain semantic level (Davdar et al., 2012).

(Nahar et al., 2014) provided a concrete method for detecting on-line harassment by measuring the score of sent and received messages (and thus their degree of involvement in a conversation) using the Hyper link-Induced Topic Research algorithm (HITS). The authors also proposed a graphical model that identifies the aggressors and their most active victims.

Other studies have attempted to go further by seeking to take into account more specific characteristics. (Davdar et al., 2012) tried to establish a system based on language features characterizing the author's

[1] http://www.bsecuregroup.com/
[2] https://www.cyberpatrol.com/
[3] https://www.veriato.com/
[4] http://www.parentalsoftware.org/bigbrother.html
[5] http://www.chatcoder.com/

genre of comments on MySpace. Their results revealed an improvement in the detection of bullying when this information is taken into account.

As we can see, recent work defines the means to respond to the cyberbullying phenomenon that is becoming more and more widespread as the use of the web does. This paper addresses such a phenomenon and discusses a number of approaches to address it.

3 Methodology and Data

The methodology employed in this study has two main parts. First, we explored the hypothesis that different levels of aggression have different difficulty levels of identification. This hypothesis implies that the classifiers behave differently for different levels of aggression, so we decided to explore each level separately in an OVR (One-Versus-Rest) approach. Furthermore, we hypothesized that Covered Aggressions (CAG) are harder to identify than the other ones. This is because of the undirected way things can be expressed by users. A high pragmatic component is present in the form of sarcasm, which requires to decode extra-linguistic information. Thus, during this first stage, a number of learning algorithms were trained along with different text representations of their input vector space modeling.

Given that this first stage was mainly an exploratory one, assessing the actual difficulties of identifying each level of aggression also involves exploring well-known text representation methods. This can be achieved, on one hand, by using classical methods such as TF-IDF (Spärk-Jones, 1972; Torres-Moreno, 2014), Bag of Words (BoW, simple word counts) and Singular Value Decomposition (SVD). And, on the other hand, we used an easy-to-use word embedding-based method allowing to observe whether Aggression Identification demands more representation ability than what is provided by traditional methods.

Exploring separately the difficulty of identification of each aggression level aims to be highly illustrative of the properties of the Aggression Identification task. Also, it aims to select the machine learning algorithms and the text representation methods that best perform in most of the exploratory experiments.

The second stage was to compete in three runs. To this end, we first used the two combinations of machine learning algorithms with text representations that best performed during the exploratory stage (out of the four that we tested). These combinations were submitted as two independent systems representing one run each. In addition, we combined these two systems as a fusion system (multi-agent system) that was submitted as our third run.

3.1 Dataset

The TRAC-1 dataset (Kumar et al., 2018) has been used in order to train our systems. The available dataset is composed of samples with a variable class distribution according to the task. It is divided into two subsets: a training set and a development set. The training set contains 12014 instances. 4241 of them are labeled as Covertly Aggressive (CAG), 5055 of them are labeled as Non-aggressive (NAG), and 2708 of them are labeled as Overtly Aggressive (OAG). The development dataset is constituted by 3003 samples. 1058 of them are labeled as CAG, 1233 of them are labeled as NAG and 711 of them are labeled as OAG. See that there is a class imbalance mainly affecting the distribution of the OAG class.

3.2 Methods

3.2.1 Classical Vector Space Modeling using TF-IDF

Term Frequency-Inverse Document Frequency (TF-IDF) is a document representation method originally used in Information Retrieval (Spärk-Jones, 1972; Torres-Moreno, 2014). This method represents a document d_i by building its associated sparse vector $x_i = (x_1, \ldots, x_n)^\top \in \mathbb{R}^n$. Each of its components $j = 1, \ldots, n$ is a weight associated to a word w_j in the vocabulary of a collection of documents $D = \{d_i\}_{i=1}^m$. In general, this weight can be seen as the information amount gained from D as w_j is observed in d_i. Each component $x_j \in \mathbb{R}$ of the TF–IDF representation x_i of a document d_i is given by equation (1):

$$x_j = \frac{f_{ij}}{1 + \log_2 m + \log_2 m_j} \tag{1}$$

where m is the total number of documents in the collection, m_j is the number of documents that share the word w_j, and f_{ij} is the frequency of occurrence of w_j within d_i. It has been showed that when the documents are small, a binary version of f_{ij} behaves better than raw frequencies (Salton et al., 1983), so f_{ij} can be defined as $f_{ij} = 1$ if $w_j \in d_i$ and $f_{ij} = 0$ otherwise. Furthermore, in our experiments we computed TF-IDF sparse representations both for character-based and for word-based $n-$grams.

3.2.2 Word Embedding-Based Method

In the NLP literature there are reported a number of text embedding methods at the sentence level. Among the State of the Art methods there is an easy to use one which uses a combination of word embeddings weighted with TF-IDF (*Word Information Series for Sentence Embedding*, WISSE). The word embeddings used were FastText of 300 dimensions (Joulin et al., 2016), which have been reported to performe well in sentence representations for Semantic Textual Similarity tasks (Cer et al., 2017; Arroyo-Fernández et al., 2017). WISSE simply represents a document d_i as in equation (2):

$$x_i = \sum_{w_j \in d_i} \varphi_j x_j, \tag{2}$$

where $x_i \in \mathbb{R}^n$ is the document/sentence representation of d_i and $x_j \in \mathbb{R}^n$ is the word embedding of w_j. φ_j is the TF–IDF weight of w_j. This weight represents the information amount provided by x_j to x_i given that w_j is observed in d_i and in D (the whole set of FB and SM posts).

3.2.3 Support Vector Machines

Let $\mathcal{X} = \{x_1, \ldots, x_n\}$ be a set of samples, $\mathcal{Y} = \{y_1, \ldots, y_n : y_i \in \{\text{CAG}, \text{NAG}, \text{OAG}\}\}$ a set of labels and $\mathcal{D} = \{(x_i, y_i) : x_i \in \mathcal{X}, y_i \in \mathcal{Y}\}$ a dataset of training examples. A Support Vector Machine (SVM) builds a model $f(x)$ such that given a new arbitrary point x it returns the category $y_i \in \mathcal{Y}$ the point belongs to (Cortes and Vapnik, 1995). The model is given by equation (3):

$$f(x) = \sum_{i=1}^{\ell} \alpha_i k(x_i, x) + b \tag{3}$$

where $k(\cdot, \cdot)$ is a kernel function acting as an inner product allowing to capture non-linear relationships among data points. The linear case can be considered by setting $k(x_i, x) = \langle x_i, x \rangle$. The so called dual coefficients α_i are associated to each training point in $\mathcal{X}$ and they indicate how important is each training point in representing $\mathcal{D}$. The coefficients are trained by means of linear programming with ℓ_1 regularization. Thus, after the training, most these coefficients result very small or zero. The remaining non-zero coefficients are associated to the so called support vectors, which are used to model the data.

For Aggression identification the SVM model is a representation of the Internet posts mapped so that they can be divided by a clear gap that is as wide as possible so that each division indicates a category.

3.2.4 Passive-Aggressive Classifier

Let $\mathcal{X} = \{x_1, \ldots, x_n\}$ be a set of samples, $\mathcal{Y} = \{y_1, \ldots, y_n : y_i \in \{\text{CAG}, \text{NAG}, \text{OAG}\}\}$ a set of labels and $\mathcal{D} = \{(x_i, y_i) : x_i \in \mathcal{X}, y_i \in \mathcal{Y}\}$ a dataset of training examples. A Passive-Aggressive learning algorithm (PA) updates a vector of weights w_i that parametrize a linear decision boundary $f(x_i) = \langle w_i, x_i \rangle$ if this vector causes an error in the training for a sample $x_i \in \mathcal{X}$. This modification is said to be aggressive. In case there is no error, the vector of weights is unchanged, i.e. $w_{i+1} = w_i$, and then the algorithm is said to be passive (Crammer et al., 2006). The update rule is given by equation (4).

$$w_{i+1} = w_i + \tau_i y_i x_i \tag{4}$$

where y_i is the training label of the $i-$th training example $(x_i, y_i) \in \mathcal{D}$ and w_{i+1} is the updated version of w_i. It's similar to the delta rule originally proposed for training one-layer neural networks (Haykin, 2009). The parameter τ_i is given by equation (5):

$$\tau_i = \min \left\{ C, \frac{L_i(w_i, x_i)}{\|x_i\|^2} \right\} \tag{5}$$

where C is a manually set parameter that regulates *aggressiveness* (which is commonly termed as *momentum* in neural networks literature). L_i is the hinge loss $L_i(w_i, x_i) = \max\{0, 1 - y_i\langle w_i, x_i\rangle\}$, which indicates whether an error occurred and its magnitude. The version of PA explained here is the simplest one, and it allows to see a general portrait of this kind of algorithms. Nonetheless, a number of different versions of them are shown in detail in (Crammer et al., 2006).

3.2.5 Classifier Fusion Using Class Distribution

Our third run was a fusion of the two classifiers having the best performance on the validation dataset. The fusion is quite simple: for each sample, if the classifiers - Passive-Aggressive and SVM - are in agreement, the predicted class is assigned to the document. Otherwise, if the classifiers are in disagreement, a third classifier (based on the most statistically probable label from the class distribution of the training set) decides the final output by using a simple vote.

4 Results and Discussion

4.1 Results of the Exploratory Stage (training and development datasets)

In the Figures 1-3 we show results (in terms of accuracy) of four approaches, i.e. Naïve Bayes (NB), Perceptron, SVM and Passive-Aggressive (PA), in identifying independently each of the three levels of aggression (OAG, CAG, NAG) in a OVR fashion. To model the input vector space of these algorithms we used two baseline representation methods, i.e. Hashing (Bag of Words, BoW) and TF-IDF, and one state-of-the-art word embedding-based method, i.e. WISSE. One of the main things we observed during the exploratory stage was how the amount of training samples influenced the accuracy of the classifiers. Therefore we plotted the accuracy as a function of the amount of training samples (from the whole training set). The accuracy was measured on the development dataset.

The results show that NB, PA and SVM classifiers perform in similar ways. However, the Perceptron showed to be a bit unstable and performed the worst. In most cases, the classifiers attained relatively stable performance after 5000 training samples. In this sense, with respect to NAG and CAG, identifying OAG aggressions represented much less complexity as a classification problem ($2000 - 3000$ TF-IDF-transformed samples were required by classifiers to stabilize their performance). On the other hand, identifying CAG resulted in a much more complex task for the classifiers. Perceptron was again the worst while Naïve Bayes and linear SVM performed better, but barely surpassed 65% accuracy. Detecting NAG aggressions required much more samples to allow the classifiers to be relatively stable. Furthermore, it was hard for the classifiers to reach $73 - 74\%$ of accuracy. Overall TF-IDF representations allowed the classifiers to be much more stable than Hashing did.

Two additional experiments were conducted for exploring baseline representations on CAG detection in more detail. First, we used Singular Value Decomposition (SVD) for dimensionality reduction on the word-based TF-IDF representations of the posts. Although a number of dimensionalities were tested (50, 100, 200, 300, 400), this modification neither showed improvements with respect to sparse TF-IDF representations. Nonetheless, training time increased considerably as the implementations are better prepared for sparse presentations. Secondly, we segmented documents into character n-grams before represent them by means of TF-IDF. Surprisingly the classification accuracy was much better ($+10\%$) in general. This time, the Passive-Aggressive and linear SVM classifiers attained about 70% of accuracy in CAG identification by segmenting the input posts into a range of $[1 - 5]$ character-based n-grams. To observe this performance, more than 9000 samples were needed (See Figures 1-3). Other n-gram ranges did not perform better (e.g. $2 - 5$, $3 - 5$, $1 - 6$, $2 - 6$, etc.).

We also used word embedding-based representations to represent the posts as sentence embeddings. Even when these embeddings showed state-of-the-art performance in STS, their performance in Aggression Identification was not better than the baseline methods presented above (see Figures 4 and 5). The behavior of classifiers with sentence embeddings showed much more unstable and the performance diminished by 5% in general with respect to the character-based TF-IDF sparse representations. Although the sparse representations are high dimensional (thousands of dimensions), 5% or less of their entries are nonzero. Most state-of-the-art implementations of the classifiers are prepared to deal efficiently with this

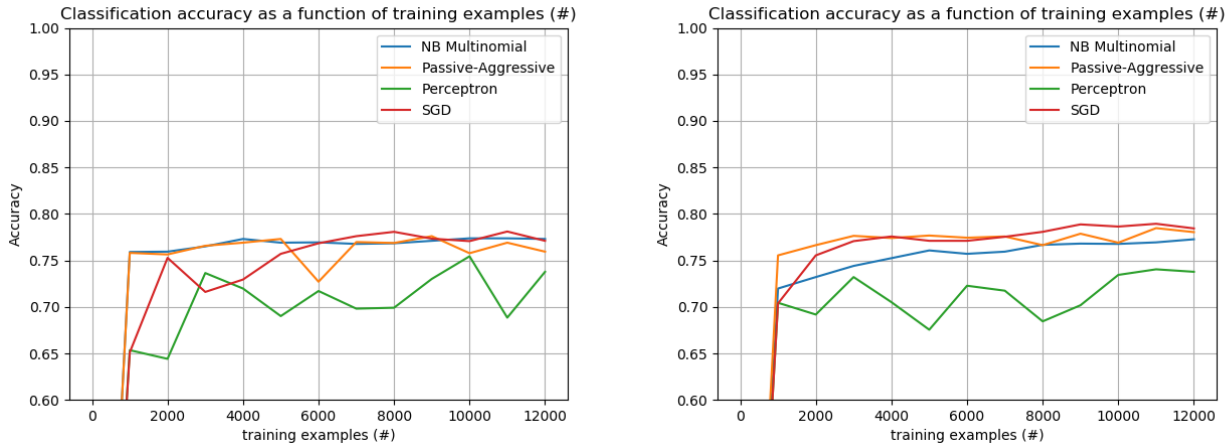

Figure 1: One versus rest accuracy of four learning machines in predicting Overtly Aggressive (OAG) Facebook posts (development dataset). The left hand side plot are accuracies of classifiers on (hashing) Bag of words vectors. The right hand side are plots of accuracies of same classifiers on TF–IDF vectors.

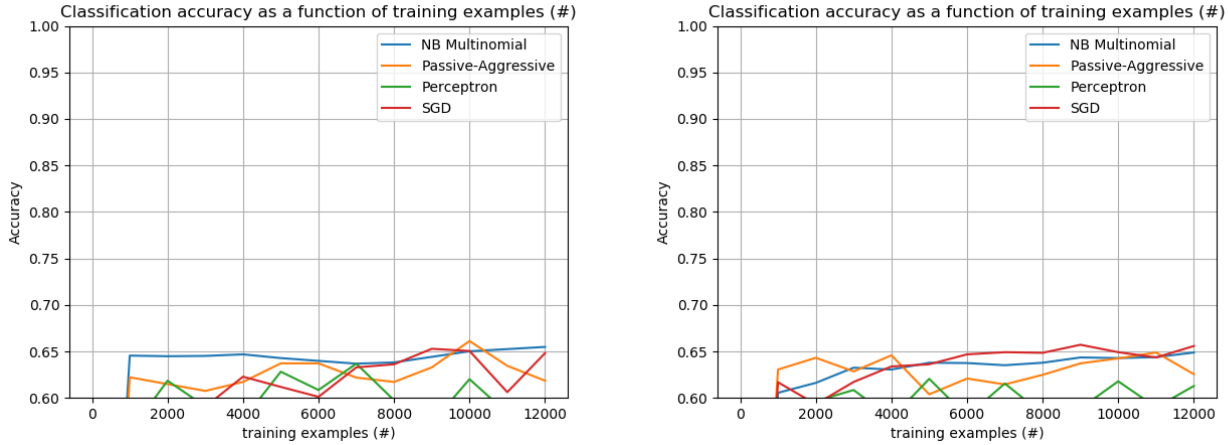

Figure 2: One versus rest accuracy of four learning machines in predicting Covertly Aggressive (CAG) Facebook posts (development dataset). The left hand side plot are accuracies of classifiers on (hashing) Bag of words vectors. The right hand side are plots of accuracies of same classifiers on TF–IDF vectors.

issue. Therefore, the classifier ends by learning from 100 or less nonzero entries by sample. Conversely, sentence embeddings based on word embeddings are dense representations of 300 dimensions, which adds complexity to the learning problem (varying dimensions of the embeddings did not offered improvements and the learning instability held). Furthermore, the original purpose of sentence embeddings is to represent an approximation of each word semantics, and then of the whole sentence semantics. This may result in much more complex patterns represented by sentence embeddings than what is needed for aggression identification.

This additional complexity in their input space can lead to over-fitting of the classifiers.

4.2 Results of the Competition Stage (test dataset)

The Aggression Identification competition required participants to apply their systems on a test dataset. This dataset consists of two files. The first one having a class distribution similar to the training set, and the second one with a so-called "surprise" configuration.

Three runs were conducted on the TRAC-1 test dataset, that was splitted into two files: Facebook and Social Media. There are 916 samples on the Facebook set, and 1257 samples on the Social Media set

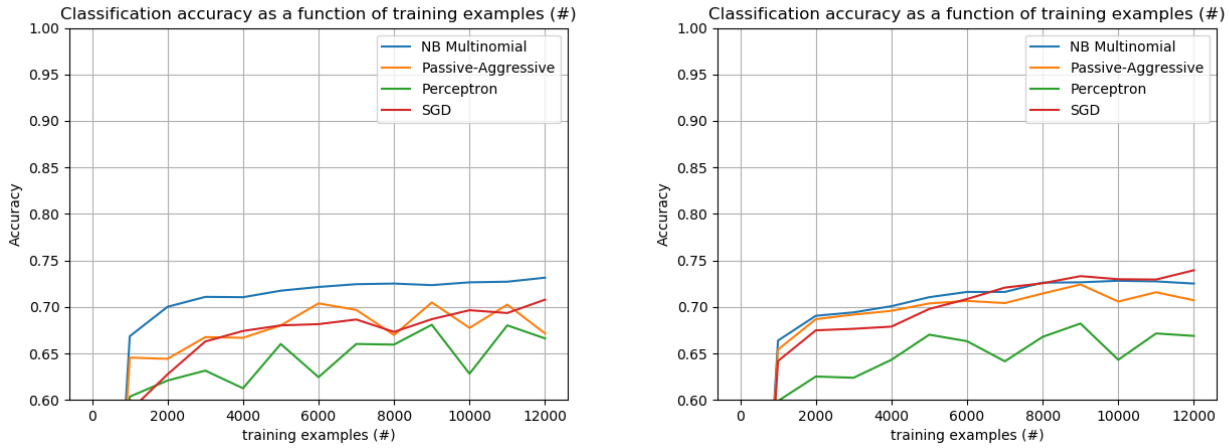

Figure 3: One versus rest accuracy of four learning machines in predicting Non-aggressive (NAG) Facebook posts (development dataset). The left hand side plot are accuracies of classifiers on (hashing) Bag of words vectors. The right hand side are plots of accuracies of same classifiers on TF–IDF vectors.

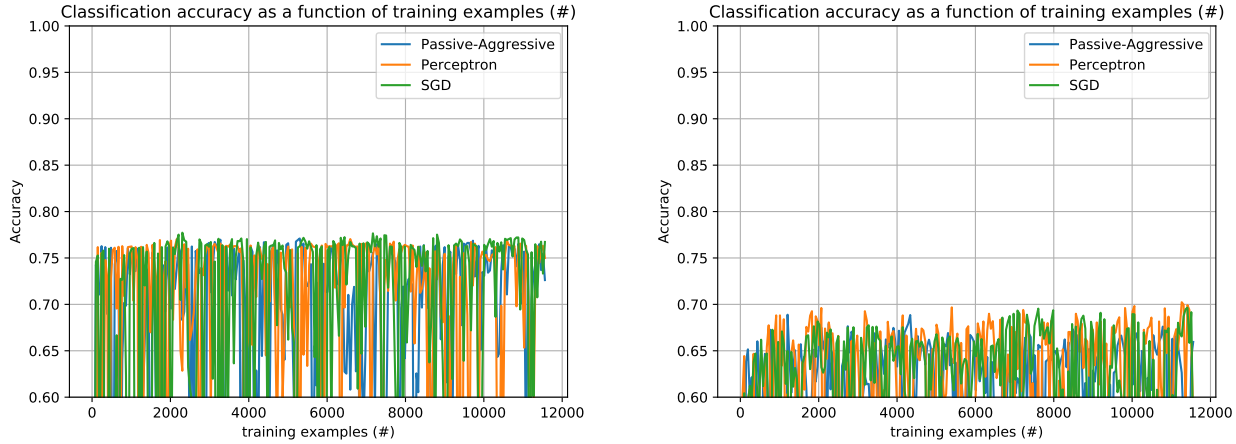

Figure 4: One versus rest accuracies of PA, Perceptron y SVM classifiers on Overtly Aggressive (left) and Non-aggressive (right) Facebook posts (development dataset). The input embeddings were WISSE of 300 dimensions.

(Kumar et al., 2018). During the exploratory stage we selected the two classifiers that performed the best in most experiments conducted, i.e. the Passive-Aggressive (PA) and the SVM classifiers. Therefore, our run 1 was executed by using a PA classifier whose hyperparameters were the best ones found via 3-fold cross-validation on the training dataset; the run 2 was executed by a SVM classifier whose hyperparameters were the best ones found with same method than for PA. And the run 3 was the fusion of both coupled with a probabilistic distribution of the classes. The PA and SVM implementations we used are the provided ones by (Pedregosa et al., 2011) (for the SVM we used the SGD version, Stochastic Gradient Descent). Furthermore, we selected the vector representation that best performed during the exploration stage, i.e. character based n-gram TF-IDF sparse representations with $n \in \{1, 5\}$. Because their low performance during the exploratory stage, we decided do not use the word embedding-based representations during the competition stage.

Tables 1 and 2 shows our rank position on the English Facebook (2nd place) and Social Media (4th place) tasks respectively.

The Figures 6 and 7 shows the confusion matrix (True label vs. Predicted label) measured on the English Facebook and Social Media files respectively. The confusion matrix in the case of Facebook

Figure 5: One versus rest accuracies of PA, Perceptron y SVM classifiers on Covertly Aggressive (CAG) Facebook posts (development dataset). The input embeddings were WISSE of 300 dimensions.

Rank	System	F1 (weighted)
1	saroyehun	0.6425
2	**EBSI-LIA-UNAM**	**0.6315**
3	DA-LD-Hildesheim	0.6178
4	TakeLab	0.6161
5	sreeIN	0.6037
...	...	...
30	bhanodaig	0.3572
	Random Baseline	*0.3535*

Table 1: Results for the English (Facebook) task. Our model EBSI-LIA-UNAM (ELU) is placed in 2nd rank.

Rank	System	F1 (weighted)
1	vista.ue	0.6008
2	Julian	0.5994
3	saroyehun	0.5920
4	**EBSI-LIA-UNAM**	**0.5716**
5	uOttawa	0.5690
...	...	...
	Random Baseline	*0.3477*
...	...	...
30	bhanodaig	0.1960

Table 2: Results for the English (Social Media) task. Our model EBSI-LIA-UNAM (ELU) is placed in 4th rank.

data, shows that the OAG and CAG classes are the most difficult to classify (with 52% error rate for OAG, 53% error rate for CAG). For the Social Media file, the CAG and OAG classes were the most difficult to categorize (with 53% error rate for OAG, 64% error rate for CAG). In both sets, the NAG class was detected correctly (33% and 11% error rate).

5 Conclusion

The methodology that we adopted allowed us to observe in broad strokes the complexity of the problem in question. On the one hand, word embeddings are possibly trained to represent much more complex or

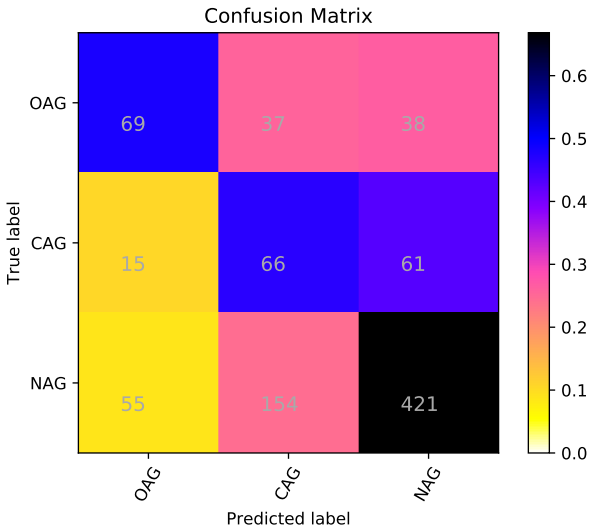

Figure 6: EN-Facebook task, EBSI-LIA-UNAM Run 03.

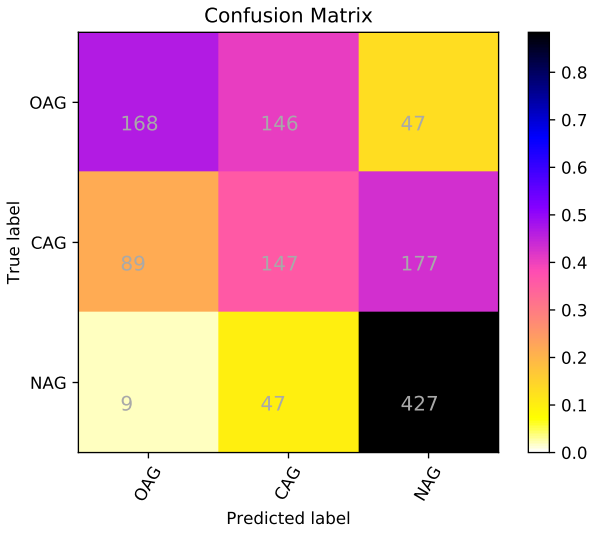

Figure 7: EN-Social Media task, EBSI-LIA-UNAM Run 03.

detailed patterns in the text (e.g., distributional semantics), which leads to over-fitting that is reflected in the instability of the predictions of the classifiers. Probably the word embedding approach is not the best option for this task. Indeed, text representations with classical methods show much better adaptation to this problem, and are much less complex to interpret than word embeddings. Finally, the datasets provided for the task allow for taking into account only linguistic evidence, we think better results require additional data contextualizing other aspects of aggressions in cyberbullying. For example, sarcasm, a pragmatic phenomenon, requires extra-linguistic knowledge to be processed properly. We would have liked to include a sentiment analysis resource in our experimentation. Finally, further processing of the data might have provided us with additional information. For example, it might have been possible to use the degree of severity associated with certain forms to refine our analysis, or to try to specify whether an insulting sentence implies a response that also contains intimidation.

The use of the Levenshtein algorithm could have allowed a more precise identification of insults. This algorithm takes into account changes in character strings, which is particularly suitable for discussions on social networks where non-homogeneous language is used (Baetens, 2013).

References

Ignacio Arroyo-Fernández, Carlos-Francisco Méndez-Cruz, Gerardo Sierra, Juan-Manuel Torres-Moreno, and Grigori Sidorov. 2017. Unsupervised Sentence Representations as Word Information Series: Revisiting TF-IDF. *arXiv preprint arXiv:1710.06524*.

M. Baetens. 2013. La détection automatique de cas de cyber harcèlement textuel dans les médias sociaux. Master's thesis, Université d'Anvers, Belgium.

B. Belsey. 2013. Cyberbullying research center. Technical report, https://cyberbullying.org/.

Michael W. Berry and Jacob Kogan, editors. 2010. *Text Mining: Applications and Theory*. Wiley, Chichester, UK.

Daniel M. Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In Steven Bethard, Marine Carpuat, Marianna Apidianaki, Saif M. Mohammad, Daniel M. Cer, and David Jurgens, editors, *SemEval@ACL*, pages 1–14. Association for Computational Linguistics.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20(3):273–297.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, December.

Maral Davdar, Franciska de Jong, Roeland Ordelman, and Dolf Triechnigg. 2012. Improved cyberbullying detection using gender information. In *Dutch-Belgian Information Retrieval Workshop DIR 2012*, pages 23–25.

Karthik Dinakar, Roi Reichart, and Henry Lieberman. 2011. Modeling the detection of textual cyberbullying. In *The Social Mobile Web*, volume WS-11-02 of *AAAI Workshops*. AAAI.

Simon S. Haykin. 2009. *Neural networks and learning machines*. Pearson Education, Upper Saddle River, NJ, third edition.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. Fasttext.zip: Compressing text classification models. arxiv:1612.03651 - ICLR 2017.

April Kontostathis, Andy Garron, Kelly Reynolds, Will West, and Lynne Edwards. 2012. Identifying predators using chatcoder 2.0. In *CLEF (Online Working Notes/Labs/Workshop)*.

Ritesh Kumar, Aishwarya N. Reganti, Akshit Bhatia, and Tushar Maheshwari. 2018. Aggression-annotated Corpus of Hindi-English Code-mixed Data. In *Proceedings of the 11th Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan.

Vinita Nahar, Xue Li, Hao Lan Zhang, and Chaoyi Pang. 2014. Detecting cyberbullying in social networks using multi-agent system. *Web Intelligence and Agent Systems*, 12(4):375–388.

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

Gerard Salton, Edward A Fox, and Harry Wu. 1983. Extended boolean information retrieval. *Communications of the ACM*, 26(11):1022–1036.

Karen Spärk-Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(5):111–121.

Juan-Manuel Torres-Moreno. 2014. *Automatic Text Summarization*. ISTE Ltd, John Wiley & Sons, Inc., London.

Dawei Yin, Brian D. Davison, Zhenzhen Xue, Liangjie Hong, April Kontostathis, and Lynne Edwards. 2009. Detection of harassment on web 2.0. In *1st Content Analysis in Web 2.0 Workshop, CAW 2.0*, Madrid, Spain.