HATERecognizer at SemEval-2019 Task 5: Using Features and Neural Networks to Face Hate Recognition

Victor Nina-Alcocer
Department of Computer Systems and Computation
Universitat Politècnica de València, Spain
vicnial@inf.upv.es

Abstract

This paper presents a detailed description of our participation in task 5 on SemEval-2019[1]. This task consists of classifying English and Spanish tweets that contain hate towards women or immigrants. We carried out several experiments; for a finer-grained study of the task, we analyzed different features and designing architectures of neural networks. Additionally, to face the lack of hate content in tweets, we include data augmentation as a technique to increase hate content in our datasets.

1. Introduction

HatEval (Basile et al., 2019) aims to identify cases of aggressiveness and hate speech towards women and immigrants in social media considering tweets messages written in English and Spanish. This task defines two main sub-tasks:

- Task1. Hate Speech Detection against Immigrants and Women: predict whether a tweet in English or in Spanish is hateful or not hateful.
- Task2. Aggressive behavior and Target Classification: to identify if a tweet is aggressive or not aggressive, and to identify the target harassed as individual or generic.

To tackle the subtasks mentioned above we will use Natural Language Processing (NLP) and Machine Learning (ML) fields to propose two approaches. The first approach pretends to know more about valuable features that allow us to get a good understanding of its embedded knowledge. To get this knowledge we study four features that we consider important: its structure, its embedded emotions, patterns on its pos tagging and skip-grams. Each of these features will allow us to know if tweets have some patterns that can be used to discriminate tweets that contain hate or not.

The second approach will try to recognize patterns using weights among neurons. To implement this focus we designed several architectures of neural networks (NN), which were fed with different kinds of corpora that were processed considering many aspects such as, lemmatization, stemming, normalized hashtags, etc.

This work is organized into three sections. The first section provides an introduction to this paper. The second one describes the proposed approaches, the experiments conducted, and the results that we achieved. The third section presents some conclusions.

2. Systems Description

In this section, we present the main features of the two approaches considered in this paper and its respective experiments.

The organizers provided a training dataset of 9000 and 4500 tweets written in English and Spanish labeled with hate speech (HS), Aggressive behavior (AG) and Target (TR). For what concerns HS the distribution is almost balanced among hate(42 %) and no-hate (58 %) tweets in both languages. Regarding AG and TR, the distribution is skewed towards tweets that do not contain TR (75 %) or AG(67 %) respectively. In order to asses, the performance of the systems, tests set of 3000 and 1600 unlabeled tweets were provided. The official evaluation metrics to evaluate the systems were: For the task1, accuracy (Acc.), precision (P), recall (R), and F1-score. For task2, The models were evaluated using EMR and F1-score as describe in (Basile et al., 2019)

2.1. Based on Classical ML

With this approach, we try to face hate detection through classical machine learning algo-

[1] http://alt.qcri.org/semeval2019/index.php?id=tasks

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 409–415
Minneapolis, Minnesota, USA, June 6–7, 2019.

rithms. Firstly, we established a baseline (called «our baseline») using Term frequency - Inverse document frequency (TF-IDF) scheme based on words and Support Vector Machine (SVM) as a supervised learning model. This baseline gives us an idea of whether the experiments that consider new features or simply the use of other machine learning algorithms help us to achieve good results.

Preprocessing: The preprocessing consisted of ridding of URLs, numbers, users, times, date, email, percents. But, we decided to keep normalized hashtags, elongated words, repetitions, emphasis, and censored words. Next, we present an example to show a raw and normalized version of a tweet:

Raw tweet:

@KamalaHarris Illegalsssssss Dump their Kids at the border like Road Kill and Refuse to Unite! They Hope they get Amnesty, Free Educatioon and Welfare Illegal #FamilesBelongTogether in their Country not on the Taxpayer Dime Its a SCAM #NoDACA #NoAmnesty #SendThe https://t.co/Ks0SHbtYqn[2]

Normalized tweet:

illegals dump their kids at the border like road kill and refuse to unite they hope they get amnesty free education and welfare illegal familesbelongtogether in their country not on the taxpayer dime its a scam nodaca noamnesty sendthe

Feature Extraction and Selection. As in (Schmidt and Wiegand, 2017) our models considered a set of features. The first one takes into account the *structure of tweets*, with this focus we are going to consider and count if tweets have hashtags, punctuation marks, presence and its *Unicode Common Locale Data Repository* (CLDR)[3] version of emojis, capital letters, number of words, numbers of user and so on. To get the second group of features was necessary to consider *pos tagging* as described (Bretschneider et al., 2014), but analyzing some patterns on how people usually write or use hate content in tweets as was mentioned by (Silva et al., 2016), for instance, we noticed in the dataset that people usually use this pattern to denigrate a woman: "eres una maldita zorra" now to discover some pattern we show its pos tagging representation below:

eres ***AUX*** *una* ***DET*** *maldita* ***ADJ_Gender=Fem*** *puta* ***NOUN_Gender=Fem*** *zorra* ***ADJ_Gender=Fem*** *.* ***PUNCT_PunctType=Peri***

As we know a sentence is composed by verbs, prepositions, adjectives, adverbs and so on. In this particular sentence shown above, we see that is composed by auxiliaries, determiner, adjectives, noun and a punctuation mark. All these tags plus the gender of each word will give us a piece of important information to discriminate hate towards women and immigrants. The third feature that we consider important is to try to *find out if a word is pejorative, derogative, offensive or vulgar*. To get this aim we created a lexicon with these four categories already mentioned before. In short, we selected words that we called "seeds" then we start to get synonyms or related words using several sources[4]. Using this method we got more than 4900 words classified in the four mentioned categories. Some of the words and its respective categories are shown in Table 1.

Word	Pejorative	Derogative	Offensive	Vulgar
* solterona (es)	✓	-	-	-
* puta/puta de quinta (es)	✓	-	✓	-
* idiota (es)	-	✓	-	-
* huevon (es)	-	-	✓	-
* asswhore (en)	-	-	✓	-
* fucknigga/nigga (en)	-	-	✓	-
* pisslamist (en)	-	-	-	✓
* dogfucker (en)	-	-	-	✓

Table 1: Categories considered in our hate_lexicon.

We noticed that extending a little bit more the four categories in the third feature, we can improve slightly our results. Therefore, we take into account *the percentage of embedded emotions* in tweets, for this goal we used Linguistic Inquiry and Word Count (LIWC)[5]. And we chose some additional categories (sexual, anxiety feeling, anger and so on.) that help us to discriminate between tweets which contents hate or not. And finally, we have considered using TF-IDF schemes over skip-grams (bigram and trigrams) (Davidson et al., 2017).

In these experiments, we used scikit-learn to test several machine learning algorithms and its respective parameters. Some algorithms that we used

[2]This raw tweet belongs to the original training dataset and it was intentionally rewritten with typos to illustrate which considerations were taken into account to its normalization.

[3]http://cldr.unicode.org/translation/short_names_and_keywords

[4]https://www.dictionary.com

[5]https://www.receptiviti.ai/liwc-api-get-started

Approach	Task1	Task1	Task2	Task2	Task1	Task1	Task1	Task1
	English	English	English	English	Spanish	Spanish	Spanish	Spanish
	Acc.	F1-sc.	F1-sc.(avg)	EMR	Acc.	F1-sc.	F1-sc.(avg)	EMR
OUR BASELINE(See section 2.1)	*0.763*	*0.766*	*0.731*	*0.629*	*0.834*	*0.840*	*0.785*	*0.710*
TFIDF+POS	0.769	0.770	0.740	0.633	0.840	0.837	**0.792**	**0.719**
TFIDF+EMOTIONS	0.759	0.749	0.747	0.711	**0.85**	**0.847**	0.788	0.698
TFIDF+STRUCT	**0.771**	**0.780**	**0.75**	**0.648**	0.790	0.798	0.789	0.702
TFIDF+POS+EMOTIONS	↓	↓	↓	0	0.849	0.842	0.791	0.700
TFIDF+POS+STRUCT	↓	↓	↓	0	0.810	0.800	0.787	0.700
TFIDF+EMOTIONS+POS	↓	↓	↓	0	0.846	0.839	0.781	0.698
TFIDF+SKIP-BIGR.	↓	↓	↓	↓	↓	↓	↓	↓
TFIDF+SKIP-TRIGR.	↓	↓	↓	↓	↓	↓	↓	↓
TFIDF+SKIP-BIGR.+EMOTIONS	↓	↓	↓	↓	↓	↓	↓	↓
TFIDF+SKIP-BIGR.+STRUCT	↓	↓	↓	↓	↓	↓	↓	↓
TFIDF+SKIP-BIGR.+EMOTIONS+STRUCT	↓	↓	↓	↓	↓	↓	↓	↓
TFIDF+SKIP-TRIGR.+EMOTIONS	↓	↓	↓	↓	↓	↓	↓	↓
TFIDF+SKIP-TRIGR.+STRUCT	↓	↓	↓	↓	↓	↓	↓	↓
TFIDF+SKIP-TRIGR.+EMOTIONS+STRUCT	↓	↓	↓	↓	↓	↓	↓	↓

Table 2: Results achieved using LinearSVC algorithm for English and Spanish development datasets.

were: RandomForestClassifier (Breiman, 2001), LogisticRegression, Naive Bayes (Kibriya et al., 2004), LinearSVC (Hearst, 1998) and so on (Fortuna, 2017). A summary of applying this approach to the developing dataset and its respective results are shown in Table 2. This table shows just results using LinearSVC because it was the algorithm that achieved good results. Other variations of corpus like stemming or lemmatization version were not used because we did not achieve good results.

Basically, we can highlight in table 2, that the use of skip-grams does not help a lot, this fact is due to the lack of skip-grams contained in tweets, for instance, in the tweet mentioned before, the skip-trigram «illegals dump kids» is just repeated once in the whole dataset. A similar situation happens with skip-bigrams and skip-trigrams. On the other hand, the use of the structure and emotions (lexicon) embedded in tweets help to improve our baseline in both languages respectively. We used down arrow symbol ↓ to show that the value is below of our baseline, so is not valuable include it.

2.2. Based on NN

This second approach tackle hate speech detection trough neural networks (Schmidt and Wiegand, 2017). In this stage, we used several neural networks architectures which works enough well with text, some of them are: Convolutional Neural Networks (CNN) (Jacovi et al., 2018), Long short term memory (LSTM) and its variations: Peephole, Bidirectional (BILSTM) and Gated Recurrent Unit (GRU) (Lu and Salem, 2017). We tested these architectures with different corpora which were slightly modified, in some cases we used a corpus with stop words, or simply a stemming or a lemmatized version of words. Additionally, we used other variations to see which of them help us to improve the performance of the system, the next paragraph describes how the corpora were processed.

Preprocessing: Is valuable to mention that the preprocessing of raw tweets was slightly different for machine learning and NN, because we saw that keeping an identical preprocess for both approaches do not help a lot. So we decided to customize the preprocess for each focus as we describe in section 2.1 and section 2.2. Swear words were kept for both focuses because they helped to achieve better results. Furthermore, emojis were normalized using its CLDR short name, For instance, we changed 🙂 for ':smiling_face_with_smiling_eyes:' cleaned (without any additional symbol).

For this particular preprocess for NN were taken to account many aspects that were already considered for ML's preprocessing. Perhaps the main difference is when we process the hashtag. For instance, its version normalized for the raw tweet shown in section 2.1, would be:

illegals dump their kids at the border like

road kill and refuse to unite they hope they get amnesty free education and welfare illegal families belong together in their country not on the taxpayer dime its a scam no daca no amnesty send the

As is shown, this version separates the content of hashtags, additionally, we correct typos using dictionaries[6]. For instance, we changed "familes" by "families" in the above tweet.

This preprocess used for NN was used for ML too, but we noticed that accuracy got down in our experiments. Then we decided to keep the original way of hashtags for ML (no splitting).

The results shown in Table 3 were achieved using the developing datatset with the preprocess commented in section 2.2. Summing up, stop words, normal words (no stemming, no lemmatization), normalized hashtags, spelling corrections were considered. Furthermore, word embeddings (Mikolov et al., 2013) (Goldberg, 2015) as efficient word representation were used for all the experiments in this stage.

Carrying on several experiments we realized that Convolutional Neural Networks performed well enough compared to other architectures, see Table 3. Therefore, we named this architecture as *architecture A*. This architecture is compound by an input embedding layer, followed by and spatial dropout and next to a convolutional layer with a set up of 256 filters and kernel size of 2. Next a globalmaxpooling is defined and finally, a dense layer of 2 neurons with softmax is used to get the results. The good performance of this architecture encouraged us to use different word embeddings to improve accuracy, the results of these experiments are shown in Table 4.

As we can see in Table 4, there is a lack of results on the Spanish language, this fact is due that we could find in Spanish all the corresponding resources exploited for experiments with English portion of the data. Sometimes some authors publish word embeddings for this specific language but mostly this embeddings does not fit with our study case due to the language used in twitter is kind of informal and use a lot of slang, swear words and so on. And this important aspect reduces the performance of our system.

Data Augmentation: As we commented at the beginning of section 2, we noticed a slight skew in favor of tweets that do not contain HS, AG or TR. To deal with this unbalanced training dataset, like in (Risch and Krestel, 2018) we adopted a technique which allows us to increase the number of tweets with hate content. To do this we analyzed some ways to get synonyms or similar words. Firstly, we try to use synonyms only if these synonyms do not change the meaning of the whole sentence. This fact is challenging because is kind of hard to use synonyms to increase our dataset. For instance, the next tweet: *"Hurray, saving us $$$ in so many ways potus realDonaldTrump #LockThemUp #BuildTheWall #EndDACA #BoycottNFL #BoycottNike"* has to be cleaned and it has to just keep words to get its synonyms. The result to process the tweet above is: *"hurray saving us in so many ways lock them up build the wall end daca boycott nfl boycott nike"*. Now, we just have to get synonyms for each word, in our case using some resources such as: wordnet, dictionaries and so on. It is important to highlight that wordnet and dictionaries contain a formal language and those have a lack of slang or informal language used in tweets. Therefore, we decided to use word embeddings to get the most similar words, Additionally, we defined a threshold[7] to control that the synonym was so close to the original word. In our case, the threshold has been defined as *0.7* and the results for the tweet mentioned before are shown in Table 5.

Some remarkable facts that we saw in our experiments are: if we keep a threshold over 0.9 for sure we would not be able to get synonyms because with a high threshold we are just recovering almost the same words used on the original sentence. On the other hand, if we keep a threshold under 0.5 we will get synonyms or similar words that are not coherent and additionally those new words change the meaning of the whole sentence. For instance, in the syn3 with a threshold 0.3, the original word (us) was changed by (states), we assume that it refers to the country called the United States, that sometimes is written like *us*, this is just an example of how many of these words change the meaning of the sentence when is used a lower threshold. Unfortunately, this approach was not able to achieve good results. But we noticed that is valuable go deeper with this focus.

[6] https://www.dictionary.com

[7] most_similar function return the top n similar words and their respective scores. In our case we assume, if the score is close to 1 then more similar is the word.

Architecture	Spanish						English					
	Task1				Task2		Task1				Task2	
	P	R	A	F1-score	F1-score(avg)	EMR	P	R	A	F1-score	F1-score(avg)	EMR
Emb.+CNN1d	0.84	0.84	0.84	0.84	0.78	0.69	0.74	0.73	0.74	0.72	0.74	0.62
Emb.+CNN1d*4+dense*2	0.70	0.70	0.70	0.70	0.67	0.58	0.63	0.64	0.64	0.64	0.62	0.51
Emb.+LSTM	0.78	0.77	0.78	0.77	0.75	0.67	0.61	0.63	0.63	0.62	0.73	0.64
Emb.+LSTM_Peep	0.77	0.77	0.77	0.77	0.74	0.65	0.57	0.57	0.57	0.58	0.70	0.58
Emb.+BILSTM	0.79	0.79	0.79	0.79	0.75	0.66	0.70	0.70	0.71	0.70	0.71	0.59
Emb.+BILSTM_GRU	0.81	0.81	0.81	0.81	0.75	0.66	0.73	0.73	0.73	0.74	0.72	0.63

Table 3: Results achieved using several NN architectures for English and Spanish development datasets.

	Task1			
	Spanish		English	
	Acc.	**F1-score**	**Acc.**	**F1-score**
glove+100	-	-	0.72	0.719
glove+200	-	-	0.74	0.745
glove+300	0.842	0.84	0.739	0.73
google+300	-	-	0.733	0.728
fasttext+300	0.777	0.78	0.732	0.75

Table 4: Results achieved by applying word embeddings to architecture A over development dataset.

Threshold	**Original Sentence**
	hurray saving us in so many ways lock them up build the wall end daca boycott nfl boycott nike
Threshold	**Similar Sentence**
0.9	syn1: 'hurray saving us in so many ways lock them up build the wall end daca boycott nfl boycott nike'
0.9	syn2: 'hurray saving us in so many ways lock them up build the wall end daca boycott nfl boycott nike'
0.9	syn3: 'hurray saving us in so many ways lock them up build the wall end daca boycott nfl boycott nike'
0.7	syn1: 'hurray save us in but several ways locks themselves up construct of walls end daca boycotting redskins boycotting adidas'
0.7	syn2: 'hurray saved us in anyway numerous ways lock them up rebuild the wall end daca boycotts usfl boycotts nike'
0.7	syn3: 'hurray saving us in even countless ways lock them up build the wall end daca boycotted packers boycotted nike'
0.5	syn1: 'hurra save u the but several way locks themselves ups construct of walls beginning daca boycotting redskins boycotting adidas'
0.5	syn2: 'hurrah saved us/uk , anyway numerous things locking they up/ rebuild which ramparts ends daca boycotts usfl boycotts reebok'
0.5	syn3: 'hooray rescuing states where even countless possibilities padlocks those messed constructing and doorway ending daca boycotted packers boycotted sportswear'
0.3	syn1: 'hurra save u the but several way locks themselves ups construct of walls beginning să boycotting redskins boycotting adidas'
0.3	syn2: 'hurrah saved us/uk , anyway numerous things locking they up/ rebuild which ramparts ends nica boycotts usfl boycotts reebok'
0.3	syn3: 'hooray rescuing states where even countless possibilities padlocks those messed constructing and doorway ending bucuresti boycotted packers boycotted sportswear'

Table 5: Similar words that were gotten using word embeddings.

Official Ranking

Table 6 shows the official results published by the organizers of HatEval. We submitted 10 runs for each task, the runs that achieved good results were run01 and run07 for English and run05 and run10 for Spanish. As we see too the performance of our the four runs or systems achieved high values over their respective baselines. Run01 and Run05 use TF-IDF scheme, run01 uses just structure features, meanwhile, run05 uses pos tagging and embedded emotions. Features that we have already described in section 2.1. Both runs achie-

ved good results using Support Vector Machine. The system used in run07 is the *architecture A* which was already described in section 2.2. Run10 used an embedding layer as input, followed by an LSTMPeephole layer and finally a dense layer of 2 neurons with softmax is used. The last two runs that use neural networks were fed with a corpus described in section 2.2. As we notice both approaches perform enough well, but architectures that use neural networks work slightly better than focuses that use classical machine learning algorithms.

	Task1		**Task2**	
	English			
	Acc.	**F1-score(avg)**	**EMR**	**F1-score(avg)**
SVC off. Base.	0.492	0.451	0.308	0.578
vmna (run01)	0.512	0.472	0.341	0.577
vmna (run07)	0.515	0.481	0.353	0.569
	Spanish			
SVC off. Base.	0.705	0.701	0.605	0.736
vmna (run05)	0.721	0.718	0.621	0.739
vmna (run10)	0.735	0.729	0.632	0.747

Table 6: Official results for task1 and task2.

Having the golden labels we realize that our proposed system misclassified several tweets. As we can see in the next examples:

1. * *(original tweet) Estas navidades mi polla mereces*
   * *(Engish version) you deserve my dick for Christmas*
   Language: Spanish
   Golden Label: no-hate (0)
   our system(run05): hate (1)

2. * *(original tweet) @shakhepri69 @KuriMelon21 @HoruSenpai @crafter657 oye puta basura, cállate*
   * *(Engish version) @shakhepri69 @KuriMelon21 @HoruSenpai @crafter657 shut the fuck up, bitch*
   Language: Spanish
   Golden Label: no-hate (0)
   our system(run05): hate (1)

3. * *(original tweet) @GMA @TVMarci His own fault #SENDTHEMBACK*
   * *(Spanish version) @GMA @TVMarci su culpa #SENDTHEMBACK*
   Language: English
   Golden Label: no-hate (0)
   our system(run01): hate (1)

All three examples have been labeled as hate by our classifier. On the first example, after preprocessing, the system just kept the words navidades (Christmas), polla (dick) and mereces (deserve). And looking for these words in our lexicons just the last two words were found, so the system marked the tweet as hate-content. The same situation happened with example 2 and 3, just the words puta (bitch), basura (trash), callate (shut up) and culpa (fault), send (enviar), back (regreso) were found. As (Zhang and Luo, 2018) noticed, that some times hate and no-hate tweets have no enough features to be differentiated. Doing a manual analysis in the datasets we notice that fact to in several tweets.

3. Conclusions

This work proposed two main approaches. The first one, take into account some features that can be considered to feed classical machine learning algorithms, some of these features are: structure, embedded emotions, pos tagging, and skip-grams. On the other hand, the second approach consists of designing several neural networks architectures to test a variety of corpora and word representations. Moreover, we face an unbalanced dataset using a technique called data augmentation. To get similar words were used pre-trained word embeddings with a tuned few thresholds. Using data augmentation our results did not improve but we noticed that is a promising field to study. Furthermore, looking at the results in Table 6 we appreciate that both approaches contribute to achieving good results, perhaps a deep study in both approaches can help us to understand and improve our results. As a future work, it is interesting to explore more deep learning or neural networks and design more complex architectures.

References

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Rangel, Paolo Rosso, and Manuela Sanguinetti. 2019. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of the 13th International Workshop on Se-*

mantic Evaluation (SemEval-2019). Association for Computational Linguistics.

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

Uwe Bretschneider, Thomas Wöhner, and Ralf Peters. 2014. Detecting online harassment in social networks. In *ICIS*.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language.

Paula Fortuna. 2017. FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO Automatic detection of hate speech in text: an overview of the topic and dataset annotation with hierarchical classes. Technical report.

Yoav Goldberg. 2015. A primer on neural network models for natural language processing. *CoRR*, abs/1510.00726.

Marti A. Hearst. 1998. Support vector machines. *IEEE Intelligent Systems*, 13(4):18–28.

Alon Jacovi, Oren Sar Shalom, and Yoav Goldberg. 2018. Understanding convolutional neural networks for text classification. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 56–65. Association for Computational Linguistics.

Ashraf M. Kibriya, Eibe Frank, Bernhard Pfahringer, and Geoffrey Holmes. 2004. Multinomial naive bayes for text categorization revisited. In *Proceedings of the 17th Australian Joint Conference on Advances in Artificial Intelligence*, AI'04, pages 488–499, Berlin, Heidelberg. Springer-Verlag.

Yuzhen Lu and Fathi M. Salem. 2017. Simplified gating in long short-term memory (LSTM) recurrent neural networks. *CoRR*, abs/1701.03441.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Julian Risch and Ralf Krestel. 2018. Aggression identification using deep learning and data augmentation.

Anna Schmidt and Michael Wiegand. 2017. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 1–10. Association for Computational Linguistics.

Leandro Silva, Mainack Mondal, Denzil Correa, Fabricio Benevenuto, and Ingmar Weber. 2016. Analyzing the Targets of Hate in Online Social Media.

Ziqi Zhang and Lei Luo. 2018. Hate Speech Detection: A Solved Problem? The Challenging Case of Long Tail on Twitter.