Linguistic generalization of slang used in Mexican tweets, applied in aggressiveness detection.

Sebastián Correa and Alberto Martin

[1] Universitat Politècnica de València, Valencia, Spain
{scorrea92, martintalberto}@gmail.com

Abstract. This paper summarizes the shared task aggressive detection in Twitter organized as part of the MEX-A3T workshop. The aim of this task is to determine whether a tweet is aggressive or not. In tasks of classifying small texts of social networks, as in many others, the use of bag of words can influence the performance of the models. In the Twitter context the formal and "slang" language are mixed increasing the number of synonyms and so the size of the bag of words. Generalizing some expressions as insults or laughs can reduce the size of the bag of words without losing the linguistic meaning or the writer intention, and provide a generalization to some unseeing words in the training set. Being that and immense bag of words for short texts is too disperse (Fang, et al. 2014), the use of reductions of components improves even more the performance of the models, as many words have a lack importance in the aggressiveness task, or appear too few times. In this paper we will develop a linguistic generalization for the common slang Mexican used in tweets to reduce the impact of the size in the bag of words. As well as a PCA implementation to improve the computational cost in the training process.

Keywords: Sentiment analysis, Machine learning, Tweet aggressiveness, Bag of words.

1 Introduction

Social media are growing fast in popularity and generating content, real and fake, also is being used by the users to express themselves, ideas, achievements or feelings. The last one can be redirected to persons, as they have positive or negative opinions to express. Attack and discredit people has become very common in this social media.

MEX-A3T organizes a task on the analysis of tweets from Mexican users. The aim is to advance the state of the art on the non-thematic analysis of short texts written in Mexican Spanish. The task is aggressiveness detection in tweets.

The data is given in two parts, training set and private test evaluation. The data has send in two text files, one with the content of each tweet per row, and the second one with the classification (1: aggressive 0: not aggressive).

2 Data treatment

Because of the high dimensionality in the feature space, the feature vectors are not suitable as input to the text classifier since the scalability will be poor (David 1989). In order to improve the scalability of the tweet classifier, for dimensionality reduction techniques, the Principal Component Analysis (PCA), are applied to reduce the feature space. The aim of the techniques is to minimize information loss while maximizing reduction in dimensionality (Lee 1999). Additionally, the proposed reduction seeks to reduce the dimensionality of the bag of words without removing or changing the meaning of the words to be reduced, the union of expressions with the same or similar meanings continues to maintain the content of the text but reduces the size of the input vector.

Unlike other written context (like papers, books, news, etc.), is pretty common to use slang, onomatopoeia or even wrong written words. So when the train data is vectorized, there are a lot of "words" that will rarely appear in any other tweet (like if someone write "jjajaajja", as all should appreciate it is a laugh, but wrong written and with a specific amount of "ja").

Furthermore, in languages there are usually many ways to express the same, in slang it is even higher. We will explained the different ways implemented to reduce these problems.

2.1 Generalization

Before tokenization, being that the problem consists in detect aggressiveness, it is relevant to focus on the different ways to express the same insult. Investigating the Mexican ways of expression, some words become more important into determining the aggressiveness in a tweet. (Lengua 2010)

For example: Hijo de puta (son of a bitch), can be expressed in a very different ways, like it is shown in Fig. 1 (but only for the male example, there will be exactly the same for the female).

Trying to generalize the insult itself, we have considered to replace all the appearances for another simplified way to express it hdp, what 's only one word, reducing the vectorization process.

Another typical way of insult for the Mexicans it is mother related (KePaXa 2017) (KennaBlue n.d.), so if the expression mentions other mothers, usually it is related in an aggressive way. But when it is about their own mother, it is never aggressive (generally speaking). There are two general ways to talk about someone else mother: su madre and tu madre, being that it is a very significant word, but with the opposite meaning if it is on its own, both got compacted in one-word structure: su_madre and tu_madre.

"hijo puta" "hijo de la gran puta" "chinga tu madre" "hijo de perra"
"hijo de la chingada" "hijo de tu pinche madre" "hijo de la rechingada"
"hijo de tu puta madre" "hijo de tu reputa madre" "hijo de tu reputisima madre"
"hijo de su pinche madre" "hijo de su puta madre" "hijo de su pinche madre"
"hijo de su reputa madre" "hijo de su reputisima madre"

Fig. 1. "Hijo de" expressions variations

The same has been done for pura mierda, alone these words could have different meanings, but together they normally have and aggressive one.

Furthermore, it is interesting to count the number of symbols ("!", "¡", "?", "¿") that appear in the sentences, many "!" express information about the writer intention, and as anyone can write different number of symbols, that would count as a different word for the vectorization, when it would be better to count the amount of them (to gather information about the writer intention). Replacing all these symbols with one extra space, makes the tokenization to consider each one as a different word, and the same word when it gets vectorized.

Now, after tokenization, as it was commented in the previous section, the laugh it is expressed in a lot of different ways, and the length of the words it is usually relevant, not the same saying "haha" or "hahahaha" for the meaning of the sentence.

For these reasons, once tokenized, for each word the amount of "j", "h" and vowels are counted and if the word only contains these letters, then it is processed. To do that it follows the next expression:

"j" + "h" + ("a"+"s" | "e" | "i" | "o" | "u") ≡ word length

The "s" has been considered due to the typo frequency shown in the data set, this happens because some users miss-click the "s" by the "a" letter in the keyboard.

It has been considered this way because there are examples like: "jahajaha", "jajsjaja", "jjajaja", "aaajaja", etc...

When it is processed we must be considered to make some exceptions, because words like "ha", "ah", "aaah", "haaa", "aaahh", are not really laughs, but will be considered like it for the previous generalization. To do this, if there is no "j" in the word (only "h"), the grade of change throw the letters is considered (for example "aah" will only have 1 change and "ahaha" will have 4), if there is only 1 grade of change, it is not considered as laugh.

Once it is taken as laugh, the word it is erased from the tokenization and n "ja"s are inserted instead. Where n is:

n=max ("j"+"h","a"+"s","e","i","o","u")

This way, the laugh it is completely generalized, only considering as one word for the bag of words, which counts the amount of "ja"s for sentence. It is also interesting for

future tweets, which can have another way to express laugh and would not be unknown for the models.

2.2 Data dispersion

As mentioned before, in tweets, like in natural language, there are many ways to express something, which increase the array needed when vectorizer the bag of words. But many of these words are rarely seen, creating an immense dispersed matrix (full of 0's).

Initially, the amount of different words were above 15,000, having less than 5,000 examples was difficult for the different models especially the Neural Network (NN), to train all the weights (it should have more data).

Having this in mind, it came to consideration to apply a PCA technique. The parameter for PCA of the amount of relevant data to keep was modified to try to reduce the data as much as possible without losing too much accuracy. The results of this are shown below.

Table 1. Result of PCA variation

PCA parameter	Input Vector Size	SVM F1-weighted
1.00	15894	0.7988
0.99	5357	0.7974
0.98	5003	0.7935
0.97	4722	0.7955
0.96	4482	0.7966
0.95	4268	0.7948

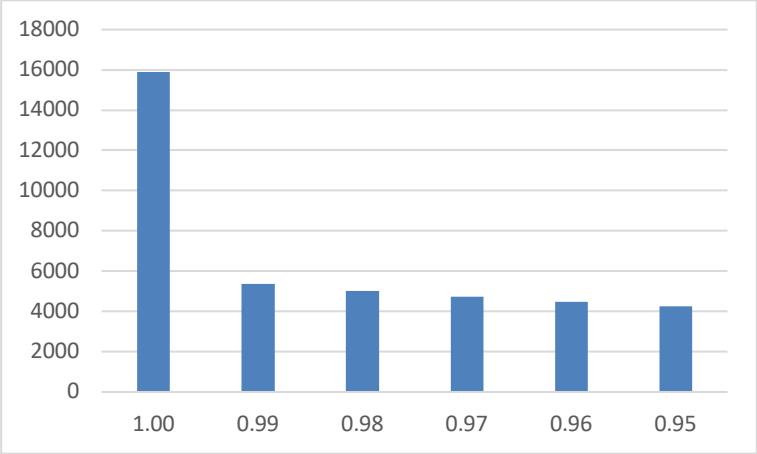

Fig. 2. PCA variation charts in input vector size

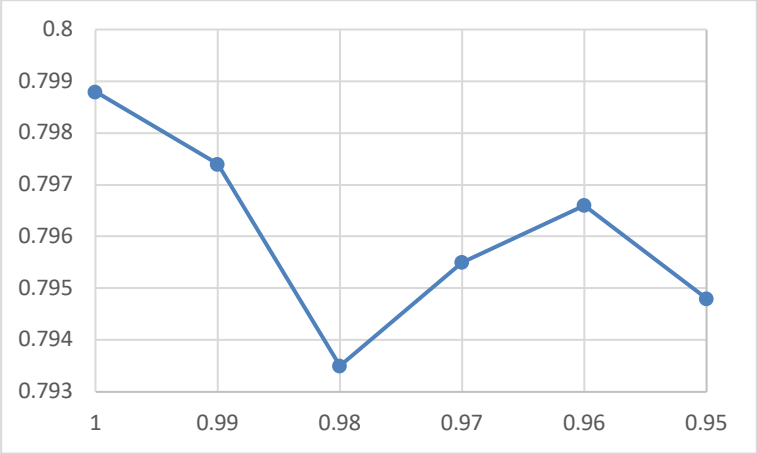

Fig. 3. PCA variation charts in SVM results

Therefore, for the final results a 0.99 data kept was choose, being that the reduction is really significant and does not reduce the SVM result too much. As said, the non-informative words in the vectorized matrix are so many that even keeping the 0.99 of the relevant eigenvalues, the data it is reduced to ⅓ of it is initial amount.

3 Methods and Results

For the task, we propose different classifications models to solve and analyse the problem. These models are SVM, NN, Decision Tree, Naives Bayes, and K-NN. They were tuned and tested with a validations test of the 20% of the original dataset. Accuracy, recall and F1-score have been used to evaluate each model.

The models used in these experiments were linear SVM, with C equal to one, penalty l2, loss squared hinge. Decision Tree Classifier, criterion gini, min samples split two, min samples leaf equal one. K Neighbors Classifier 100, weights uniform, leaf size 30, metric Minkowski. Neural Network, with Adam optimizer with batch size 32, 100 epochs, learn rate 0.001, step decay, loss function of macro fm.

Below are the results of all the proposed models for the cleaned dataset and the dataset with the complete bag of words.

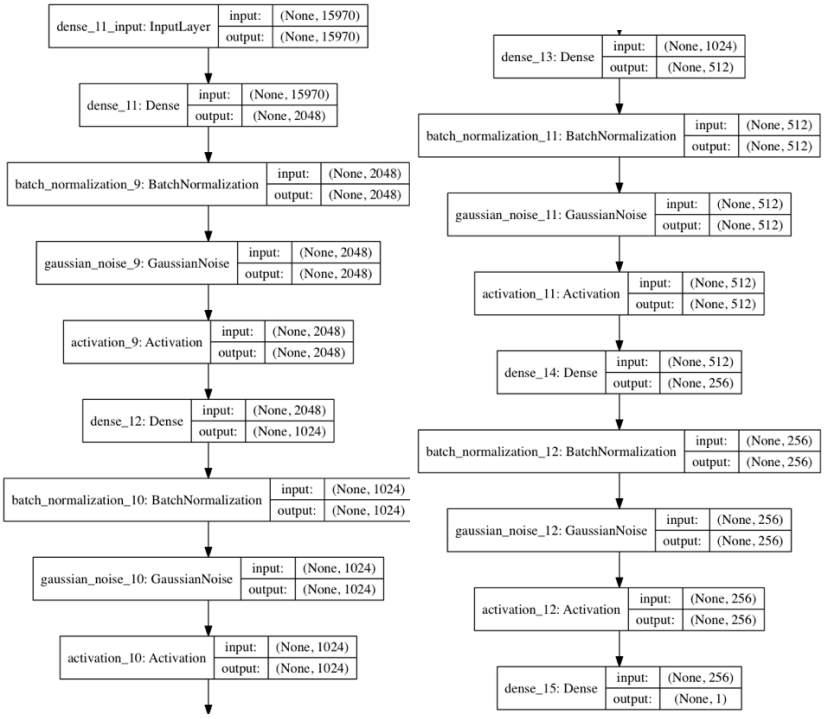

Fig. 4. Neural network architecture

Table 2. Results for not cleaned data

Model	Accuracy	Precision	Recall	F1-Score Micro	F1-Score Macro	F1-Score weighted
SVM	0.8026	0.7758	0.6360	0.8026	0.7760	0.7975
Neural Networks	0.8071	0.7780	0.6504	**0.8071**	0.7822	0.8027
Decision Tree Classifier	0.7467	0.6577	0.6198	0.7467	0.7217	0.7450
Gaussian Naive Bayes	0.5701	0.4357	0.6540	0.5701	0.5659	0.57786
KNeighbors	0.7760	0.8302	0.4756	0.7760	0.7242	0.7576

Table 3. Results for cleaned data

Model	Accuracy	Precision	Recall	F1-Score Micro	F1-Score Macro	F1-Score weighted
SVM	0.8026	0.7770	0.6342	**0.8026**	0.7754	0.7969
Neural Networks	0.7909	0.7995	0.5604	0.7909	0.7541	0.7806
Decision Tree Classifier	0.6721	0.5506	0.4901	0.6721	0.6349	0.6674

Gaussian Naive Bayes	0.7513	0.4084	0.5182	0.5182	0.5179	0.5147
KNeighbors	0.7565	0.8435	0.3982	0.7565	0.6876	0.7286

As it can be seen SVM an NN had similar results in all the scores. But when the data is cleaned the NN has a reduction of the F1-Score (used in the MEX-A3T contest) that's why we used SVM for final results. As it is seen in the **Table 3** the precision of the NN has increases in comparation with not cleaned data, it means that the NN detects more aggressiveness in the tweets but it get more false positive, as seen in the recall decrement.

We compare the best two models, SVM and NN, using the F1 metric for each class (aggressive and not aggressive). Without cleaning the data as we have described we get an initial results for the SVM of 70% for aggressiveness class and 85% for not aggressive, for the NN 70% and 84% in the respective order. Once the data is clean and applied PCA the results deteriorate a little bit, being for the SVM the results are 70% and 85% and for the NN 67% and 83%. But as we have described before the decreasement of the computer computational cost compensate the loss of the F1 score.

The results for the contest are expressed in the next table:

Table 4. MEX-A3T contest results

Team	**Acc**	**F**	**P**	**R**	**Positive**	**Negative**
INGEOTEC_task_aggressiveness_run_1	0,6673	0,6209	0,6226	0,6578	0,4883	0,7535
CGP_Team_Aggressiveness2	0,667	0,6056	0,6035	0,6273	0,45	0,7612
GeoInt-b4msa-MEXA3T	0,6876	0,6091	0,6049	0,6188	0,434	0,7842
aragon-lopez_aggressiveness	0,7117	0,6191	0,6176	0,6207	0,4312	0,8069
Trigrams	0,6888	0,6082	0,6042	0,6167	0,4304	0,786
nochebuena-aggressiveness-run2	0,6644	0,5955	0,593	0,6115	0,4285	0,7625
MXAA_aggressiveness	0,7136	0,6148	0,6153	0,6143	0,4198	0,8098
Our proposal	***0,6946***	***0,5988***	***0,5971***	***0,6008***	***0,4027***	***0,7948***
nochebuena-aggressiveness-run1	0,6917	0,5968	0,5949	0,5993	0,4012	0,7924
BoW	0,6771	0,5764	0,5751	0,5781	0,3698	0,783
simsom_aggressiveness_track_run_2	0,6702	0,5585	0,5585	0,5585	0,3365	0,7805
simsom_aggressiveness_track_run_1	0,5865	0,5094	0,5149	0,5183	0,315	0,7039
sergioCoraza	0,5963	0,5168	0,5392	0,4962	0,3123	0,7212
CGP_Team_Aggressiveness_1	0,7649	0,5837	0,6784	0,5797	0,3091	0,8583

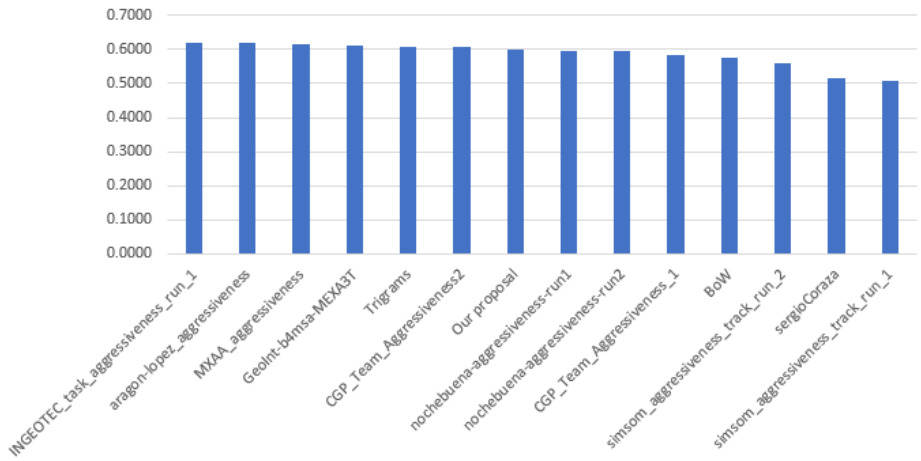

Fig. 5. MEX-A3T contest results chart F1 metric performance.

As it can be seen in **Table 4**, the model developed here, even that didn't get the best results, it was in the average in all the columns, generally nearer to the upper result.

Considering that the F1 metric was the one to compare in this competition, this result only it's extracted to compare the performance for all the users, as it's shown in the **Fig. 5**.

As it's seen in the **Fig. 5**, almost all the users got a similar performance, with an average of 0.5867 and a medium of 0.5978.

The model presented in this paper improves this two statistics, being 0.0221 away from the better result and 0.0894 of the worst.

4 Conclusions

Seeing these results, it is reasonable to establish that the generalization developed in this approximation worked. Also, the model does not sacrifice certain metrics to obtain better results in others, such as precision or recall.

For some models the quantity of inputs is more critical than for others, and the ordering and cleaning proposed in this project implies an improvement of calculation time (as in the NN) as well as precision in its metrics (such as Naive Bayes). In other cases, such as the SVM or the neural network there is no significant improvement or deterioration, but we can suppose that the use of these conventions could improve a later result, either in generalization or in optimization.

If this task was replicated in a larger scenario (100 thousand or millions of samples) the bag of words would grow exponentially, which would imply a much higher computational cost. The standardization of expressions would help in a significant way to reduce the bag of words without losing their meaning and importance in the text, which is vital in short texts of social networks where there are many cases of typo, spelling mistakes and use of unique words or expressions.

Reference

Álvarez- Carmona, Miguel Á and Guzmán-Falcón, Estefanía and Montes-y-Gómez, Manuel and Escalante, Hugo Jair and Villaseñor-Pineda, Luis and Reyes-Meza, Verónica and Rico-Sulayes, Antonio. 2018. «Overview of MEX-A3T at IberEval 2018: Authorship and aggressiveness analysis in Mexican Spanish tweets.» *Notebook Papers of 3rd SEPLN Workshop on Evaluation of Human Language Technologies for Iberian Languages (IBEREVAL), Seville, Spain, September.*

David, E. B. Baum and H. 1989. «What size net gives valid gener- alization?» *Neural Computation.*

Fang Wang, Zhongyuan Wang, Zhoujun Li, and Ji-Rong Wen. 2014. "Concept-based Short Text Classification and Ranking." *CIKM '14.*

KennaBlue. n.d. *Wattpad.* Accessed may 2018. https://www.wattpad.com/146314658-conociendo-méxico-groser%C3%ADas-mexicanas.

KePaXa. 2017. *Steemit.* August. Accessed May 2018. https://steemit.com/spanish/@kepaxa/groserias-a-la-mexicana-diccionario-basico-para-el-forastero.

Lee, Savio L. Y. Lam & Dik Lun. 1999. «Feature Reduction for Neural Network Based Text Categorization.» *Database Systems for Advanced Applications, 1999. Proceedings., 6th International Conference on.*

Lengua, Academia Mexicana de la. 2010. *Academia.* Accessed May 2018. https://www.academia.org.mx/obras/obras-de-consulta-en-linea/diccionario-de-mexicanismos.