An Evolutionary Approach to Build User Representations for Profiling of Bots and Humans in Twitter

Notebook for PAN at CLEF 2019

Roberto López-Santillán[1], Luis Carlos González-Gurrola[1], Manuel Montes-y-Gómez[2], Graciela Ramírez-Alonso[1], and Olanda Prieto-Ordaz[1]

[1] Facultad de Ingeniería, Universidad Autónoma de Chihuahua, Circuito No. 1, Nuevo Campus Universitario, Apdo. postal 1552, Chihuahua, Chih., México. C.P. 31240
{jrlopez,lcgonzalez,galonso,oordaz}@uach.mx

[2] Departamento de Ciencias Computacionales, Instituto Nacional de Astrofísica, Óptica y Electrónica, Tonantzintla, Puebla, México
mmontesg@inaoep.mx

Abstract In this work, we describe a novel weighting scheme of terms to produce document representations of Twitter posts for the Author Profiling task at PAN 2019. The purpose of this task is to predict the type and gender of the author, *bot/human*, and *bot/female/male*, respectively. The novelty of our approach resides in that we use an evolutionary approach to successfully combine traditional statistics features, e.g. *tf*, *idf*, generating, then an improved weighted scheme. Results suggest that our proposal outperforms the baseline, which uses the mean of the word embeddings as the weight, for up to 6% in accuracy when predicting author type in the English language.

1 Introduction

Bots are autonomous programs that aim to pose as humans on online platforms. They can be employed in a variety of applications that range from *Customer Service* activities to attempting to influence mass opinion. Take for instance the 2016 U.S. presidential election, where political organizations influenced voters through the use of these software agents [4]. The problem of correctly identifying *bots* out of human beings has been addressed following a variety of approaches that include computing some statistics of its behavior summarized for example on the *"friend to followers ratio"*, *"friend count"* and *"follower count"* [13]. However, given its constant improvement in its principles, more sophisticated approaches are needed with the capacity to analyze even the discourse that they create.

Author Profiling (AP) consist in predicting characteristics of authors based on information that they create or share. Based on the characteristic of bots it comes naturally

then, to approach its recognition as an AP Task. The PAN at CLEF [3], a yearly competition for AP, is proposing for 2019 to identify bots in two prediction problems: type of writer (human or bot) and gender (bot, female or male), based on anonymized Twitter posts [11,10].

For this competition we addressed the problem using Word Embedding (WEs) and Genetic Programming (GP), the latter to calculate a weighting-scheme so the former could be aggregated to produce Document Embeddings (DEs). The DEs were later deducted with the first component, obtained with Principal Component Analysis (PCA), to boost the accuracy of prediction. Combining WEs to produce DEs has been attempted before, using aggregate functions such as *sum*, *mean* and *median* [1]. Nonetheless we propose custom aggregate formulas evolved through GP, which to the best of our knowledge have been used for the AP problem only in our more comprehensive work [5], and in the present report.

In the present document we report the results attained by our approach in the *training* and *competition* stages at PAN 2019 [9]. Next we demonstrate our methodology, the results and conclusions.

2 Methodology

Our proposed approach to solve the *bot* / *human* distinction, depicted in Figure 1, uses an ensemble of novel ML and Natural Language Processing (NLP) algorithms. The competitive results attained in other datasets by this technique, rises the question of whether this method could deliver a practical outcome in the PAN 2019 AP task. We have already tested a variant of this methodology in the AP datasets from PAN 2013-2018, achieving promising results as demonstrated in the work of López-Santillán et. al in [5]. Next we elaborate on the main steps of this approach.

2.1 Word Embeddings

The first step in our technique is to construct a vocabulary of Word Embeddings (WEs) from the training dataset. To do so we used the *Skip-gram* variant of *word2vec* (w2v) [6] to produce our own WEs set. We also tested several WEs algorithms such as GloVe [8] and fastText [2], but preliminary results showed a better performance of *w2v* in our task. Furthermore, we tried pre-trained WEs from large datasets like Wikipedia, available for the English and Spanish languages. Nonetheless, WEs trained with the PAN 2019 dataset delivered better accuracy.

2.2 Computing Statistical Features of Dataset

Simultaneously with the previous step, we calculate several statistics from the training data. These variables represent the importance of terms within the dataset from different perspectives. In Table 1, we present the list of features that we considered (below an explanation of what each variable captures). At the end of this stage there will be seven different dictionaries, all consisting of the words in the vocabulary of the dataset and their respective values according to each formula.

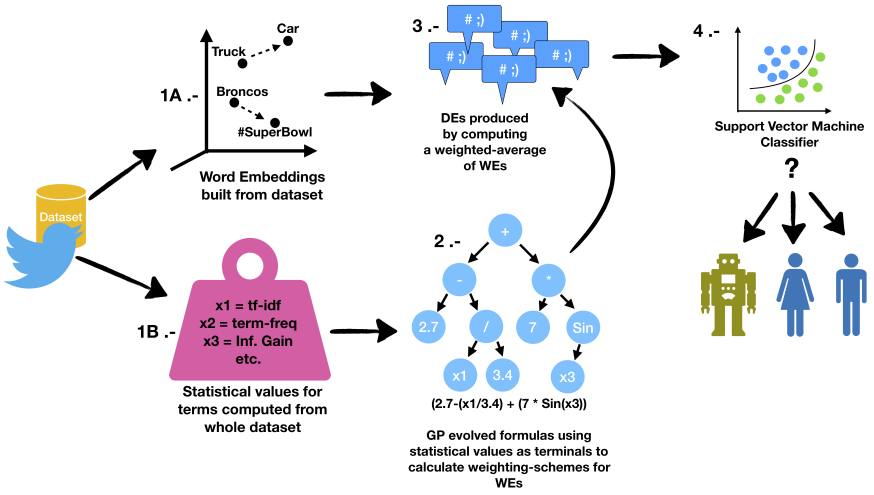

Figure 1: The method is an ensemble of Word Embeddings (WEs) and Genetic Programming (GP) to produce Document Embeddings (DEs) to identify bots from humans, and females from males.

Table 1: Statistical features computed over the dataset.

Variables (used to evolve)	Mathematical formula
X0	$\text{tf}_{t,d} = tf(t,d)$
X1	$\text{idf}_{t,d} = (1 + \log \text{tf}_{t,d}) \cdot \log \frac{N}{\text{df}_t}$
X2	$\text{tf}_{t,len(d)} = \frac{tf(t,d)}{\sum_{i=1}^{D} len(d(i))}$
X3	$\text{wdist}_{w_0,w_{last}} = w_{last} - w_0$
X4	$\text{ddist}_{d_0,d_{last}} = d_{last} - d_0$
X5	$\text{iGain}_{tfidf} = \sum_{x,y} P(x,y) \ln \frac{P(x,y)}{P(x)P(y)}$
X6	$\text{tf-idf}_{t,d} = tf(t,d) * (1 + \log \text{tf}_{t,d}) \cdot \log \frac{N}{\text{df}_t}$
X7	$\text{W} = \frac{F}{H}$

- $X0$ is the term frequency by user, t is a word from the vocabulary and d is a set of documents (*tweets*) by user.
- $X1$ is the idf value, which represents the importance of the term across the dataset, N is the total number of documents (*tweets*) in the dataset and df_t is the number of

documents where the term appears, thus assigning words with values accordingly to how often appear in documents.

- $X2$ is the term frequency by user, divided by the summation of the length of all documents (*tweets*) where the term appears, where D is the number of documents containing the term.
- $X3$ computes the distance (number of words) between the 1st. and last appearance of a term by user, where w_0 is the initial position and w_{last} the last position.
- $X4$ is the distance (number of documents) between the first and last appearance of a term by user, where d_0 is the initial document (*tweet*) and d_{last} is the last document where a term appears.
- $X5$ establishes the mutual information (dependence) between x (features from a *tf-idf* sparse matrix) and y (the target), where $P(x, y)$ is the probability of x given y, $P(x)$ is the probability of the vector of features and $P(y)$ the probability of the target.
- $X6$ is the product of the term frequency in a document and its idf value. This means that a term that appears frequently in just a few documents (*tweets*) will have a greater tf-idf value than a term that appears very often in many documents.
- $X7$ determines the importance of terms according to the most likely topic a document (*tweet*) belongs to. Topics are extracted from the dataset using the Non-Negative Matrix Factorization method, where F is a tf-idf sparse matrix of features, W is an array of the shape ($topics$, $terms\_in\_vocabulary$) which contains values for all terms according to the topic, and H is a non negative matrix that multiplied by W results in the original features (F).

We designed the last variable ($X7$) particularly for AP tasks. As already mentioned, a more extensive work has been done in a larger number of datasets, where this feature has shown promising results in the AP problem, as detailed in López-Santillán et. al [5].

2.3 Composing Document Embeddings: An Evolutionary Approach

Once the WEs and the statistical values for all terms in the training data vocabulary are computed, we need a strategy to represent all the *tweets* from each user. The representation of larger chunks of text is a matter of current interest in the NLP community. Our approach to compose Document Embeddings (DEs), that comprise all posts from a user into a single vector, is an aggregation of all WEs from terms in the tweets, using a *Weighted Average Scheme* (WAS) of them. To calculate the weights of terms for the WAS, we opted for the *evolutionary* algorithm Genetic Programming (GP), which uses a symbolic regression approach to evolve mathematical equations to approximate a target, by minimizing the error in each passing generation of new individuals (equations). GP codes the math equations in form of tree graphs, where the terminal nodes are variables and constants. We implemented the GP phase using a library called *GPLearn* in the Python language [12]. Table 2 shows the parameters employed by GP.

A different equation was devised for each combination of *language* / *target*. For instance, equation 1 is the evolved formula to aggregate WEs into DEs to predict *gender (bot / female / male)* in the *Spanish* language.

Table 2: Parameters of the Genetic Programming Process.

Parameter	Value
'Population size'	1000
'Generations'	100
'Function set'	'add', 'sub', 'mul', 'div', 'sqrt', 'log', 'sin', 'cos', 'tan', 'neg', 'max', 'min', 'abs'
'Metric'	mean absolute error
'Tournament size'	20
'Constant range'	-10-10
'Tree depths range'	2-6
'Init. method'	ramped half and half
'Crossover prob.'	0.9
'Mutation prob.'	0.1

$$GPWeight[w_n] = sub(sqrt(min(neg(cos(log(X2_{w_n}))), add(log(X2_{w_n}), X4_{w_n}))), neg(X5_{w_n})) \quad (1)$$

Where $X2_{w_n}$ is the term frequency in documents containing the word; $X4_{w_n}$ is the term dispersion in # of documents (number of Twitter posts between the first and last appearance of a term) and $X5_{w_n}$ is the information gain value of such term to predict the *gender*.

Finally we produce DEs for all Twitter posts of each user, by computing a weighted-average of the WEs of all terms in such posts, using the according formula to calculate the term weights. Equation 2 shows how we integrate the DEs for each user in the dataset. It is worth noting that each statistical value (as detailed in section 2.2), was tested collectively as well as individually, yet the evolutionary approach to combine such statistics, performed better every time.

$$\text{DEs-GP-Fmla} = \frac{\sum_{n=1}^{i}(WE_n * GPWeight[w_n])}{\sum_{n=1}^{i} GPWeight[w_n]} \quad (2)$$

Where WE_n is the Word Embedding of the n term in the tweets of each user, and $GPWeight[w_n]$ is the GP computed weight value of n term in the posts.

2.4 Boosting the Accuracy of DEs

Inspired by the work of of Arora et. al in [1], we used Principal Component Analysis (PCA) to reduce the dimensionality of the DEs, but only its first component was employed, then we multiply it by its transpose and the DEs themselves. Finally we deduct such product from the original DEs vectors. Arora et. al. proposed this step as they found a boost up to 10% in accuracy in textual similarity tasks. Equation 3 shows this final processing of the DEs.

Table 3: Distribution of users in the provided dataset.

Language	# users (training)	# users (validation)
English	2880	1240
Spanish	2080	920

Table 4: Parameters of the SVM classifier.

Classifier	Target	Parameter	Value
Support Vector Machine	Type	'C'	10
		'degree'	1
		'gamma'	1.0
		'kernel'	'rbf'
		'coef0'	0.0
	Gender	'C'	1
		'degree'	1
		'gamma'	1.0
		'kernel'	'rbf'
		'coef0'	0.0

$$\text{DEs} = DEs - (DEs * firstComp(DEs) * firstComp(DEs)^T) \quad (3)$$

Although we did not achieve the 10% boost in accuracy attained by Arora et. al. results in the training phase showed a small but clear increase in accuracy, probably because the AP problem is more difficult.

2.5 Classification

The dataset provided included data in the *English* and *Spanish* languages. Also the entries were divided into training and validation partitions. Table 3 details the distribution of samples in the dataset, note that there is a 70% to 30% proportion for the training and validation partitions respectively. Moreover each user contains 100 Twitter posts.

We approach the classification stage using a *Support Vector Machine* (SVM) algorithm implemented in the Scikit-learn library for Python [7]. We trained the SVM classifier using the proper partition, then we performed a *Grid-search* of hyper-parameters using the *validation* partition. Table 4 details the best parameters found to predict each target. The SVM classifier received the *training* and *validation* samples in the form of the DEs previously built, as detailed in section 2.3.

3 Results

Since the dataset was already provided with training and validation partitions, we evaluated our method using 70% of all data to train and 30% to test. For comparison reasons we also implemented a common baseline method, which consisted in composing DEs using the *mean* of the WEs. Table 5 shows the accuracy results attained in the training stage, by both the *evolutionary weighted-scheme* and the *baseline* (simple *mean*), to predict the *type* of user (*bot* or *human*) and *gender* (*bot*, *female* and *male*). Moreover,

for the competition phase we trained our SVM classifier with the whole available data (*training* + *validation*). Table 6 demonstrates the performance of our approach in the actual competition test data.

Table 5: Accuracy in *training* dataset predicting type (*bot*, *human*) and *gender* (*bot*, *female* and *male*).

Method	**Type**	**Gender**
Evol.-weighted-avg.		
English	0.9105	0.7903
Spanish	0.9130	0.7120
Average	**0.9118**	**0.7511**
Baseline-mean		
English	0.8589	0.7742
Spanish	0.9087	0.7207
Average	**0.8838**	**0.7474**

Table 6: Accuracy in *competition-test* dataset predicting type (*bot*, *human*) and *gender* (*bot*, *female* and *male*).

Method	**Type**	**Gender**
Evol.-weighted-avg.		
English	0.8867	0.7773
Spanish	0.8544	0.7100
Average	**0.8706**	**0.7437**

4 Conclusion

As can be noticed in the *results* section, our approach outperforms an accepted popular baseline. Additionally a decrease in accuracy was expected for the *competition* stage, as seen in Table 6. We already tested a similar version of the approach explained in this paper on several datasets (AP task in PAN 2013-2018), and we attained competitive results, a similar performance is expected for the PAN 2019 dataset. Although we can not guarantee the same performance or even to score in the top-quartile of the participants, we believe our approach can be implemented as an engineering application that could deliver good results in the real world.

5 Acknowledgments

This work was supported by CONACYT project FC-2410.

References

1. Arora, S., Liang, Y., Ma, T.: A simple but tough-to-beat baseline for sentence embeddings (2017)
2. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics 5, 135–146 (2017)

3. Daelemans, W., Kestemont, M., Manjavancas, E., Potthast, M., Rangel, F., Rosso, P., Specht, G., Stamatatos, E., Stein, B., Tschuggnall, M., Wiegmann, M., Zangerle, E.: Overview of PAN 2019: Author Profiling, Celebrity Profiling, Cross-domain Authorship Attribution and Style Change Detection. In: Crestani, F., Braschler, M., Savoy, J., Rauber, A., Müller, H., Losada, D., Heinatz, G., Cappellato, L., Ferro, N. (eds.) Proceedings of the Tenth International Conference of the CLEF Association (CLEF 2019). Springer (Sep 2019)
4. Howard, P.N., Woolley, S., Calo, R.: Algorithms, bots, and political communication in the us 2016 election: The challenge of automated political communication for election law and administration. Journal of Information Technology & Politics 15(2), 81–93 (2018), https://doi.org/10.1080/19331681.2018.1448735
5. López-Santillán, R., Montes-Y-Gómez, M., González-Gurrola, L.C., Ramírez-Alonso, G., Prieto-Ordaz, O.: A genetic programming strategy to produce document embeddings for author profiling tasks. Under Review (2019)
6. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 26, pp. 3111–3119. Curran Associates, Inc. (2013), http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf
7. 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., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830 (2011)
8. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: In EMNLP (2014)
9. Potthast, M., Gollub, T., Wiegmann, M., Stein, B.: TIRA Integrated Research Architecture. In: Ferro, N., Peters, C. (eds.) Information Retrieval Evaluation in a Changing World - Lessons Learned from 20 Years of CLEF. Springer (2019)
10. Rangel, F., Franco-Salvador, M., Rosso, P.: A low dimensionality representation for language variety identification. In: Proceedings of the 17th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing 2016). Springer-Verlag (2016)
11. Rangel, F., Rosso, P.: Overview of the 7th Author Profiling Task at PAN 2019: Bots and Gender Profiling. In: Cappellato L., Ferro N., M.H.L.D. (ed.) CLEF 2019 Labs and Workshops, Notebook Papers. CEUR Workshop Proceedings. CEUR-WS.org. CEUR Workshop Proceedings, CLEF and CEUR-WS.org (Sep 2019)
12. Stephens, T.: gplearn documentation pp. 1–55 (Apr 2019), https://buildmedia.readthedocs.org/media/pdf/gplearn/stable/gplearn.pdf
13. Van Der Walt, E., Eloff, J.: Using machine learning to detect fake identities: Bots vs humans. IEEE Access 6, 6540–6549 (2018)