Bot and gender recognition on tweets using feature count deviations
Notebook for PAN at CLEF 2019

Hans van Halteren

Centre for Language Studies, Radboud University Nijmegen
P.O. Box 9103, NL-6500HD Nijmegen, The Netherlands
hvh@let.ru.nl

Abstract This paper describes the system with which I participated in the Author Profiling task at PAN2019, which entailed first profiling Twitter authors into bots and humans, and then the humans into females and males. The system checked to which degree feature counts for a test sample were compatible with the corresponding feature count ranges in the training data. Two features sets were used, one with surface features (token unigrams and character n-grams with n from 1 to 5), the second one with overall measurements (e.g. percentage of retweets, type-token ratio and variation in tweet length). On the training set, recognition quality was extremely high, but much lower on the test set, indicating that some type of overtraining must have taken place.

1 Introduction

The Author Profiling task in PAN2019 was the differentiation between bots and humans, and subsequently for humans the differentiation between female and male authors, for samples of 100 tweets in English or in Spanish. A detailed description of the task is given by Rangel and Rosso[11].[1] As early experiments showed a severe risk of overtraining, the organisers provided splits into training and development sets, where the development sets were said to be similar to the eventual test sets. The provided tweets were not preprocessed. My approach for this task[2] built on earlier work. First of all, there was the long term work on authorship and other text classification tasks, which used to be published under the name Linguistic Profiling, which because

[1] In this paper, I will focus on my own approach. I refer the reader to the overview paper and the other papers on the PAN2019 profiling task for related work. Not only will this prevent overlap between the various papers, but most of the other papers, and hence information on the current state of the art, are not available at the time of writing of this paper.

[2] I also participated in the Author Attribution task. The differences in handling the two tasks were such that I preferred to describe the other task in a separate paper [5]. However, there will obviously be some overlap.

of ambiguity of that term has now been replaced by the working title “Feature Deviation Rating Learning System” (henceforth *Federales*). Although the full name implies a specific learning technique, the acronym indicates a combination approach. Which form of combination was used in this task is described below (Section 3). Furthermore, I reused specific previous work related to the current task. [7] addressed, among other things, recognition of Twitter bots (for Dutch tweets) by noticing that their artificial language use leads to overall measurements (e.g. type-token ratio) different from that of the more variable language use of human authors. [4] addressed gender recognition on, again, Dutch tweets, concluding that counts of all words are the best performing features: with these we measure the authors’ (described) life rather than their language use. Although the two studies differed in language (Dutch versus English/Spanish), sample size (full production over several years versus 100 tweets) and time period (when the majority of Twitter users were still reporting on themselves versus when the majority was slowly moving towards business users), I kickstarted the current experiment from the basics of the earlier work.

2 Feature Extraction

The most important choice in any classification task is the selection of features for the learning components. For this task, I mostly wanted to investigate the potential of features relating to regular, botlike, language use. In support I included more standard features, but kept these simple, by taking only character n-ngrams and token unigrams.

2.1 Tokenization

As some of the features were to be based on tokens, I tokenized all text samples, using a specialized tokenizer for tweets, as used before for [7]. Apart from normal tokens like words, numbers and dates, it is also able to recognize a wide variety of emoticons.[3] The tokenizer is able to identify hashtags and Twitter user names to the extent that these conform to the conventions used in Twitter, i.e. the hash (#) resp. at (@) sign are followed by a series of letters, digits and underscores. URLs and email addresses are not completely covered. The tokenizer counts on clear markers for these, e.g. http, www or one of a number of domain names for URLs. Assuming that any sequence including periods is likely to be a URL proves unwise, given that spacing between normal words is often irregular. And actually checking the existence of a proposed URL was infeasible as I expected the test machine[9] to be shielded from internet. Finally, as the use of capitalization and diacritics is quite haphazard in tweets, the tokenizer strips all words of diacritics and transforms them to lower case.[4]

[3] The importance of this has dropped seriously after the introduction of emojis.

[4] The system has worked suboptimally here, as the check for out-of-vocabulary words was implemented incorrectly, comparing the normalized word forms from the samples with the unnormalized word forms in the word list. For English, the difference was probably negligible, but for Spanish, with all its diacritics, the OOV counts were greatly exaggerated, as we will see in Section 2.3.

2.2 Surface Frequency Features

Although I generally prefer to use a wide range of features, including syntactic ones (cf. the notebook on the attribution task [5]), tweets do not lend themselves well for many such features. I therefore decided on more local patterns, also since token unigrams performed best in the experiments in [7]. Given the "informal" nature of tweets, I complemented unigrams with character n-grams (with n from 1 to 5). Token unigrams were built with the normalized tokens, whereas character n-grams were built on the basis of the original tweet. Both types of features were counted separately in original tweet and retweets. In order to be included in the feature set, a feature needed to be observed in at least five different authors (in the full training data). This led to about 1.23M features for English and 0.94M features for Spanish. However, in any specific classification, only those features present in training or test texts were used.

2.3 Overall Measurement Features

In addition to the straightforward frequency counts, I followed the strategy described in [4]. The idea is that bots show more regular behaviour than humans, as they are driven by algorithms. Such regularities should lead to relatively extreme behaviour, such as low or high numbers of URLs or out-of-vocabulary words. Other examples might be low type-token ratio because a limited vocabulary is used, high type-token ratio because the tweets are less mutually related, or low standard deviations for tweet length. In total I took 71 measurements, including totals over all tweets, means and coefficients of variation for specific measurements per tweet, and some richness measures. Specific examples (those actually used in the experiments) are shown in Tables 1 and 2.[5] [6]

Now many of these measurements are mutually correlated. In the initial phases of my work on the task, I handpicked[7] subsets for English and Spanish that together yielded the best classification by themselves. In later phases it turned out that the standard Federales models performed particularly well (on the training data), especially after splitting the data into clusters. There was no time to return to the measurement features, so in principle this part of the system can still be improved. This will have to wait for future work. Information for the features that have been used in the current experiments is listed in Tables 1 and 2. The counts for low and high values here are based on a threshold of 2 for a z-score with regard to the mean and standard deviation for human authors, as listed in the tables.[8]

[5] Words were called out-of-vocabulary if they did not occur in the word lists I had available. For Spanish I used the file espanol.txt as provided at http://www.gwicks.net/dictionaries.htm. Unfortunately, the words in the list were not normalized as the words from the text were, which led to the rather high OOV measurements of 34% and 52%. However, as both human and bot authors are mismeasured in the same way, the results still hold information. For English I used a wordlist derived from the British National Corpus combined with a wordlist which on double checking the software turned out (major embarrassment) to be the wrong one, namely one for Dutch. Obviously, both lists can be improved upon.

[6] IDF for English is based on the British National Corpus. For Spanish I did not have access to an IDF list.

[7] This procedure can be automated, but I did not do this at this time.

[8] In the actual recognition, all values over 0.7 are taken into account.

Table 1. Overall measurements for English.

Measurement	Human	Human	Low	High
	mean	sdev	bot%	bot%
Original tweets (vs retweets)	0.64	0.27	3.8	
Properties original tweets				
@-mentions	0.057	0.042		0.7
URLs	0.029	0.024		21.6
OOV tokens	0.15	0.066	8.7	12.2
Upper case characters	0.20	0.017	9.9	21.4
Average IDF	2.64	0.77	5.4	14.5
Type-token ratio	0.43	0.091	29.3	1.8
Variation length in chars	0.52	0.16	38.6	0.5
Variation length in tokens	0.61	0.18	34.8	0.7
Variation @-mentions	1.72	1.09		7.1
Variation hashtags	3.09	2.43		6.1
Properties retweets				
@-mentions	0.066	0.022	84.7	0.9
URLs	0.023	0.013		7.0
Average IDF	2.64	0.53	0.5	5.3

3 Learning Techniques

As stated above, I used two types of features. These were processed in different ways.

3.1 Frequency Vector Comparison

The Federales system builds on the Linguistic Profiling system, which has been used in various studies, such as authorship recognition [1][2], language proficiency[6], source language recognition[3], and gender recognition[7]. The approach is based on the assumption that (relative) counts for each specific feature typically move within a specific range for a class of texts and that deviations from this typical behavior indicate that the deviating text does not belong to the class in question. If the frequency range for a feature is very large, the design of the scoring mechanism ensures that the system mostly ignores that feature. For each feature, the relative counts[9] for all samples in the class are used to calculate a mean and a standard deviation.[10] The deviation of the feature count for a specific test sample is simply the z-score with respect to this mean and standard deviation, and is viewed as a penalty value. Hyperparameters enable the user to set a threshold below which deviations are not taken into account (the *smoothing threshold*), a power to apply to the z-score in order to give more or less weight to larger or smaller

[9] I.e. the absolute count divided by the corresponding number of items, e.g. count of a token in a retweet divided by all tokens within retweets, or a character n-gram count divided by the number of characters in the text.

[10] Theoretically, this is questionable, as most counts will not be distributed normally, but the system appears quite robust against this theoretical objection.

Table 2. Overall measurements for Spanish.

Measurement	Human	Human	Low	High
	mean	sdev	bot%	bit%
Original tweets (vs retweets)	0.59	0.29	4.2	
Properties original tweets				
URLs	0.031	0.034		8.9
OOV tokens	0.34	0.058	8.7	11.9
Type-token ratio	0.45	0.074	34.8	1.9
Variation length in chars	0.54	0.16	54.6	0.2
Variation length in tokens	0.64	0.21	30.0	0.4
Variation @-mentions	2.11	1.44		6.1
Variation URLs	2.41	1.54		5.7
Variation hashtags	2.99	2.57		3.9
Variation OOV tokens	0.52	0.11	42.5	1.9
Variation amount punctuation	0.91	0.46		5.2
Properties retweets				
URLs	0.021	0.013		6.1

deviations (*deviation power*), and a *penalty ceiling* to limit the impact of extreme deviations. When comparing two classes, a further hyperparameter sets a power value for the difference between the two distributions (*difference power*), the result of which is then multiplied with the deviation value. The optimal behaviour in cases where a feature is seen in the training texts for the class but not in the test sample, or vice versa, is still under consideration. In the current task, features only seen in the test sample are ignored; features only seen in the training texts are counted as they are, namely with a count of 0 in the test sample. The penalties for all features are added. A set of benchmark texts is used to calculate a mean and standard deviation for the penalty totals, to allow comparison between different models. For verification, the z-score for the penalty total is an outcome by itself; for comparison between two models, the difference of the z-scores can be taken; for attribution within larger candidate sets (such as the clusters described in Section 5), the z-scores can be compared. In all cases, a threshold can be chosen for the final decision.

3.2 Extreme Language Use (XLU) Determination

The features for the overall measurements could have been mixed into the general feature set, but there their influence would be minimal seeing the enormous number of surface features. Instead I processed these measurements in a different way, for now dubbed an XLU score (eXtreme Language Use). After investigation of the values on the training data, I set a consideration threshold of 0.7 on the z-score. Any feature having a z-score with regard to the mean and standard deviation for human authors higher than 0.7 or lower than -0.7 scores the excess is counted as XLU points. The XLU score for the sample is simply the sum of the XLU points over all selected features. My expectation was that bots should be recognizable by their high XLU score. As explained above (Section 2.3) feature selection was done early in the work on the task and may not be optimal. This is also true for the threshold of 0.7. However, the remaining hyperparameter, a threshold above which an author is predicted to be a bot, has been chosen in the final phases, separately for each cluster (see below in Section 5).

4 Training Procedure

From the initially provided data sets, I held out 400 English authors (200 bots, 100 females, 100 males) and 300 Spanish authors (150 bots, 75 females, 75 males), all randomly selected from the whole data set, as development material. Later during the task, the organisers provided their own train-development split, with 620/310/310 and 460/230/230 samples held out. With both divisions, I trained XLU and Federales models on the training sets (just the human samples for the gender tests) and classified the development sets. Using the gold standard annotations, I selected optimal hyperparameter settings and tested the system. The results were surprising. Where accuracies were over 90% for my original held-out test set, they were much lower (for Spanish gender under 70%) for the organizer train-development split. Although overtraining is likely to play a role (as perfect and near perfect scores are inherently suspicious), the differences between the two train-development splits are much larger than one would expect from simple overtraining effects. Not having information about the exact composition of the train and development sets provided by the organisers, one of my hypotheses[11] was that the text in the development set was somehow different from that in the training set. A visual check did not immediately show clear differences. However, when I trained a classifier for distinguishing between training and development set, most settings led to an accuracy over 90% and some even over 95%. Unfortunately, inspection of the most distinguishing unigrams still did not provide a clear picture of the exact difference between the sets. Seeing that no information was available on the nature of the sets, and obviously also not on the composition of the eventual test set, I had to adapt to potential unknown differences. The first step in this was a simplification. The different sets led to very different optimal hyperparameter settings. I therefore decided to drop hyperparameter tuning and select the simplest hyperparameters: no smoothing threshold, a penalty ceiling of 40, no power applied to deviation and to model distance difference. I would have preferred to avoid score thresholds for Federales score as well, which would ideally be at 0. However, it turned out that on the training and development data the optimal thresholds were not 0. As a result, I picked the various thresholds by hand. In addition, thresholds were also needed for XLU scores, which have no natural threshold. These too I picked by hand. An author was predicted to be a bot if either score exceeded its threshold.[12] The second adaptation was all but a simplification. As different subsets of the data proved to lead to different outcomes, it seemed a good idea to split the authors into (data-driven) subsets. Any new author could then first be assigned to a subset, after which the models for that subset would be applied. Given the size of the whole data sets, I intuitively decided on seven subsets of authors. How these were derived is described in the next section.

[11] The others including a bug in my systems.

[12] Unfortunatly, the testing phase showed that this “tuned” thresholding again led to overtraining.

5 Clustering

For both languages, I build frequency lists of normalized original tokens[13] in the full data set, i.e. training plus development. I then examined the top of the list to select around 1000 most frequent tokens. For English this led to a list of 1162 tokens occurring in at least 300 authors and for Spanish 1294 tokens occurring in at least 150 authors. I then built frequency vectors for each sample and used k-means clustering to produce seven clusters.[14] The resulting clusters had sizes 189, 448, 270, 138, 418, 277 and 320 for English, and 357, 268, 141, 145, 66, 225 and 298 for Spanish. For these seven clusters per language, I ran Federales classification (with the hyperparameters mentioned above) using the full dataset for both training and testing. Only samples for which the score for their own cluster was higher than 0.5, and for which the second highest scoring cluster was more than 10% behind, were used as prototype samples for the cluster, i.e. used as training samples in the final cluster classification. For English, clusters 1 and 4 kept all their samples, cluster 2 lost 46 (3 assigned to other cluster, 43 unassigned), cluster 3 lost 8 (0, 8), cluster 5 lost 84 (18, 66), cluster 6 lost 17 (6, 11) and cluster 7 lost 20 (0, 20). For Spanish, cluster 1 lost 124 (30 assigned to other clusters, 94 unassigned), cluster 2 lost 27 (3, 24), cluster 3 lost 4 (1, 3), cluster 4 lost 4 (2, 2), cluster 5 lost none, cluster 6 lost 32 (5, 27) and cluster 7 lost 47 (11, 36). In the final classification, the threshold for acceptance was lowered to 0, but the minimum distance to the runner up was kept to 10%. Tables 3 and 4 show the final attribution to clusters for the training and development sets. For some clusters we see that there are indeed differences between training and development set, e.g. English cluster 5 where predominance of females/males switches from training to development set, or Spanish cluster 4 where the training set still has 18 males on a total of 117 authors but the development set no males at all versus 55 females. We also see that clustering already goes quite far in distinguishing between bots and humans. Females and males on the other hand are well present in all clusters.[15]

In the prediction phase, each test sample was submitted to an attribution choice between a cluster and the set of all human train samples, for each of the seven clusters. The models for the cluster with the strongest attribution score were applies to that sample.[16] Models were also created from the samples in the training data that were not assigned to any sample, to be used for unattributed test samples.

6 Training Results

Application of the finally submitted system on the training and development set led to the confusion tables shown in Tables 5 and 6. The corresponding accuracies are also

[13] In the full feature set, these are the features marked CTO.

[14] I used the function stats::kmeans in R,[10] with the Hartigan-Wong method[8], a maximum of 40 iterations, 20 restarts and obviously a target of 7 centers.

[15] It would be very interesting to investigate how the clusters differ from each other. I have postponed this investigation until the test data and hopefully metadata have become available.

[16] I contemplated a combination of all clusters accepting the sample, but I deemed this too complicated for the current experiments.

Table 3. Cluster composition of training sets for English.

	Train			Dev		
	bot	female	male	bot	female	male
Cluster 1	25	80	75	20	33	32
Cluster 2	6	117	180	0	37	69
Cluster 3	0	58	130	0	22	72
Cluster 4	1	80	13	0	46	17
Cluster 5	0	107	132	0	81	20
Cluster 6	4	106	97	12	55	39
Cluster 7	33	148	69	12	31	57
Not in cluster	1371	24	24	576	5	4
Total	1440	720	720	620	310	310

Table 4. Cluster composition of training sets for Spanish.

	Train			Dev		
	bot	female	male	bot	female	male
Cluster 1	2	77	120	4	23	41
Cluster 2	6	49	122	5	24	63
Cluster 3	29	53	55	4	40	21
Cluster 4	8	99	18	0	55	0
Cluster 5	255	35	46	89	21	12
Cluster 6	5	117	33	3	35	34
Cluster 7	1	73	109	10	29	47
Not in cluster	734	17	17	345	3	12
Total	1040	520	520	460	230	230

shown in these Tables. For this material, it appeared to be possible to separate bots from humans and females from males with an extremely high level of accuracy. If the test samples were sufficiently similar to the training data, they too should be classified with high accuracy.[17]

I would like to point out that, theoretically, the high accuracy was not a natural consequence of testing on the training data. Both XLU and Federales are greedy methods based on means and z-scores over all samples. It was not as if the classifier could recognize an individual sample and reproduce its class. The feature values for each sample were embedded in distributions for all samples in the class training set, which tended to be tens or hundreds of samples.

On the other hand, the accuracy was too high to be believable. The earlier discrepancy between tests on a small random held-out set and on the organizer-provided train-test split also was reason for doubt. Unfortunately, if indeed there was some regu-

[17] The original version of this paper was written before the test results were available. For the revised version, the test scores were known, but not the test data or metadata. I have decided to leave this section in its original form, showing my reasoning before the test phase, and to insert a new section below, commenting on the test results.

Table 5. Confusion table for prediction per cluster on the training material for English.

Actual	bot			female			male			Accuracy
Predicted	bot	female	male	bot	female	male	bot	female	male	
Cluster 1	45				113				107	100
Cluster 2			6		154				249	98.5
Cluster 3					80				202	100
Cluster 4		1			126				30	99.4
Cluster 5					188				152	100
Cluster 6	16				161				136	100
Cluster 7	45				178		3	1	123	98.7
Not in cluster	1947			2	27		2		26	99.8
Total	2053	1	6	2	1027	1	5	0	1025	99.6

Table 6. Confusion table for prediction per cluster on the training material for Spanish.

Actual	bot			female			male			Accuracy
Predicted	bot	female	male	bot	female	male	bot	female	male	
Cluster 1	1		5		100				161	98.1
Cluster 2			11		73				185	95.9
Cluster 3	33				93				76	100
Cluster 4		8			154				18	99.6
Cluster 5	344		2		56				56	99.6
Cluster 6		8			152				67	96.5
Cluster 7			11		102				156	95.9
Not in cluster	1079			1	19				29	99.9
Total	1457	16	27	1	749	0	2	0	748	98.5

larity in the current training data which would not reoccur in the test data, and scores on the test data therefore would turn out to be much lower, more extensive metadata was needed to determine the nature of this regularity and from there the way to adapt the system to become robust against such overtraining.

A side-effect on the high accuracy of the Federales models was that the XLU scores hardly played a role anymore. As this was a major focus of my initial plan, I ran a separate prediction with only XLU, with new thresholds.[18] The results are found in Table 7.

In general, results were very good. For Spanish, the bot-dominated cluster 5 was problematic, as many humans also had high scores. For English, there were no bot-dominated clusters but cluster 7 had a slightly larger minority of bots and also yielded somewhat lower results; cluster 1 was similar but did not seem to be problematic. Again, these were the results on the training data, with optimal thresholds.

[18] The thresholds in the submitted run were tuned for optimal correction of mistakes by the Federales models.

Table 7. Results on the training data for scoring with XLU alone. A threshold of 0 means that all authors in the cluster were predicted to be bot, and 999 that all were predicted to be human.

	English		Spanish	
Cluster	Threshold	Accuracy	Threshold	Accuracy
Cluster 1	27	97.7	15	98.1
Cluster 2	999	98.5	999	95.9
Cluster 3	15	100	25	91.1
Cluster 4	30	99.4	30	95.6
Cluster 5	20	100	5	77.5
Cluster 6	15	95.2	20	96.5
Cluster 7	25	88.9	999	95.9
Not in cluster	0	97.2	0	95.7
Total		97.0		92.3

Given the promising results on the training data, I selected these models and thresholds for the system to upload to TIRA for a blind test[9], in order to see if the quality would hold up on the test data.

7 Test Results

As stated in the previous section, results on the training data were good, in fact suspiciously good.[19] Still, before the test run, the system appeared to be the best choice at that time. The test results showed, however, that it was not. With bot recognition scores of 89.6% (English) and 82.8% (Spanish),[20] the system was not even close to the best scores (96.0% for English and 93.3% for Spanish)[21] and worse than the serious baselines, based on character n-grams (93.6%/89.7%), word n-grams (93.6%/88.3%), word2vec (90.3%/84.4%) and LDSE[12] (90.5%/83.7%). The enormous gap between training scores and test scores demonstrates that some kind of overtraining must have occurred. The clustering, which improved scores on the training data, now probably only served to aggravate the overtraining.

The question, now, is what the cause of the overtraining was. Generally, in machine learning, overtraining is the result of insufficient similarity between training and test data. If the test authors had been drawn randomly from the same pool as the training authors, the system should in principle have done better. If, however, the test authors stem from another source, this would explain the rather disappointing quality in the test run. However, the author profiling task was not presented as a cross-genre task, like the author attribution task was, so this should not be the main cause. To determine which other factor(s) might still have been at work, I will have to investigate the test data and, possibly even more importantly, the metadata describing the sources and their sampling.

[19] This section was inserted into the paper after the test results were made available.

[20] As gender scores (English 74.2% and Spanish 67.3%) are partly based on the bot scores, I cannot judge at this time how well my gender recognition worked by itself.

[21] Not reached by the same system.

8 Conclusion

The data that was provided for training could be modeled very well with both feature sets that I applied, token unigrams and character n-grams on one side and overall measurements on the other. Classification quality was high when training on the set as a whole, and improved further after clustering the authors and applying separate thresholds for the various clusters.

However, the derived model performed disappointingly on the test data. The reasons for this can only be determined when the test data and metadata become available, and will therefore have to wait for future work. This future work will then also have to show the real potential of my proposed approaches.

References

1. van Halteren, H.: Linguistic Profiling for authorship recognition and verification. In: Proceedings ACL 2004. pp. 199–206 (2004)
2. van Halteren, H.: Author verification by Linguistic Profiling: An exploration of the parameter space. ACM Transactions on Speech and Language Processing (TSLP) **4**(1), 1 (2007)
3. van Halteren, H.: Source language markers in Europarl translations. In: Proceedings of COLING2008, 22nd International Conference on Computational Linguistics. pp. 937–944 (2008)
4. van Halteren, H.: Metadata induction on a Dutch Twitter corpus. initial phases. Computational Linguistics in the Netherlands Journal **5**, 37–48 (2015)
5. van Halteren, H.: Cross-domain authorship attribution with Federales, Notebook for PAN at CLEF2019. In: Cappellato, L., Ferro, N., Müller, H., Losada, D. (eds.) CLEF 2019 Labs and Workshops, Notebook Papers. CEUR Workshop Proceedings. CEUR-WS.org (2019)
6. van Halteren, H., Oostdijk, N.: Linguistic Profiling of texts for the purpose of language verification. In: Proceedings of the 20th international conference on Computational Linguistics. p. 966. Association for Computational Linguistics (2004)
7. van Halteren, H., Speerstra, N.: Gender recognition of Dutch tweets. Computational Linguistics in the Netherlands Journal **4**, 171–190 (2014)
8. Hartigan, J.A., Wong, M.A.: Algorithm AS 136: A k-means clustering algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics) **28**(1), 100–108 (1979)
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. R Development Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing (2008), http://www.R-project.org
11. Rangel, F., Rosso, P.: Overview of the 7th author profiling task at PAN 2019: Bots and gender profiling. In: Cappellato, L., Ferro, N., Müller, H., Losada, D. (eds.) CLEF 2019 Labs and Workshops, Notebook Papers. CEUR Workshop Proceedings. CEUR-WS.org (2019)
12. Rangel, F., Rosso, P., Franco, M.: A low dimensionality representation for language variety identification. In: Proceedings of the 17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLingŠ16),. pp. 156–169. Springer-Verlag, LNCS(9624) (2018)