Profiling Twitter Users Using Autogenerated Features Invariant to Data Distribution

Notebook for PAN at CLEF 2019

Tiziano Fagni and Maurizio Tesconi

National Research Council (CNR), Institute of Informatics and Telematics (IIT)
via G. Moruzzi 1, 56124, Pisa, Italy
`{tiziano.fagni, maurizio.tesconi}@iit.cnr.it`

Abstract With the diffusion of Web and Social Media, automatic user profiling classifiers applied to digital contents have become extremely important in application contexts related to social and forensic studies. In many research papers on this topic, an important part of the work is devoted to a costly manual "feature engineering" phase, where the semantic, syntactic, and often language-dependent features need to be accurately chosen to be relevant for profilation task. Differently from this approach, in this work we propose a Twitter user profiling classifier which exploits deep learning techniques to automatically generate user features being a) optimal for user profilation task, and b) able to fight covariance shift problem due to data distribution differences in training and test sets. In the best configuration found, the built system is able to achieve very interesting accuracy results on both English and Spanish languages, with an average final accuracy of more than 0.83.

1 Introduction

Since the 19th century, authorship identification (AI) analysis have been used in several contexts [13,28] in order to draw some conclusion about specific cases where the original author was uncertain. In the last years, with the advent of Web and Social Media, the attention of researchers has moved to the analysis of digital contents (e-mail, blogs, Twitter, etc.) in contexts related to social studies and forensic analysis. Author profiling, one the main sub-tasks of AI, is a method of analyzing a corpus of texts in order to identify the main stylistic and content-based features characterizing the user profiles (e.g., gender, age, etc.) we are interested in. Every year, PAN workshop proposes a specific challenge on this topic, offering multilingual annotated datasets where participants can test their own techniques. This year, "Bots and Gender Profiling" task is focused on the profilation of Twitter users considering both the user's type (bot vs. human) and, in case of human, the user's gender (male vs. female) [18]. Differently from multimodal data in author profiling task of PAN 2018 [19], this year we only have textual contents ready for analysis, making it similar to the edition of 2017 [20].

Previous editions have shown a prevalence of software solutions based on traditional machine learning approaches, based on costly manual feature engineering phases. Except for some works, the usage of language modeling and deep learning techniques for textual features encoding has been quite limited and the level of accuracy reachable by using only these techniques is unclear. In this work we thus propose a software solution strongly based on these recent approaches able to a) perform textual encoding in a language-agnostic way and with almost no manual feature engineering, and b) attenuate the data covariate shift [24] between training and test datasets in order to improve classifier accuracy.

The rest of this paper is organized as follow. In Section 2 we briefly introduce related works on this topic, and in Section 3 we show how we performed text encoding. After having introduced the covariate shift problem of challenge datasets in Section 4, we describe in details the design of the proposed solution in Section 5, the experimental results in Section 6, and the conclusions on Section 7.

2 Related Work

Until 2017, "Author profiling" task in PAN challenges was focused on age and gender identification [22,21], while in 2017 the age part was replaced by a sub-task focused on language variety identification [20]. In 2018, the main focus of the challenge was instead targeted to gender prediction but using different types of data sources: text and images. In the following, let's describe the types of software solutions proposed on author profiling task in the last 2 editions of the challenge.

Two main different types of feature representation have been used in past works. A first one is a traditional approach with features (content-based or style-based) selected through a manual "feature engineering" phase. Many works [9,2,8,3,26] use chars and word n-grams (or a combination of them) for basic feature representations, while others [15,6,2,14] also used stylistic features obtained by counting or computing ratios for stopwords, hashtags, user mentions, character flooding, emoticons, and laugher expressions. Another type of feature representation, similar in spirit to what we propose in this work, has been the usage of features obtained through neural networks. Some works [1,10,27] exploited word embeddings, chars embeddings, or a their combination with traditional features to encode textual contents. In this work, in addition to this standard approaches, we propose also a deep learning network able to learn the latent syntax encoding of a text (see Section 3.2).

The classification approaches used by many works [1,9,26,3] are based on traditional machine learning algorithms like SVM, Random Forest, logistic regression, Naive-Bayes, or ensemble of these methods. An increasing number of works [12,10,25] use instead deep learning approaches adopting mainly RNN (like LSTM, GRU) and CNN nets. Often these networks are combined together or mixed with traditional machine learning methods in order to get best of both worlds or differentiate the strategy used to analyze a specific type of content (e.g., texts and images). In this work we heavily exploit deep learning algorithms to generate automatically a set of optimal and distribution-invariant user features for problem's representation, while we use SVM over these computed features to build the final classifier.

3 Textual Encoding

A common approach to encode textual contents is to use a NLP (Natural Language Processing) processing pipeline [23]. This type of solution suffers from a costly "feature engineering" phase necessary to try and test several types of features in order to obtain sufficiently informative data representations for the problem to deal with. Furthermore, domain knowledge and the availability of high quality analysis tools in the target language are critical for successfully performing this step: unfortunately these conditions can not always be met. In order to overcome these limitations, in the following we propose 3 different approaches based on language modeling techniques and deep learning algorithms.

3.1 W2V: a Semantic Encoding Based on Word2Vec

The first type of text encoding used (called *W2V*) is based on the usage of a language modeling algorithm very popular in the last years: *word2vec* [11]. This technique allow to compute word vectors (called embeddings) by analyzing a set of unannotated texts according to an objective function based on distributional hypothesis[1]. The resulting vectors have very interesting properties matching the relationship existent between words (e.g., $W(\text{“}queen\text{”}) \cong W(\text{“}king\text{”}) - W(\text{“}man\text{”}) + W(\text{“}woman\text{”})$).

Given a raw tweet, in order to obtain a vector representation to be used as text encoding with machine learning methods, we proceed as following. We first convert the text to lowercase, next we replace each distinct URL with the word "url" and we remove all punctuation characters. If present, we remove also all the emoticons occurring in the text. The remaining textual content is then tokenized into distinct tokens (eventually appearing more than 1 time) and the final encoding vector is computed as

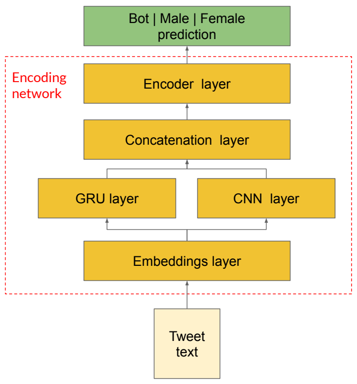
$$\boldsymbol{V_{tweet}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{we}(token(i)) \tag{1}$$

where $\boldsymbol{V_{tweet}}$ is the final tweet vector, N is the total number of tokens belonging to final tokenized tweet, $token(i)$ is the function returning the token at position i in the tokenized vector, and $\boldsymbol{we}(x)$ is the function returning the word vector associated to token x.
For both target languages, we used precomputed 300-dimensional embedding models built on big data collections: GoogleNews for English[2] and SBWCE for Spanish[3].

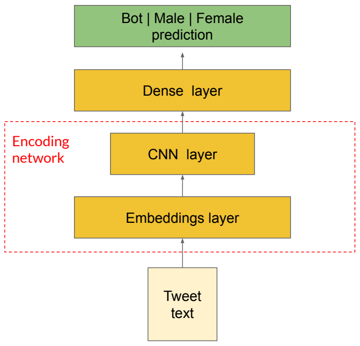
3.2 SYN: a Syntax-Modeling Encoding Based on Deep Learning

The W2V encoding described previously has the main drawback in trying to only capture semantic information from Twitter data, skipping all the remaining knowledge related to write-style and syntax used by the different type of users to write tweets. In

[1] The meaning of this hypothesis is that words appearing in similar contexts often have a similar meaning.
[2] Available at http://tiny.cc/0vhz6y
[3] Available at https://bit.ly/2Q9DBzU



(a) Neural network architecture for SYN encoder.



(b) Neural network architecture for CHARS encoder.

Figure 1: Deep learning architectures for SYN and CHARS tweet encoders.

order to extract syntax-related information from textual data, we proposed an encoding method based on deep learning technology, as shown on Figure 1a.

The original tweet text is tokenized before being fed to the encoding neural network. The tokenization process is performed in the following way. We replace all distinct URLs with the word "url" but we do not remove neither the punctuation nor the emoticons found on text. Each distinct user mention (e.g. `@realdonaldtrump`) and hashtag (e.g. `#politic`) is replaced respectively by the words `mention` and `hashtag`. Each remaining word is transformed into one of these 3 words: `uppercase_w` for a completely uppercase word, `lowercase_w` for a completely lowercase word, and `mixed_w` for a word including both uppercase and lowercase characters.

In order to learn a tweet encoder using the architecture proposed in Figure 1a, we used only training data split and transform the dataset of our original problem (proposed to classify users into profiles) into a new dataset suitable to classify tweets into user profiles. For each user profile in training data, we thus take its tweets and its original assigned label (bot, female, or male) and we extract 100 different tweets where each one has assigned the label of the user. At the end of this process we have produced 2 new training datasets, one for English and one for Spanish, consisting in respectively 288,000 and 208,000 training documents, which can next be used to learn the two profile classifiers. The proposed neural network uses two sub-networks to learn different feature types. A first part exploits a GRU net [5] to find interesting temporal patterns in the sequence of input tokens. A second sub-network based on a CNN [7] try instead to identify spatial features which are invariant to respect of original position in the text. These two types of found features are next concatenated together and used to find an optimal vector encoding for the tweet, before performing the final classification.

The syntax classifiers so obtained are not interesting in themselves, what it does matter for us is the possibility to take the trained nets and extract, as tweet encoding vector, the output produced by "Encoder layer" layer. The resulting vector should be in fact an effective representation of the syntax information convey by the original tweet.

3.3 CHARS: a Characters Encoding Based on Deep Learning

This encoding, differently from W2V method, want to exploit data at sub-word level in order to find interesting recurrent patterns and potentially useful information for user profile classification. The tokenization process of original tweets is simpler than the others two seen encoders. In this case, we have defined the set of valid symbols in the characters dictionary as $(A - Za - z0 - 9)$, the punctuation symbols, and all the distinct emoticons found on the training data split. The tokenization process thus simply consists in removing all invalid characters from input raw text. In the same manner as described for SYN encoder, starting from training data split, we create two new different training datasets to solve a *tweet* $\Rightarrow$ *user_profile* classification task. As shown on Figure 1b, the neural network architecture used to learn the encoder is slightly different from that used in SYN encoder. The network only use a CNN sub-network for encoding the tweet's data and the layer taken as final tweet encoder is the CNN layer.

4 Data Mismatch in Contest Dataset

The first attempt to build a user's profile classifier for the "Bots and Gender Profiling" task was to take all the data included in train and validation splits, merge them together, and then try to perform a k-fold evaluation [23] on the resulting dataset using one of the textual encoding defined in the previous section and a simple feed forward neural network classifier. With a similar attempt, we obtained a final software accuracy on profiling task of 0.973 for both English and Spanish classifiers. Unfortunately these results were a lot worse if we used the splits suggested by organizers, with a final classifier accuracy of 0.785 for English and just 0.721 for Spanish. From additional performed tests, we have verified that if just use only training data split, we have no evidence neither of bias nor variance [4] in the built classifier, the accuracy discrepancy is only evident when the classifier learned on training data is applied on test data of validation split. If we reasonably assume that provided validation split matches the data distribution of the unknown test dataset used to evaluate the system for the challenge, we can infer that there is a "covariance shift" problem [24] between the two distributions that need to be treated appropriately.

5 The Proposed User's Profile Classifier

For each language, we designed a single-label multiclass classifier [23] able to assign a user to exactly 1 of the 3 defined classes: *bot*, *male*, and *female*. The high level architecture of the software is shown on Figure 2. The tweets of a user timeline are encoded through a specific text encoder *TE* and then passed to a component called *User Encoder* with the aim to analyze the timeline and to produce a user vector including useful information for final profilation. The vector so obtained is later passed to an SVM classifier which will determine the label to assign to the user.

We tested three different types of textual encodings:

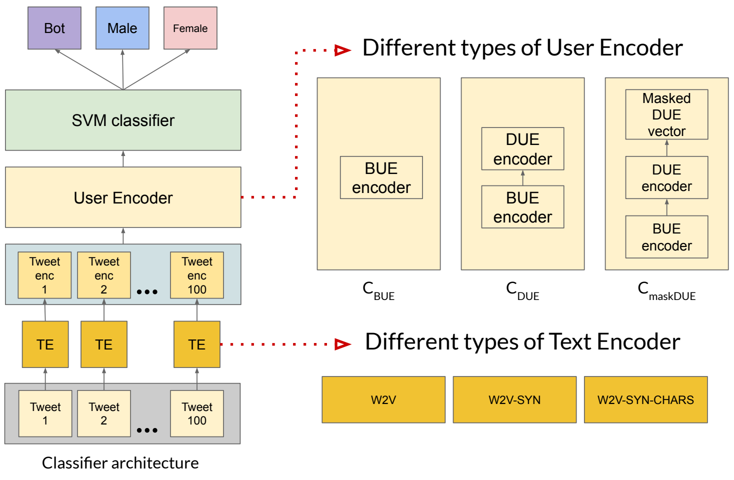


Figure 2: High level architecture of the proposed system.

W2V This encoding is the same encoding as defined in Section 3.1. The idea here is to exploit only semantic information from original data.

W2V-SYN This encoding is the concatenation of W2V encoding and the encoding generated by SYN encoder (described in Section 3.2). The idea in this case is to use a representation conveying both syntactic and semantic information from original data.

W2V-SYN-CHARS This encoding is the concatenation of W2V encoding, SYN encoding, and the encoding generated by CHARS encoder (described in Section 3.3). The idea here is the same as W2V-SYN representation but with the additional ability to learn also information at sub-string level, useful for handling particular cases such as orthographic errors.

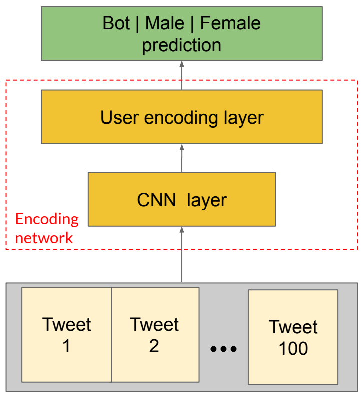
We used 3 different types of user encoder configurations, as described in the following:

C_{BUE} A configuration producing final user vector by only applying a BUE encoder (described in Section 5.1).

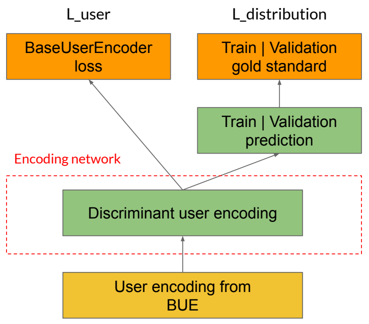
C_{DUE} A configuration producing final user vector by applying in sequence first a BUE encoder and then a DUE encoder (described in Section 5.2).

$C_{maskDUE}$ A configuration similar to C_{DUE} but with the addition of a final step used to mask the vector features most discriminative for data distributions (described in Section 5.3).

In the remainder on the section, we will describe the two proposed core user encoders (BUE and DUE encoders) used in tested configurations, the strategy used to mask most discriminating features, and how we built the final user's profile classifier.



(a) Base user encoder (BUE) architecture.


L_user
L_distribution
BaseUserEncoder loss
Train | Validation gold standard
Train | Validation prediction
Encoding network
Discriminant user encoding
User encoding from BUE

(b) Discriminant user encoder (DUE) architecture.

Figure 3: Deep learning architectures for SYN and CHARS tweet encoders.

5.1 Learning a Base Twitter User Encoder

In this section we describe a first type of user encoder (called BUE, Base User Encoder) which is able to process user timelines and provide good vector representations of the user profiles. In order to auto-learn very informative features for the final task, we designed a classification system based on a deep learning architecture (see Figure 3a) and trained it using only train data split. The final built classifier, thanks to CNN sub-network[4], learn how to encode properly the sequence of tweets submitted by the user by identifying the space-invariant inter-tweets patterns most relevant to solve the classification problem. In particular, the trained network can be used to compute a good user vector representation starting from its timeline. We tuned the network to have 512 as vector size for user encoding layer (layer "User encoding layer"), while for CNN we used 512 different filters with a window size set to 10.

5.2 Learning a Discriminant User Encoder

User encodings produced by BUE encoder are optimal user representations for training data split and final classification task, but as we have seen before are suboptimal for encoding and classify test datasets due to different characteristics of data. The main aim of this step is thus to start from BUE vector representations and try to find similar user representations, suitable for classification task but where the features high discriminative for training and test distributions are highlighted and easily recognizable. In order to tackle this issue, we propose a discriminative user encoder (called DUE, Discriminative User Encoder) based on the neural network architecture shown on Figure 3b.

[4] We tested other variants of deep learning architectures (e.g., using sub-nets based on LSTM, GRU, or a combination of them with CNN) but we finally choosed CNN because it was able to return back slightly better user vectors encoding (giving better effectiveness on final classifier).

This encoder is trained on a new dataset composed by both train and validation data but where each user has different target labels respect to original task: "True" if the user was originally included in validation split, "False" otherwise. The neural network structure is very simple. It takes as input a user vector representation computed by BUE encoder and then in the first hidden dense layer (Discriminant user encoding) try to learn a user representation optimal for this new binary classification task. To ensure that the net would learn a user representation suitable also for original "Bots and Gender Profiling" task, we introduce a constraint of what the net can learn at this level. In particular, we force the net to minimize two different losses in gradient descent optimization: the loss on labels related to data distribution ($L_{distribution}$), and the loss due to the difference from learned encoding and the encoding produced by BUE encoder (L_{user}). With the imposed constraints we aim to obtain a trained net which will produce user vectors of dimension 512 very similar to those produced by BUE encoder but containing some specific feature resulting very discriminative in recognizing the original data distribution of a user.

5.3 Masking most Discriminating Features

DUE encoder provides user vector representations containing some high informative features for identifying the origin of the data distribution of a user. A feature is high discriminative for data distribution if, after having built a binary classifier using only the considered feature, we are able to distinguish quite well the distribution origin of a user. To weight the discriminative power of a feature we can use ROC-AUC [23], a measure able to telling how much a model is able to distinguish between classes. To measure ROC values for all features, we built a new balanced dataset (D_{mask}) mixing some data from train split and some from validation split and measured classification performance of each feature using a Random Forest classifier.

On Figure 4 we show a typical distribution of the features related to their ROC-AUC values emerged from the tests. The majority of features have a ROC value around 0.55 but a long queue exists on the right with few features having high discriminative power (up to more than 0.8). These last type of features are the those we want to mask in the final user encoding vector. Indeed, without these features the user vector should be less sensible to distribution origin and therefore, in the successive learning phase, we force the classifier to concentrate on the other distribution-invariant features to solve the final task.

5.4 Building the Final Task Classifier

The final user's profile classifier has been choosed and tuned using remaining train and validation data not included in dataset D_{mask}. We tested two different classifiers (Random Forest and SVM) but we finally adopted SVM because it performed better in terms of accuracy in all tested configurations. The classifier has been optimized by fine-tuning model parameters through the use of a k-fold validation. For both languages and independently from the used text encoding, the best found configuration uses an RBF kernel and standard parameters "gamma" and "C" set to respectively 0.01 and 0.1 . The best threshold determining the ROC minimal value over which we apply

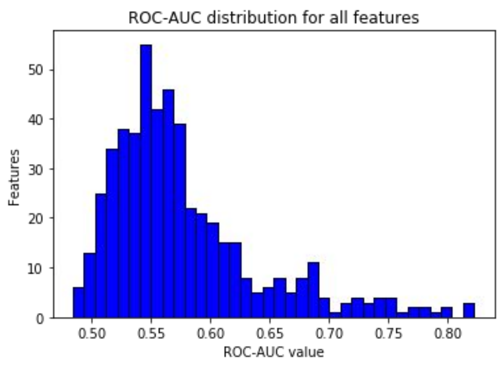


Figure 4: ROC-AUC values computed for all user features.

feature masking depends on the adopted text encoding and it will be reported in the experimental section.

After found the best model parameters and chosen a threshold for ROC value, we built the final optimized classifier using train split as training data and using validation split as test data to measure the accuracy of a specific solution. The final models submitted on early test dataset for evaluation have instead been built using all the data available (the union of data in training and validation splits). Detailed results of the tested configurations will be presented in the experimental section.

6 Experimental Results

In this section we will report all the results obtained in our experimentation. A detailed description of the dataset used in the challenge is available in [18], while the final evaluation on test sets has been obtained by using TIRA platform [16]. The base measure used to evaluate the goodness of the proposed software solutions is the accuracy [23]. The software has been implemented entirely in Python using various external libraries for specific functions. In particular, the deep learning part has been developed using Keras[5] with a Tensorflow[6] backend, while we used scikit-learn[7] for traditional machine learning methods.

For each language, on Table 1 we report the system global accuracy on validation dataset obtained a) using the three different strategies seen above for encoding tweets and b) using three ways to encode the user vector before building the final classifier. In particular, we report the 3 types of user encoder configurations introduced in Section 5, together with cut threshold for masking features ("Masking threshold" column). This first type of experiment has the main aim in identifying, for each language and each tweet encoding, the best user encoding which next will be used to test the system

[5] Available at https://keras.io/ .
[6] Available at https://www.tensorflow.org/ .
[7] Available at https://scikit-learn.org/stable/ .

Tweet encoding	**C_{BUE}**	**C_{DUE}**	**Masking threshold**	**$C_{maskDUE}$**
ENGLISH				
W2V	0.7960	**0.8032**	0.65	0.7998
W2V-SYN	0.8065	0.8225	0.65	**0.8282**
W2V-SYN-CHARS	0.8203	0.8169	0.70	**0.8266**
SPANISH				
W2V	0.6920	**0.6967**	0.65	0.6945
W2V-SYN	0.7670	0.7695	0.65	**0.7741**
W2V-SYN-CHARS	0.7789	0.7771	0.70	**0.7793**

Table 1: Classification global accuracy of the software on validation data split.

		VALIDATION SPLIT			EARLY TEST DATASET		
Tweet encoding	**User encoding**	**Type task**	**Gender task**	**Global**	**Type task**	**Gender task**	**Global**
ENGLISH							
W2V	C_{BUE}	0.8693	0.7370	0.8032	0.8902	0.7576	0.8239
W2V-SYN	$C_{maskDUE}$	**0.9080**	0.7483	**0.8282**	**0.9091**	**0.7955**	**0.8523**
W2V-SYN-CHARS	$C_{maskDUE}$	0.8967	**0.7564**	0.8266	0.8939	0.7424	0.8181
SPANISH							
W2V	C_{BUE}	0.8630	0.5304	0.6967	0.8778	0.6889	0.7833
W2V-SYN	$C_{maskDUE}$	**0.8863**	0.6619	0.7741	**0.8944**	**0.7556**	**0.8250**
W2V-SYN-CHARS	$C_{maskDUE}$	0.8826	**0.6760**	**0.7793**	0.8778	0.6722	0.7750

Table 2: Comparison of software accuracy between validation and early test datasets.

performance on early test dataset. For this reason, for each language and each tweet encoding tested, we also report in bold the best user encoding found.
On both languages, the results for W2V-SYN and W2V-SYN-CHARS configurations show that the better results are obtained by using $C_{maskDUE}$ configuration, while for W2V the best ones make use of C_{DUE} without using feature masking. The optimal threshold identified for the various configurations is very similar and is always comprised between 0.65 and 0.70. It is also interesting to note that the accuracy for W2V-SYN always increases while passing from C_{BUE} to $C_{maskDUE}$ configurations. W2V-SYN-CHARS instead, thanks probably to a richer representation of input data, seems instead to be a strong competitor also using the base C_{BUE} user encoding.

On Table 2, we report a comparison between the accuracy obtained for both languages on validation data and on the early test dataset provided by the challenge organizers. The reported results are referred to the best configurations identified on Table 1 and show the accuracy obtained on the 2 sub-tasks defined in the competition ("Type task" and "Gender task" columns) together with the averaged global accuracy of the system ("Global" column). For each type of dataset and each language, we have also reported in bold the best results.
Globally, the results show clearly that W2V-SYN is the best performer, in particular,

	"TYPE" TASK		"GENDER" TASK		
Method	**English**	**Spanish**	**English**	**Spanish**	**Global**
MAJORITY [18]	0.5000	0.5000	0.5000	0.5000	0.5000
RANDOM [18]	0.4905	0.4861	0.3716	0.3700	0.4295
CHAR N-GRAMS [18]	**0.9360**	0.8972	0.7920	0.7289	0.8385
WORD N-GRAMS [18]	0.9356	0.8833	**0.7989**	0.7244	0.8355
WORD EMBEDDINGS [18]	0.9030	0.8444	0.7879	0.7156	0.8127
LDSE [17]	0.9054	0.8372	0.7800	0.6900	0.8031
W2V-SYN with C_{maskDUE}	0.9148	**0.9144**	0.7670	**0.7589**	**0.8387**

Table 3: Final software accuracy on challenge test set using W2V-SYN and C_{maskDUE} configurations. On the table we report also the accuracy of the baselines proposed by challenge organizers.

on early test dataset and for both languages, it always provided the best accuracy. W2V-SYN-CHARS seems to be competitive only on validation dataset but on early test dataset the accuracy is not very good, especially on "Gender task". A possible explanation of this worst performance respect to W2V-SYN is that the addition of features generated by CHARS encoder confuse the classifier while learning the best parameters to build a classification model invariant to data distribution. W2V is instead the worst performer on both datasets, probably a sign that this type of text encoding has a limited expressivity in representing the data of the classification task.
If we observe the results from the point of view of languages and classification subtasks, we can note the the proposed configurations always perform better on English, with a remarkable difference in terms of accuracy between the two classification subtasks. In particular, solving the "Type" task seems a lot easier than solving the "Gender" task. This difference is probably due from one side to the greater number of training examples available for "Type task", from the other side for the greater difficulty[8] in identifying the differences in the write style between males and females.

Finally, on Table 3 we show the accuracy results obtained by our best configuration (W2V-SYN with C_{maskDUE} encoder) on the test dataset used for final evaluation of system performance in the challenge. In addition, on the table we report also the accuracy obtained by various baseline methods proposed by organizers. Our method outperforms all the baselines, in particular seems more effective especially on Spanish language. On both languages, the global accuracy results obtained by our algorithm on final test dataset are quite similar to those obtained on early test dataset (0.8409 and 0.8366 vs. 0.8523 and 0.8250). An interesting observation is that in final test dataset the difference in terms of accuracy between the two sub-tasks "Type" and "Gender" is very similar for both languages, suggesting that the proposed solution is a good balanced system and not sensible to the target language.

[8] If we manually inspect a timeline of a human user of the dataset, it is not always so easy identify its gender. Generally, a bot user is instead easily recognizable because certain types of writing patterns occurred frequently in the timeline.

7 Conclusions

In this work we tackled the "Bots and Gender Profiling" task of PAN 2019 challenge by proposing a software solution making extensive use of state of the art language modeling and deep learning techniques. In particular, these type of algorithms have been used in order to automatically performing feature encoding of textual contents, avoid almost completely the usual costly and manual "feature engineering" phase. In addition, deep learning has been also exploited to find textual features invariant to train and test datasets, helping fighting data covariate shift which usually has remarkable negative effect on final classifier accuracy. The experimental results show that, among all tested configurations, the solution using W2V-SYN textual encoding and combined with user encoder C_{maskDUE} is globally the best performer on both validation and early test datasets. In particular, on the final test set, such configuration allow to obtain a balanced multilingual classifier able to obtain very interesting accuracy results on both languages, with a global combined accuracy of 0.8387.

Possible future works could try to use more advanced language modeling techniques like BERT and Elmo for textual encoding. These methods provide contextual embeddings for text representation, by automatically disambiguate both the terms and the used syntax in analyzed contents. A comparison with the textual encoding methods proposed here could thus be very helpful in order to address future research directions.

References

1. Akhtyamova, L., Cardiff, J., Ignatov, A.: Twitter author profiling using word embeddings and logistic regression. In: CLEF (Working Notes) (2017)
2. Alrifai, K., Rebdawi, G., Ghneim, N.: Arabic tweeps gender and dialect prediction. In: CLEF (Working Notes) (2017)
3. von Daniken, P., Grubenmann, R., Cieliebak, M.: Word unigram weighing for author profiling at pan 2018. In: Proceedings of the Ninth International Conference of the CLEF Association (CLEF 2018) (2018)
4. Geman, S., Bienenstock, E., Doursat, R.: Neural networks and the bias/variance dilemma. Neural Comput. **4**(1), 1–58 (Jan 1992). https://doi.org/10.1162/neco.1992.4.1.1, http://dx.doi.org/10.1162/neco.1992.4.1.1
5. Jozefowicz, R., Zaremba, W., Sutskever, I.: An empirical exploration of recurrent network architectures. In: International Conference on Machine Learning. pp. 2342–2350 (2015)
6. Karlgren, J., Esposito, L., Gratton, C., Kanerva, P.: Authorship profiling without using topical information: Notebook for pan at clef 2018. In: 19th Working Notes of CLEF Conference and Labs of the Evaluation Forum, CLEF 2018, Avignon, France, 10 September 2018 through 14 September 2018. vol. 2125. CEUR-WS (2018)
7. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)
8. Markov, I., Gómez-Adorno, H., Sidorov, G.: Language-and subtask-dependent feature selection and classifier parameter tuning for author profiling. In: CLEF (Working Notes) (2017)
9. Martinc, M., Skrjanec, I., Zupan, K., Pollak, S.: Pan 2017: Author profiling-gender and language variety prediction. In: CLEF (Working Notes) (2017)

10. Martinc, M., Skrlj, B., Pollak, S.: Multilingual gender classification with multi-view deep learning. In: Proceedings of the Ninth International Conference of the CLEF Association (CLEF 2018) (2018)
11. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2. pp. 3111–3119. NIPS'13, Curran Associates Inc., USA (2013), http://dl.acm.org/citation.cfm?id=2999792.2999959
12. Miura, Y., Taniguchi, T., Taniguchi, M., Ohkuma, T.: Author profiling with word+ character neural attention network. In: CLEF (Working Notes) (2017)
13. Mosteller, F., Wallace, D.: Inference and disputed authorship: The federalist (1964)
14. Oliveira, R.R., Neto, R.F.O.: Using character n-grams and style features for gender and language variety classification. In: CLEF (Working Notes) (2017)
15. Patra, B.G., Das, K.G.: Dd. multimodal author profiling for arabic, english, and spanish. In: Proceedings of the Ninth International Conference of the CLEF Association (CLEF 2018) (2018)
16. 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)
17. Rangel, F., Franco-Salvador, M., Rosso, P.: A low dimensionality representation for language variety identification. In: International Conference on Intelligent Text Processing and Computational Linguistics. pp. 156–169. Springer (2016)
18. Rangel, F., Rosso, P.: Overview of the 7th Author Profiling Task at PAN 2019: Bots and Gender Profiling. In: Cappellato, L., Ferro, N., Losada, D., Müller, H. (eds.) CLEF 2019 Labs and Workshops, Notebook Papers. CEUR-WS.org (Sep 2019)
19. Rangel, F., Rosso, P., Montes-y Gómez, M., Potthast, M., Stein, B.: Overview of the 6th author profiling task at pan 2018: multimodal gender identification in twitter. Working Notes Papers of the CLEF (2018)
20. Rangel, F., Rosso, P., Potthast, M., Stein, B.: Overview of the 5th author profiling task at pan 2017: Gender and language variety identification in twitter. Working Notes Papers of the CLEF (2017)
21. Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M., Stein, B.: Overview of the 4th author profiling task at pan 2016: cross-genre evaluations. In: Working Notes Papers of the CLEF 2016 Evaluation Labs. CEUR Workshop Proceedings/Balog, Krisztian [edit.]; et al. pp. 750–784 (2016)
22. Rangel Pardo, F.M., Celli, F., Rosso, P., Potthast, M., Stein, B., Daelemans, W.: Overview of the 3rd author profiling task at pan 2015. In: CLEF 2015 Evaluation Labs and Workshop Working Notes Papers. pp. 1–8 (2015)
23. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. **34**(1), 1–47 (Mar 2002). https://doi.org/10.1145/505282.505283, http://doi.acm.org/10.1145/505282.505283
24. Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P.V., Kawanabe, M.: Direct importance estimation with model selection and its application to covariate shift adaptation. In: Advances in neural information processing systems. pp. 1433–1440 (2008)
25. Takahashi, T., Tahara, T., Nagatani, K., Miura, Y., Taniguchi, T., Ohkuma, T.: Text and image synergy with feature cross technique for gender identification. Working Notes Papers of the CLEF (2018)
26. Tellez, E.S., Miranda-Jiménez, S., Moctezuma, D., Graff, M., Salgado, V., Ortiz-Bejar, J.: Gender identification through multi-modal tweet analysis using microtc and bag of visual words. In: Proceedings of the Ninth International Conference of the CLEF Association (CLEF 2018) (2018)

27. Veenhoven, R., Snijders, S., van der Hall, D., van Noord, R.: Using translated data to improve deep learning author profiling models. In: Proceedings of the Ninth International Conference of the CLEF Association (CLEF 2018) (2018)
28. Yule, G.U.: On sentence-length as a statistical characteristic of style in prose: With application to two cases of disputed authorship. Biometrica **30**, 363–390 (1939)