Are You Human? Detecting Bots on Twitter Using BERT

David Dukić
University of Zagreb,
Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
david.dukic@fer.hr

Dominik Keča
University of Zagreb,
Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
dominik.keca@fer.hr

Dominik Stipić
University of Zagreb,
Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
dominik.stipic@fer.hr

***Abstract*—Dissemination of fake news on Twitter is a rapidly growing problem, mostly due to the increasing number of bots. Hence, automatic bot detection is becoming an important area of research. In this work, we present the BERT-based bot detection model along with exploratory data analysis of tweets written by bots and humans. We statistically prove that including additional features alongside contextualized embeddings boosts model performance. Furthermore, we develop a gender prediction model using derived features and compare the difficulties of the two tasks. Finally, we demonstrate how Logistic Regression outperforms Deep Neural Network on both tasks.**

***Index Terms*—BERT model, bot detection, emoji2vec, gender prediction, latent Dirichlet allocation, shallow vs. deep learning, t-SNE**

I. Introduction

Twitter is a popular social network where people can share their opinions in a form of so-called *tweets*. The number of Twitter accounts is increasing rapidly. Consequently, the number of published posts is becoming more diverse. Users write about politics, sports, news, emotions, etc. Here is an example of one such tweet: "*Those who think Trump acts solely on impulse are not paying attention.*". When reading this tweet, we can ask ourselves about the author's authenticity? Can we trust everything published online? Unfortunately, there are new players in the game which studies refer to as **bots**. These non-humans are prone to the dissemination of fake news which can be really dangerous. Novel research shows that there are groups of people who trust everything they read online and tend to share fake news believing that they are true [1].

Bots are autonomous agents that can be programmed with a goal to create and share fake news, political ideas, and spread tension in order to influence people's opinions. Although they only operate on social media, the way they affect people has a direct impact on real life. Spreading fake news on social media can influence the outcome of elections [2], spread problematic behavior and ideology. Many papers stress the importance of identifying bots in order to stop online manipulation, violence, and abuse [3].

To encourage further research and understanding of bots on Twitter, PAN[1] organizes a series of scientific events and NLP shared tasks. In 2019 PAN proposed two tasks under mutual name "*Bots and Gender Profiling 2019*" [4]. From now on we refer to them as bot detection and gender prediction, respectively. The competition was given in two languages: English and Spanish depending on the language in which tweets were written.

In this paper, we focus on exploratory data analysis of tweets written by bots in service of developing a machine learning model that differentiates between bots and humans based on their tweets' content. We present inferences from this analysis in Section III together with tweet pre-processing steps. For model training and evaluation we used English tweets from the PAN competition data set. We bring to the mind of the reader that our approach for solving given tasks was single tweet-based and not user-based. Thus, it is not comparable with the results of other competition's participants. To extract contextualized embeddings from tweets we used $\mathbf{BERT_{BASE}}$ model [5]. We also utilized emojis. They were represented using pre-trained emoji2vec embeddings [6]. Additionally, we made use of retweets, URLs, mentions, and hashtags. How we created features for our models along with their thorough description is presented in Section IV. While focusing on bot detection problem, we also built gender prediction models using the same types of features that we used for bot detection and compared difficulties of these two tasks. We report this in great detail in Section VI. In the same section, we also show a comparison of shallow and deep learning models and present conclusions from the analysis of shallow models' learned weights.

II. Related Work

In 2019 PAN organized competition whose goal was to improve bot detection and gender prediction on Twitter. Among the 56 teams that participated, the highest score for bot detection problem on English tweets was achieved by [7] using Random Forest. For gender prediction, best results were obtained by [8] where authors combined Logistic Regression

[1] https://pan.webis.de/

978-1-7281-8206-3/20/$31.00 ©2020 IEEE
DOI 10.1109/DSAA49011.2020.00089

with n-grams. A vast number of teams approached these problems using traditional, shallow, machine learning classifiers. However, in recent years, many deep learning models were introduced and they usually outperform shallow models on the same tasks. Therefore, some participants utilized models like Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), BERT model, etc.

The most similar approach to ours is the one that used BERT model [9]. While this approach puts focus on both tasks, our main objective was bot detection. It is important to note that we did not perform fine-tuning of BERT's weights on our tasks, but alternatively utilized BERT contextualized embeddings to develop Logistic Regression and Deep Neural Network models. Like many other participants, we also derived some additional features. In order to do that, we made use of retweets, URLs, mentions, and hashtags which we refer to as indicators. In spite of the fact that similar approaches used raw or processed frequencies of the same indicators as features, we only focused on their existence in tweets and converted them to categorical features before removing them in the cleaning process.

On social media, emojis play a great role in solving tasks like sentiment analysis. Sometimes, counting and grouping emojis into categories is not enough. As mentioned before, we represented emojis with emoji2vec embeddings. To the extent of our knowledge, none of the competitions' participants used this way of representation. We came to the conclusion that using emoji2vec in combination with other features improves models' performance.

Considering the popularity of deep learning, we wanted to see if its usage is justified. To explore this, we compared performances of shallow and deep learning models.

III. Exploratory Data Analysis

Before building a model, we did detailed exploratory data analysis to get some insights about our data set. First of all, we needed to transform our data set into the desired format, suitable for analysis and classification. The original data set had 100 tweets per user. Every user was labeled as either bot or human and additionally as male or female in the case of the human class. We wanted to conduct tweet-based and not user-based analysis and classification. Therefore, we transformed the initial data set in the form where each tweet was assigned its class corresponding to the class of the user who wrote it. This way we ended up with more examples than in the initial setting.

For an exploratory data analysis, we adopted a standard NLP cleaning pipeline which is composed of: language detection, tokenization, word normalization, POS tagging, and lemmatization. It is important to note that we worked only with English tweets and discarded tweets that contained more than 80% of non-English words. For detecting the language in which the tweet was written we used Python library langdetect[2]. Table I shows our data set summary before and after the cleaning phase.

[2]https://pypi.org/project/langdetect/

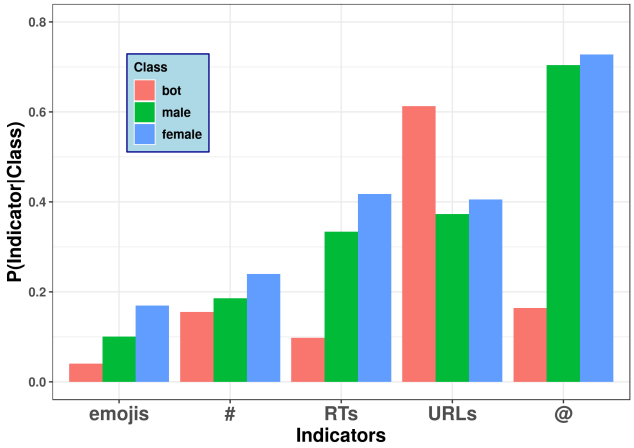


Fig. 1: Distributions of different indicators (#-hashtags, RTs-retweets, @-mentions) in each class of the train data set. Probabilities were estimated using MLE.

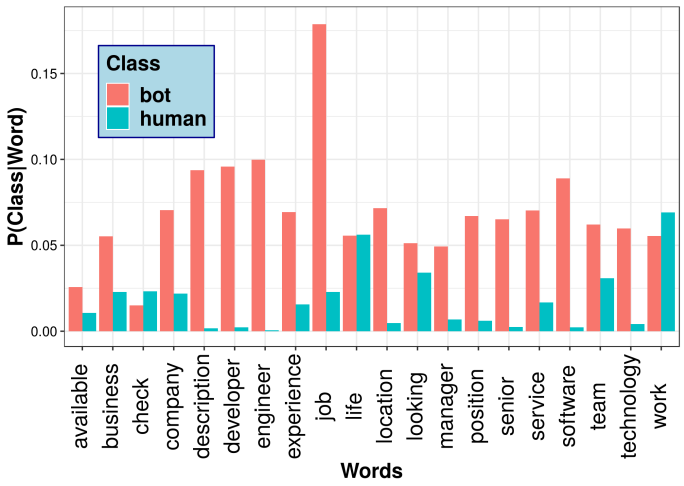
After cleaning our data set, we analyzed distributions of different tweet indicators. The results of this analysis are shown in Figure 1. These insights encouraged us to use the above-mentioned indicators as features for our models. In section VI we conclude that these features significantly improve our models' performances, especially for bot vs. human task.

Next, we went a step further and performed the Latent Dirichlet Allocation (LDA) which is an unsupervised clustering algorithm for document topic identification. With the LDA method, we obtained word distributions for each topic. Afterwards, we estimated the posterior probabilities $P(Class|Word)$ for each class with a MAP estimator:

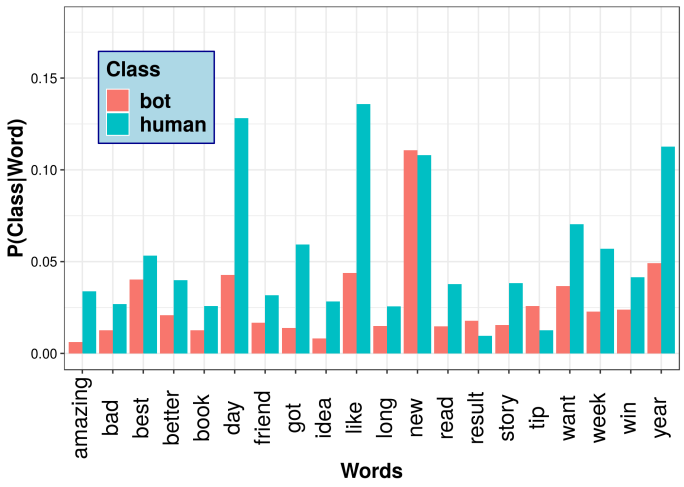
$$P(Class = c|Word = w) = \frac{C(c;w) + 1}{C(w) + |Word|}$$

where C denotes frequency. This method revealed several distinguished groups of tweets. Distributions of words for the most prominent topics are shown in Figure 2. Bots are dominant in Figure 2a and we concluded that they are in fact company bots that are used for employee recruitment or marketing purposes. Distributions shown in Figure 2b are largely dominated by humans. We assume that tweets that contain these words express common human terms like home, day, word, best, etc.

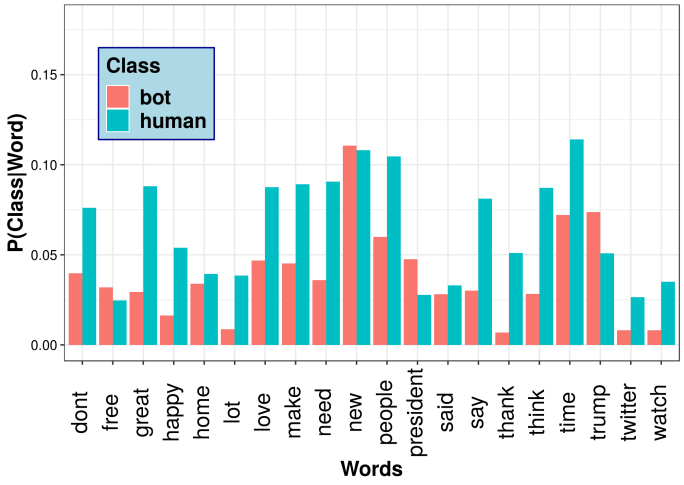
Furthermore, we found another significant topic that contains words such as American, law, tweet, woman, trump, news, story, etc. The mentioned set of words has a clear association with politics and it is in principle more common to human agents, but there is also a large proportion of bots that tweet about this topic. The latter discovery supports a claim that bots tweet about politics. Tweeting about politics might create conflicts between political groups. This was explained in [10] where authors found that bots polarized



(a) *Business* topic.



(b) *Everyday life* topic.



(c) *Politics* topic.

Fig. 2: Posterior distributions $P(Class|Word)$ for the most prominent topics of tweets (*business*, *everyday life*, and *politics*) for classes **bot** and **human**. Distributions were plotted for the 20 most important words in each topic.

TABLE I: Data set summary before and after cleaning of non-English tweets.

	Original		Cleaned	
	Train	Test	Train	Test
Bot	206000	132000	109317	74342
Male	103000	66000	89628	58480
Female	103000	66000	89233	57389
Total	412000	264000	288178	190211

members in online discussions during the Catalan referendum. Distributions of words for the so-called *politics* topic are shown in Figure 2c.

IV. Model Description

We start this section with a thorough description of features that we used for our machine learning models. Description of the features is followed by a discussion about shallow and deep learning models that have been utilized for both tasks.

A. Feature Extraction

In our first approach, we used word2vec embeddings [11]. Idea was to sum up tweet token vectors and fill them into the shallow machine learning model. This approach yielded satisfying results, but not as significant as our second approach with contextualized tweet embeddings. These were extracted using the BERT model and work remarkably well compared to the word2vec approach. We applied the so-called **BERT$_{\text{BASE}}$** model, which is the smaller version with 12 encoder components that produces 768-dimensional embeddings. For tokenization, the BERT tokenizer was applied.

Before we filled tweets into BERT, they were complemented with two special tokens [CLS] and [SEP]. The former is a so-called classification token and it is inserted at the beginning of the tweet. The latter is a special separation token for tweets that contain multiple sentences and it is inserted between them as a delimiter. We extracted tweet embeddings corresponding to the [CLS] token from the last hidden state and used this embedding as a crucial part of our proposed feature vector. The [CLS] token contains contextualized embedding of the whole text of the tweet. A more detailed demonstration of embedding extraction from tweet can be found in Figure 3.

Along with an initial set of features, we included additional features that represent emojis, retweets, URLs, mentions, and hashtags. The motivation for including this addition, alongside initial tweet text embeddings, can be found by consulting Figure 1. One can notice that there are significant differences between bots and humans in the number of indicators they use. This difference is less accentuated when comparing tweets written by males and females, but still significant.

Emojis were represented with 300-dimensional emoji embeddings called emoji2vec. Retweets, URLs, mentions, and hashtags were removed from original tweets and converted to the categorical variables which denote the existence of certain

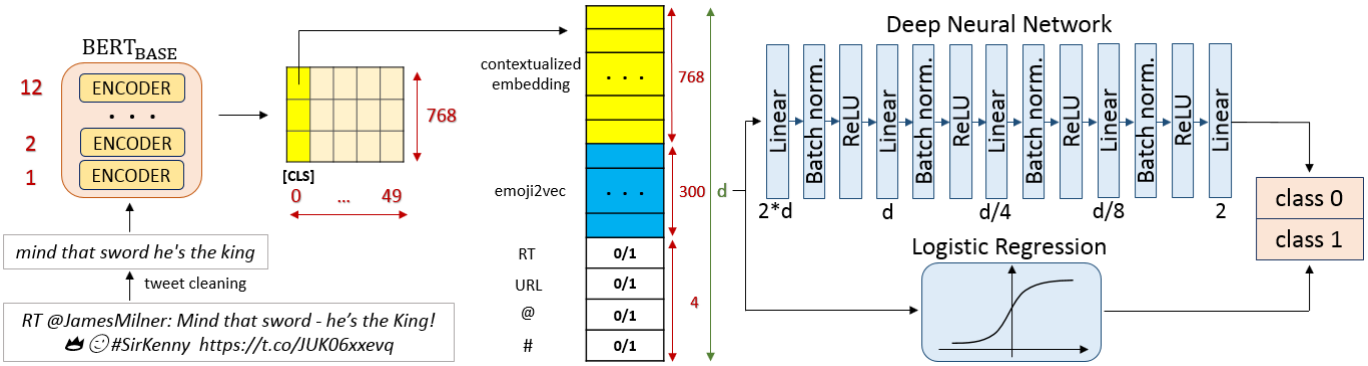


Fig. 3: A model schema for the proposed set of features.

category in the tweet. The proposed feature vector design for each tweet is shown in the middle of Figure 3. We concatenated BERT contextualized tweet embeddings with emoji embeddings and categorical features in the mentioned order. There were special cases e.g., when the original tweet didn't contain emojis or when it contained only emojis etc. To explain how we handled these cases, we introduce the following set: $tweet = \{text; emoji; RT; URL; mention; hashtag\}$ where *text* corresponds to string of the cleaned tweet. *emoji*, *URL*, *mention*, and *hashtag* are sets of emojis, URLs, mentions, and hashtags from original tweet, respectively. Feature function that covers all cases and for given *tweet* set extracts features can be written as:

$$f(tweet) = [\begin{cases} \mathbf{c} & text \neq \text{''} \\ \mathbf{0} & \text{otherwise} \end{cases};$$
$$\mathbf{0} + \sum_{k=0}^{|emoji|-1} \mathbf{e_k};$$
$$\mathbf{1}\{RT\};$$
$$\mathbf{1}\{URL\};$$
$$\mathbf{1}\{mention\};$$
$$\mathbf{1}\{hashtag\}]$$

where $\mathbf{c} \in \mathbb{R}^{768}$ denotes [CLS] embedding, '' denotes empty string, and $\mathbf{e_k} \in \mathbb{R}^{300}$ refers to emoji2vec embedding.

We bring to the mind of the reader again that we did not do fine-tuning of BERT's weights on given tasks, but instead used extracted contextualized embeddings as features for our shallow and deep learning models which we discuss next.

B. Shallow and Deep Machine Learning Models

It is well-known that deep learning models often outperform their shallow predecessors, but this need not be the case. The major disadvantage of deep models is the lack of their interpretability. In order to see which models work better on our features, we selected one shallow and one deep model for solving both bot detection and gender prediction tasks.

The traditional model perspective led us to try various models like SVM with radial basis function kernel, K-Nearest Neighbor, and Random Forest. However, the model with the utmost experimental success on derived features was L2-Regularized Logistic Regression. On account of having limited hardware resources, Logistic Regression proved to be the optimal choice for our setting. This is due to the fact that we only had to fine-tune its regularization strength. This way more resources were freed for conducting deep learning model experiments.

In the case of the deep learning model, we used a Feed-Forward Deep Neural Network whose architecture is presented in the right part of Figure 3. We experimented with various architectures and layers in our neural network, but neither of them provided a significant performance boost. In line with the *Occam's razor* principle, we opted for the simplest architecture and hyperparameter tuning. Our final architecture is composed of 3 different components: Linear layer, Batch normalization layer, and ReLU activation function. More details about our implementation settings are presented in the next section.

V. Implementation Details

To extract BERT contextualized embeddings we had to choose maximum sentence length hyperparameter. The decision was made to set maximum sentence length to 50 tokens since the majority of tweets were shorter than 20 words. This means that shorter tweets were padded with special [PAD] token to the length of 50 and longer tweets were cut-off after 50 tokens.

Regarding Logistic Regression implementation, we used the one from scikit-learn[3]. To extract contextualized embeddings, pretrained **BERT$_{\text{BASE}}$** uncased model from Hugging Face[4] was applied.

Pytorch[5] framework was utilized for implementing a Deep Neural Network. Its training was carried out using mini-batch Adam optimizer with batch-size 32, the initial learning rate of 0.5, and the same exponential decay factor value. To train and evaluate both shallow and deep learning models, we used train and test set splits provided by PAN. The validation set

[3] https://scikit-learn.org/stable/
[4] https://huggingface.co/transformers/
[5] https://pytorch.org/

TABLE II: Weighted-F_1-scores on the test set for all models. Bold scores denote best results obtained for both tasks.

	Model			
	L2-Regularized LR		DNN	
Task	Baseline	Proposed	Baseline	Proposed
Bot vs. human	77.717	**83.358**	74.896	81.784
Male vs. female	62.281	**64.176**	58.736	59.574

TABLE III: Results of two-tailed permutation tests on model pairs (L2-Regularized Logistic Regression Baseline - LR^1, L2-Regularized Logistic Regression Proposed - LR^2, Deep Neural Network Baseline - DNN^1, Deep Neural Network Proposed - DNN^2).

	Model pairs		
Task	LR^1 vs. LR^2	DNN^1 vs. DNN^2	LR^2 vs. DNN^2
Bot vs. human	$< 10^{-4}$	$< 10^{-4}$	$< 10^{-4}$
Male vs. female	$< 10^{-4}$	$< 10^{-4}$	$< 10^{-4}$

was constructed by randomly sampling 20% of tweets from the train set. We did this separately for bot detection and gender prediction task since the former has one class for both genders and the latter doesn't contain bot class. It is important to note that we used the same validation set for fine-tuning all models, either shallow or deep, but different depending on the prediction task.

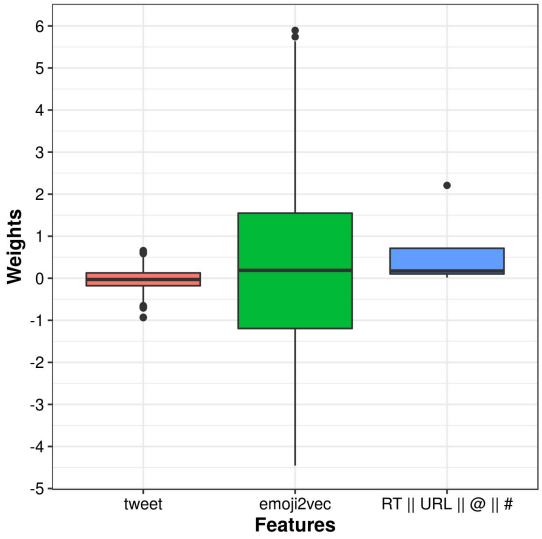
All models were trained on Google Colab[6]. For Logistic Regression models, we did regularization strength hyper-parameter tuning by choosing the model with the highest weighted-F_1-score on the validation set. This procedure took about an hour for each Logistic Regression model. Similarly, all deep learning models were fine-tuned on the validation set by measuring loss and applying early stopping. Learning on Google Colab with 10 epochs for each Deep Neural Network model took approximately 5 hours.

VI. Results

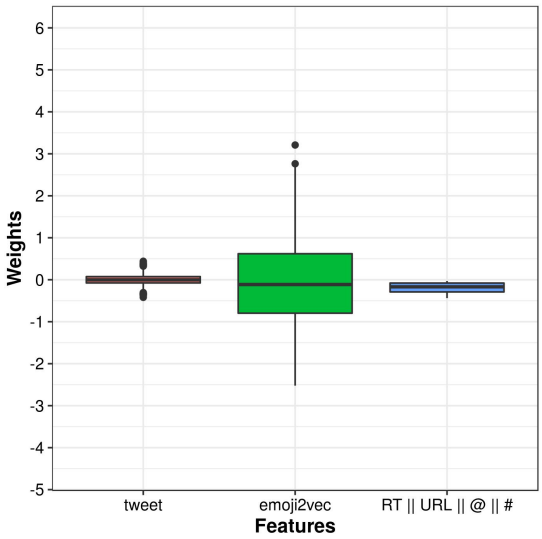
In this section, we report the evaluation results for Logistic Regression (LR) and Deep Neural Network (DNN) models with **proposed** features and features without additional indicators. We refer to the latter setting as **baseline** features. We evaluated our models on both tasks using weighted-F_1-score. Justification for the use of the weighted average comes from proportions of classes in the data set. Although the original data set had the same proportions for each class per task, the cleaned data set has more human than bot tweets and a roughly equal number of male and female tweets. This setting is more probable to be found in a real-world scenario. Consequently, we give more weight to human tweets than to bot tweets and equal weight to male and female tweets.

Experimental results are presented in Table II. Statistical tests between models are shown in Table III where we used

[6]https://colab.research.google.com/



(a) Bot vs. human task.



(b) Male vs. female task.

Fig. 4: Box plots of learned weights for L2-Regularized Logistic Regression on different feature types for both tasks (k represents concatenation of categorical features).

two-tailed permutation tests with 10000 rounds. Given outcomes induced a couple of interesting inferences.

We observed that proposed models achieved higher scores than baselines for both bot detection and gender prediction task. These differences are statistically significant at = 0:01 for both LR and DNN. Obtained results are given in columns two and three of Table III.

Additionally, we analyzed learned LR weights for the proposed model on both tasks. Here comes the advantage of the interpretability of LR over DNN. If we measure the importance of features by analyzing corresponding learned LR weights, we can conclude that more important feature groups have their weights less pulled down towards zero. This happens because