Detecting Bots in Social-networks using Node and Structural Embeddings

Ashkan Dehghan[1], Kinga Siuta[1,2], Agata Skorupka[1,2], Akshat Dubey[1], Andrei Betlen[3], David Miller[3], Wei Xu[1], Bogumił Kamiński[2][a] and Paweł Prałat[1][b]

[1]*Ryerson University, Toronto, ON, Canada*

[2]*SGH Warsaw School of Economics, Warsaw, Poland*

[3]*Patagona Technologies, Pickering, ON, Canada*

Keywords: Structural and Node Embeddings, Detecting Bots, Social Networks.

Abstract: Users on social networks such as Twitter interact with and are influenced by each other without much knowledge of the identity behind each user. This anonymity has created a perfect environment for bot and hostile accounts to influence the network by mimicking real-user behaviour. To combat this, research into designing algorithms and datasets for identifying bot users has gained significant attention. In this work, we highlight various techniques for classifying bots, focusing on the use of node and structural embedding algorithms. We show that embeddings can be used as unsupervised techniques for building features with predictive power for identifying bots. By comparing features extracted from embeddings to other techniques such as NLP, user profile and node-features, we demonstrate that embeddings can be used as unique source of predictive information. Finally, we study the stability of features extracted using embeddings for tasks such as bot classification by artificially introducing noise in the network. Degradation of classification accuracy is comparable to models trained on carefully designed node features, hinting at the stability of embeddings.

1 INTRODUCTION

Internet and social media impact all aspects of our lives. We use them to read news, connect with friends and family, share opinions, buy products, and entertain us. It affects our beliefs, behaviour and so it shapes our political, financial, health, and other important decisions. Unfortunately, as a result, social networks created an information platform in which automated accounts (including human-assisted bot accounts and bot-assisted humans) can try to take advantage of the system for various opportunistic reasons: trigger collective attention (Lehmann et al., 2012; De Domenico and Altmann, 2020; González-Bailón and De Domenico, 2021), gain status (Cha et al., 2010; Stella et al., 2019), monetize public attention (Carter, 2016), diffuse disinformation (Bail et al., 2020; Freelon et al., 2020; Monti et al., 2019), or seed discord (Woolley and Howard, 2018). It is known that a large fraction of active Twitter users are bots and they are responsible for much disinformation—see, (Yang et al., 2019) for many examples of manipulation of public opinion. Having said that, not all bot accounts are designed to harm or take advantage of other users. Some of them are legit and useful tools such as chatbots that respond to common questions of users, or knowbots that are designed to automatically retrieve some useful information from the Internet. On the other hand, human accounts may also spread disinformation and be responsible for some other malicious behaviour. Detecting bots and understanding roles they play within the system falls into a common machine learning task of node classification.

The main objective of this paper is to investigate whether graph embeddings extract information from the associated network that can be successfully used for node classification task. In our experiments, we concentrate on Twitter data and the task of identifying bot accounts but our questions (and answers) are much broader and so potentially more influential. They are applicable to all kinds of networks and data sets that are naturally represented as graphs which, of course, includes social media platforms such as Twitter. Moreover, they are applicable to a much wider class of machine learning tasks: node classification

[a] https://orcid.org/0000-0002-0678-282X

[b] https://orcid.org/0000-0001-9176-8493

Dehghan, A., Siuta, K., Skorupka, A., Dubey, A., Betlen, A., Miller, D., Xu, W., Kamiński, B. and Prałat, P.
Detecting Bots in Social-networks using Node and Structural Embeddings.
DOI: 10.5220/0011147300003269
In *Proceedings of the 11th International Conference on Data Science, Technology and Applications (DATA 2022)*, pages 50-61
ISBN: 978-989-758-583-8; ISSN: 2184-285X

algorithms train a model to learn in which class a node of the graph belongs to. Bot detection is a specific example of this class of problems in which a binary classification is performed (nodes are categorized into bots and humans) (Antenore et al., 2022). However, in general, multi-class classifications is also often considered and needed. Other important applications of this nature include, for example, identifying nodes associated with users that might be interested in some specific product, or detecting hostile actors. For this reason there is an increasing need for effective methods of analysis data represented as graphs. For more details we direct the reader to a recent survey (Matwin et al., 2021) and a book (Kamiński et al., 2021).

There are many approaches that can be used to perform node classification in graphs. Most techniques attempt to detect bots at the account level by processing many social media posts and analyzing them using various NLP techniques with the goal to extract some important and distinguishing features. These features are usually complemented with user metadata, friend metadata, content and language sentiment, as well as temporal features (Dong and Liu, 2018). In this paper, we will refer to these features as **NLP+P** (**NLP+P**rofile). These techniques are very powerful but a supervised machine learning algorithm is only as good as the data used for training. Unfortunately, good quality datasets with the ground-truth are rarely available. Additional challenge is that social bots evolve rapidly and so one needs to constantly update datasets and evolve the set of features to keep up with the other side. In particular, hot topics discussed on social media evolve rapidly; for example, NLP features that were important for bot detection before presidential elections in some country might become quickly outdated after the elections. Similarly, results of NLP analysis cannot be easily transferred from one language to another or across differ geographical regions or countries. As a result, a collaborated effort of many researchers and data scientists is needed to maintain bot detection models. One successful example is *Botometer*, bot detection tool developed at Indiana University using various labelled datasets and 1,209 features (current, 3rd version of the model) (Yang et al., 2019); see also (Sayyadiharikandeh et al., 2020) for a new supervised learning method that trains classifiers specialized for each class of bots and combines their decisions through the maximum rule (ensemble approach). *Botometer* handles over a quarter million requests every day! However, since the bot score is intended to be used with English-language accounts, what can one do with non-English accounts? What if the content or metadata is not easily available? Finally, how about other node classification tasks which cannot enjoy such powerful tools such as *Botometer*?

An alternative approach is to use some features of nodes that can be calculated exclusively using graph data. The main advantage of this approach is that such information is easier to obtain and is typically less sensitive as it does not include the analysis of user messages and metadata associated with them. More importantly, it can be hypothesised that the signal is more stable in time and graph space, that is, if some topological structure of the network indicates that some nodes are likely to be bots, then such signal is likely to loose its predictive power slower than, for example, discussion topics extracted from NLP features. Typical features concentrate on local properties of nodes such as node degree, various node centralities, local clustering coefficient, etc. We will call features derived using this approach as **GF** (**G**raph **F**eatures). The idea behind is that bots need to use some strategies to form an audience. They employ various algorithms to gather followers and expand their own social circles such as following popular accounts and ask to be followed back (Aiello et al., 2012), generating specific content around a given topic with the hope to gain trust and catch attention (Freitas et al., 2015), or even interacting with other users by engaging in conversation (Hwang et al., 2012). These algorithms create networks around the bots that should be structurally and topologically distinguishable from the ones around real human beings which, in turn, affect the extracted graph features. The same rationale applies to other applications of node classification.

The above approach, based on analysis of predefined graph features, was proved to be useful in various node classifications tasks but it has a few issues. First of all, very often features of one node alone are not enough to adequately classify this node. Indeed, bots typically work in a coordinated way and are not usually suspicious when considered individually. Hence, bot detection requires combining information about multiple bots and analyzing them together (Chavoshi et al., 2016). This often is very challenging, both conceptually as well as computationally, as it requires to consider at least a quadratic number of pairs of nodes. As mentioned earlier, graph features capture properties that are rather local whereas some embedding algorithms aim to extract more global and structural properties. Moreover, we often do not have access to a complete network but rather sample it using some sampling method. Unfortunately, the choice of a sampling algorithm may substantially affect **GF**. Finally, the features that are to be analyzed need to be predefined by the analyst.

Therefore, the result of this approach depends heavily on skills, knowledge, or just sheer luck of the user.

In order to solve at least some of these problems, we propose to utilize node embedding algorithms that assign nodes of a graph to points in a low dimensional space of real numbers. The goal of the embedding is to decrease the dimension but, at the same time, to extract the most important features of nodes and ignore noise. We will call features obtained based on this approach **EMB**. These algorithms have quickly became an intensely researched topics in recent years; see, for example, (Cai et al., 2018) or a recent book (Kamiński et al., 2021), and the list of potential applications constantly increases. After reducing the dimension via node embeddings, node classification can be done more efficiently compared to extracting graph features and using the original network to identify synchronized behaviour. On the other hand, synchronized behaviour should create similar network structure around the involved nodes and so should be captured by the embedding. Such group of nodes may be then potentially extracted (even in an unsupervised way) by some machine learning tools such as DBSCAN that are able to identify dense regions of the embedded space. Some of embedding algorithms not only capture local properties of graphs but also try to pay attention to global structure and different roles the nodes play within the network (Ribeiro et al., 2017; Donnat et al., 2018) which might carry more predictive power than local **GF**. Additional benefit of such approach, in comparison to using **GF**, is that features are identified automatically in an unsupervised way by the algorithm, as opposed to having to identify them manually by the analyst. Finally, embeddings seem to be less sensitive to sampling techniques and so they might be used as a foundation for more robust classification algorithms.

In this paper we show the following:

- **Profile** features are predictive for identifying bots, since they capture core social properties of the ground-truth network, such as *friends* and *followers* count.
- **NLP** approaches are powerful but are data specific and so cannot be generalized across time, regions and languages. Predictive features of NLP change over time and need to be constantly updated.
- **EMB** approaches provide an unsupervised way of capturing statistically significant predictive features.
- Features built using **EMB** techniques perform equally well for classifying bots as compared with carefully designed **GF** features.
- **EMB** techniques are as stable as features built using **GF** method, as the network is perturbed with noise.
- **EMB** features contribute incremental predictive power to bot classification when used in combination with **GF** features.

Finally, let us stress that despite the fact that these results are optimistic and show a potential of algorithms based on graph embeddings, this is an early stage of research in this direction. We finish the paper with a discussion of future work that will deepen our understanding of the power (as well as potential issues) of embeddings.

2 DATASETS

Developing and evaluating bot detection algorithms relies on the availability of unbiased labeled datasets. Although there are numerous datasets used for building and benchmarking bot detection algorithms, we mainly focus on using two recently curated Twitter datasets by Feng *et al.* (Feng et al., 2021) [**TwiBot-20**] and Setlla *et al.* (Stella et al., 2019) [**Italian Election**]. We recognize that labeled bot-datasets often contain some level of bias, since the real ground-truth is not readily available. In general, labels are identified by careful analysis of humans or by cleverly designed algorithms. Throughout our study, we ensure to stay aware of this fact and highlight any impact this may have on our findings.

In the **TwiBot-20** dataset, the authors focus on building a comprehensive Twitter dataset composed of semantic, property, and neighbourhood information. Here, semantic is the Tweet text generated by the user; property is the information related to user profile such as number of followers and following, and finally, neighbourhood is the network structure of the user. We highlight the features used from this dataset below. To capture a natural representation of the ground-truth Twittersphere, the authors implemented a breadth-first search algorithm, to sample and build the dataset. In this methodology, a user is selected as the root of the tree and subsequent layers are built using the directed follow edges of each user. This process is repeated up to layer 3, creating a sample network with a selected user at its root (Feng et al., 2021). The sampling algorithm used by the authors builds a directed graph, where nodes are users and edges are follow relationship. As highlighted by Feng *et al.*, this method of sampling does not focus on any particular topic or pattern and should be a more natural representation of the Twittersphere.

Raw features available from the **TwiBot-20** dataset are listed and discussed in the journal version

of this paper. Note that the values for these features are a snapshot captured at the time of sampling. We categorize each feature into three types: Profile, NLP and Graph. The profile features are datapoints available through Twitter's API, and highlight some properties of each user. As pointed out by Feng *et al.*, the followers and following are randomly selected. We use the raw user Tweets as the input to our NLP feature engineering—details are provided in the journal version of this paper. Lastly, graph features are build using the raw edge list provided in the **TwiBot-20** dataset. As mentioned before, an edge between two nodes indicated a follow relationship between the nodes. Although the original network provided by Feng *et al.* is a directed graph, we convert it to undirected graph for our analysis. Lastly, we note that the profile feature *verified* is excluded from the bot classification process. This is done for two main reasons. Firstly, most users accounts are not subject Twitter's verification process, where an account is confirmed to be owned by the user it claims to be. This process would inherently exclude bots from being verified. Secondly, due to the nature of the verification process, this feature could introduce bias for any classifier, thus making the discovery other meaningful features more difficult.

In the **Italian Election** dataset, Setlla *et al.* (Stella et al., 2019) aim to investigate the online social interactions during a 2018 Italian election and how it helps to understand the political landscape. In their work, the authors study relationship between real users and bots, using the Twitter network. Unlike the **TwiBot-20** dataset, the authors build a sample of the social network by focusing on tweets containing a list of political topics; such as "#ItalyElection2018", "#voto", etc. The sampling technique used by Setlla *et al.* results in a network with a vastly different graph topology than that created by Feng *et al.*. By sampling the TwitterSphere based on topics, the authors created a dataset in which nodes are users and edges represent interactions between users, such as retweets or mentions. Although this makes it difficult to compare the performance of bot detection algorithm between these two datasets, having diversity in how a social network is constructed helps us understand how bots manifest themselves within a network. The **Italian Election** dataset also contains labels indicating if a user is identified as a bot or not. As described by the authors, the bot/not-bot labels were generated by using an a classifier trained using Twitter user's profile information (Stella et al., 2019). Although the original dataset used by Setlla *et al.* (Stella et al., 2019) contains user profile and raw Tweet data, in this work we only have access to the network data and thus we can only focus on features extracted from the underlying network structure. Similar to the **TwiBot-20** network, the **Italian Election** graph is directed, with edges pointing from users who interact with other user's content. We also convert the **Italian Election** graph into an undirected graph for the purpose of our study.

High-level statistics of both networks are provided in the journal version of this paper. It is important to note that we apply additional data cleansing and filtering to provided dataset. For example, we run our analysis on the largest component of each graph, and convert both graphs into undirected networks.

3 PROFILE AND NLP FEATURES

In the journal version of this paper, we provide a detailed analysis of features extracted from user's profile information and their tweets. We perform feature engineering, specially on the raw tweets using various NLP techniques. Since we only have profile and tweet data from the **Twibot-20**, our analysis is centered around this dataset.

To maximize their impact on a social network, bots aim to mimic real-user behaviour. To this end, bots aim to create accounts and content that seem natural, such that it was generated by a real user. An example of such actions could include following other users, tweeting about relevant topics and engaging in conversations. Despite their effort, bots often leave behind signs that allow us to distinguish them from non-bots. Having said that, bots are constantly getting more clever. We provide a strong evidence that **NLP** approach cannot be easily generalized and might require constant re-training.

As one would expect, the way tweets are written seems to be different linguistically between bots and non-bots. However, with recent advances in Transformer models, computer generated text is becoming evermore human like. The current state-of-the-art is the OpenAI's GPT-3 (Brown et al., 2020), a generative model for NLP tasks with 175 billion parameters! GPT-3 has been demonstrated to be effective on a variety of few-shot tasks: due to its extensive pre-training and size, it is able to learn rapidly from very few training examples. It generates texts that are nearly indistinguishable from human-written texts: Humans correctly distinguished GPT-3 generated texts from real human texts approximately 52% of the time, which is not significantly higher than a random chance (Brown et al., 2020). For more details on GPT-3 and other related topics we direct the reader to, for example, a recent survey (Matwin et al., 2021).

4 GRAPH DERIVED FEATURES

Another potentially independent source for extracting features is rooted in the way users/bots interact with others in the network. One can capture this information by analyzing various graph properties derived from the underlying social-network. This can be done in two ways. One, by carefully designing statistical features of the nodes. Second, using unsupervised methods to learn node and structural representations of the nodes. In this section, we provide a detailed analysis of node feature engineering in addition to features extracted using various embedding techniques.

4.1 Node Features

In this section, we build node features derived from the underlying network structure using both **TwiBot-20** and **Italian Election** datasets. For extracting features we use `NetworkX` as well as `igraph` python packages depending on the efficiently of the corresponding algorithms. Here is the list of extracted node-features that were computed for all nodes. For detailed definition we direct the reader to, for example, (Kamiński et al., 2021) or any other textbook on network science.

- *degree_centrality* - degree (the number of edges the vertex has)
- *strength* - minimum ratio of edges removed/components created during graph decomposition process
- *eigen_centrality* - eigenvector centrality, a measure of the importance of the vertex (using relative scores)
- *closeness* - closeness centrality, a measure of the importance of the vertex calculated using the sum of the length of the shortest path between the vertex and other vertices
- *harmonic_centrality* - harmonic centrality (another variant of closeness centrality, calculated similarly)
- *betweenness* - betweenness centrality, a measure of the importance of the vertex calculated using number of shortest paths that pass through the node
- *authority* - authority score, sum of the scaled hub values that have edge to the given node
- *hub_score* - hub score, sum of the scaled authority values of the nodes it has edge to
- *constraint* - Burt's constraint, an index that measures the extent to which a person's contacts are redundant
- *coreness* - coreness (unique value of k such that a node belongs to the k-core but not to the $(k+1)$-core)
- *eccentricity* - eccentricity (the maximum distance from a given node to other nodes)
- *pagerank* - another way of measuring node importance - invented by Google Search to rank web pages in Google search engine output

In addition to the above list of features, we compute average, standard deviation, minimum and maximum of every feature for the neighbouring nodes of each vertex. A full list of these features is given in the journal version of the paper.

As we highlighted in section 2, there are major differences in how the **TwiBot-20** and **Italian Election** datasets were constructed. Firstly, the underlying network constructed in the **TwiBot-20** captures follower-following relationship, while the network in the **Italian Election** dataset represents interactions between users. Secondly, the sampling technique used in the **TwiBot-20** dataset results in much more uniform graph topology since at each sampling layer a fix number of nodes (followers) were sampled. This is in contrast to the **Italian Election** dataset, were nodes were more randomly sampled. The difference in the network topology between these two datasets is reflected in the values captured using the node-features (a full analysis of these differences is presented in journal version of the paper). This is indeed an important observation, since one could extract more meaningful node-features by resampling the same underlying graph using different techniques.

The features whose calculation did not involve neighbours indicate only slight differences between bots and non-bots, both in terms of feature count and magnitude of discrepancies.

Discrepancies between bot and non-bots groups are more visible on the distributions of features involving particular nodes' neighbours' during calculation. Similarly to the previous group of characteristics, differences are more visible on the **Italian Election** data and again, values in this dataset seem to have lower variance or variance among groups. In particular, values for bots' features seem to have even lower standard deviation, which may be an indicator of the fact that bots constitute a homogenous group. Nevertheless, as different conclusions may be drawn on the basis of **TwiBot-20** dataset, so this observation may be attributed to different sampling or annotating method.

The fact that node features constructed on the basis of data about vertices' neighbours may help in explaining being bot versus non-bot (at least more than pure node features) indicates the purposefulness of node embeddings usage. However, as this assumption is based solely on graphical analysis, one may be interested in modelling the relationship of node features and "being a bot". This is done in the following sections.

4.2 Node and Structural Embedding

There are over 100 algorithms proposed in the literature for node and structural embeddings which are based on various approaches such as random walks, linear algebra, and deep learning (Goyal and Ferrara, 2018; Kamiński et al., 2021). Moreover, many of these algorithms have various parameters that can be carefully tuned to generate embeddings in some multidimensional spaces, possibly in different dimensions. In this paper, we typically set all parameters but the dimension to the default values recommended by their authors. Once parameters are fixed, the algorithms learn the embedding in an unsupervised way. Having said that, some algorithms are randomized and so the outcome might vary. For our experiments, we selected 6 popular algorithms that span different families and includes both node as well as structural embeddings.

The first two algorithms, `Deep Walk` (Perozzi et al., 2014) and `Node2Vec` (Grover and Leskovec, 2016), are based on random walks performed on the graph. This approach was successfully used in NLP; for example the `Word2Vec` algorithm (Mikolov et al., 2013) is based on the assumption that "words are known by the company they keep". For a given word, embedding is achieved by looking at words appearing close to each other as defined by context windows (groups of consecutive words). For graphs, the nodes play the role of words and "sentences" are constructed via random walks. The exact procedure how one performs such random walks differs for the two algorithms we selected.

In the `Deep Walk` algorithm, the family of walks is sampled by performing random walks on graph G, typically between 32 and 64 per node, and for some fixed length. The walks are then used as sentences. For each node v_i, the algorithm tries to find an embedding e_i of v_i that maximizes the approximated likelihood of observing the nodes in its context windows obtained from generated walks, assuming independence of observations.

In `node2vec`, biased random walks are defined via two main parameters. The *return parameter* (p) controls the likelihood of immediately revisiting a node in the random walk. Setting it to a high value ensures that we are less likely to sample an already-visited node in the following two steps. The *in-out parameter* (q) allows the search to differentiate between inward and outward nodes so we can smoothly interpolate between breadth-first-search (BFS) and depth-first search (DFS) exploration.

The above algorithms primarily capture proximity among the nodes, nodes that are close to one another in the network are embedded together. But proximity among the nodes does not always imply similarity, as in the specific application we consider in this paper, bot detection. A role the nodes play within the network depends more on the structure of the network around them more than the distance between them. (See (Rossi and Ahmed, 2014) for a survey on roles.) The next four algorithms aim to create embeddings that capture structural properties of the network.

The first algorithm from this family, `Role2Vec` (Ahmed et al., 2019), generalizes the above techniques based on traditional random walks. To capture whether two nodes have the same role within the network, the notion of attributed random walks is introduced which is not tided to node identity but is instead using a function that maps a node attribute vector to a role. As a result, the algorithm learns associations among subsets of nodes (that is, roles) instead of properties of individual nodes.

`RolX` (Henderson et al., 2012) is another approach to explicitly identify the role of nodes using exclusively the network structure. This algorithm is based on enumerating various structural features for nodes in an unsupervised way, and finding the most suited basis vector for this joint feature space. Then, the algorithm assigns every node with a distribution over the identified roles (basis), allowing for mixed membership across the roles.

The next algorithm, `Struc2Vec` (Ribeiro et al., 2017) uses a hierarchy to measure node similarity at different scales. As a result, it constructs a multilayer graph to encode structural similarities and generate structural context for nodes. This hierarchical view is useful as it provides a sequence of more restricted notions of what it means to be structurally similar. At the bottom of the hierarchy, similarity between nodes depend exclusively on their degrees whereas at the top of the hierarchy similarity depends on the entire network.

The last algorithm we tested, `GraphWave` (Donnat et al., 2018) uses techniques from graph signal processing. It learns structural embeddings by propagating a unit of energy from a given node and characterizes its neighbouring topology based on the response

of the network to this probe. The runtime of this algorithms scales linearly with the number of edges.

As mentioned earlier, we fix most of the hyperparamters of each algorithm to their default value, and only adjust the embedding dimension. Of course, it is important to note that it is possible to optimize the outcome of each algorithm by searching for the ideal set of parameters for the task at hand, however our goal is not to optimize for the best metrics, but rather learn whether embeddings can help us in identifying bots in a social network.

5 BOT CLASSIFICATION

Thus far, we have focused on engineering and analyzing features built using user-profile, NLP, node-features and embeddings. In effort to understand the predictability of these features in identifying bot accounts, we train and test various classification models using the **TiwBot-20** and **Italian Election** datasets. Since the underlying data for these datasets are different, we divide our analysis into two section accordingly, focusing on each dataset separately. In both cases, datasets are sampled such that bot/non-bot classes are balanced (50/50). Furthermore, we use a 80/20 split for the train/test datasets. A 5-fold cross-validation process is then used to arrive the best performing model. All metrics are then computed using the test-set.

5.1 Bot Detection using TwiBot-20 Dataset

In this section, we use **NLP+P**, **GF** and **EMB** features to build a bot classification model. Our goal is not to optimize for the best performing model, but rather understand the predictive power of each feature-set. We build five models using various combinations of feature-sets. A summary of the performance of each model is highlighted in Table 1. As shown, the best performing model (based on accuracy) is the one trained on all features combined, achieving an accuracy of 81.76%. Furthermore, we note that models trained on **EMB** perform slightly better than those trained on **GF** alone. This enforces the fact that unsupervised embedding algorithms have the potential to learn complex and meaningful node features. More importantly, a model built on a combined **GF+EMB** performs better than **GF** and **EMB** separately, hinting that embedding features capture incremental predictive information about the nodes. Lastly, we note that models built using features extracted from the underlying network (**GF** and **EMB**) suffer from the uniform topology of the **TwiBot-20** dataset, as described previously. A different sampling technique could potentially result in a boost in the predictive power of features built using the network structure.

Next, we use feature importance analysis to explore the contribution of various features in the **GF + NLP+P** feature-sets to the performance of models. Since the embedding algorithms learn continuous representation of nodes in an unsupervised way, it is not easy to reverse engineer what each embedding dimension represents. Starting with feature importance using **GF** feature-set, we find that top two most predictive features for the **TwiBot-20** datasets are *degree_centrality* and *pagerank*. Given the high overlap in the performance of models built using **GF** and **EMB** datasets, one could postulate that embedding algorithms learn some form of centrality measure about the nodes.

Next, we apply similar feature importance analysis to the **NLP+P** feature-sets. According to top 20 variable importances the set of the top three predictor features are two original Twitter API variables, $followers\_count$, $listed\_count$, and the percentage of English tweets, *perc_en* extracted in the course of this study. This feature's importance has manifested in the basic EDA too. Further behind, we can see three variables with a similar importance, i.e, $friends\_count$, $links\_per\_tweet$ and av_tweet_len. Quite noticeable impact was noted for user mentions per tweet as well as for sentiment related features.

A complementary feature importance analysis can be done using Shapley technique. It is confirmed that the more followers users have, the least chance of them being a bot. Also, even though bots are not generally as followed on Twitter as bots are, the relationship is opposite for Twitter friendships—the extreme values for $friends\_count$ are generally associated with bots in the ML classifier as well. Another significant distinction between bots and humans can be found in $listed\_count$. Large number of public lists that this user is a member of is associated with being a human. The percentage of tweets written in English is not as monotonously related to being a bot, but the highest values are characteristic exclusively for bots. These users also seem not to be keen on using geographical tagging when they're posting (this was an opt-in feature on Twitter). The links and mentions per tweet tend to be higher for the identified bots, even though the initial basic EDA could not detect this tendency clearly for the links feature.

Table 1: Performance measure for models trained using **TwiBot-20** dataset.

Procedure	Accuracy	Precision	Recall	F1	MCC
NLP+P	0.8036	0.8059	0.8675	0.8356	0.5952
GF	0.6352	0.6357	0.8563	0.7297	0.2274
EMB	0.6481	0.6562	0.8153	0.7271	0.2594
GF+EMB	0.6620	0.6599	0.8507	0.7433	0.2905
NLP+P+GF+EMB	0.8176	0.8256	0.8657	0.8452	0.6246

5.2 Bot Detection using Italian-Election Dataset

In this section, we build and analyze bot classification models using the **Italian Election** dataset. As pointed out earlier, we do not have user profile or tweet data for the **Italian-Election** dataset, and thus all models are built using only the **GF** and **EMB** features. As before, we train our models on balanced datasets containing 50/50 proportion of bot and non-bot examples. Test, train split is kept consistent to 80/20 split. Furthermore, we keep the model architecture and all hyperparameters the same for all runs, to keep performance comparison consistent across all feature sets. We measure a number of metrics when comparing the performance of each feature set, including accuracy, recall, precision and Matthews correlation coefficient (MCC). Since some embeddings such as `Node2Vec` and `DeepWalk` are randomized (rely on random walks), we run those algorithms multiple times (sample size 100) and report our findings on the sample.

We summarize the result of our classification analysis in Table 2, highlighting the metrics used for comparing the performance measures. As a reference, we plot the model accuracy for each feature set in Figure 2. We note that the goal for this analysis was not to optimize the model performance, buy rather learn about the predictive power of each algorithm. As highlighted, using carefully designed node feature (as highlighted in the previous section), one could achieve accuracy of 66% in detecting bot vs. non-bot accounts.

Features such as *pagerank* and *coreness* show predictive power as compared with other node features. One could use this information to design similar features related to node centrality and coreness to capture additional predictive power, however this process is time consuming and could miss important features. As we have suggested, a different approach is to leverage unsupervised machine learning techniques to capture various properties on nodes, without time consuming featuring engineering. As we show in this section, one could utilize node and structural embedding to learn different types of representations of node, and combine them to gain even greater predictive power.

We see in Figure 2 that models trained exclusively on features extracted using embeddings perform inline with a model trained using engineered node features. Node embeddings, such as `Node2Vec` and `DeepWalk`, could learn information about the node's local community and proximity, while structural embeddings could learn information about the local structure of the network around each node. As we can see, a combination of these features could capture a broader representation of nodes, and in fact perform as well, or even better than a model trained on node features. In this case, *All Embedding* is a model trained on a feature set built by combining all the embeddings together. We note that to make our comparison fair, we apply Principal Component Analysis (PCA) to this combined feature set to reduce the dimensionality to 64 (same as embedding dimension of the embedding algorithms). We also highlight the fact that models trained using *Role2Vec* embedding outperforms all other embedding, indicating that for the **Italian Election** dataset, the structural property of the nodes is a better indicator if an account is bot vs. non-bot. Other metrics in presented in Table 2 tell similar stories, where models trained on all embedding perform best in our study.

Lastly, we perform feature importance analysis on the **GF** feature-set. Similar to the **TwiBot-20** dataset, *pagerank* appear in the top two most predictive features. Given the difference in the topology of the two graphs, **TwiBot-20** and **Italian Election**, it is interesting to observe *pagerank* appearing as one of the most important features. However, unlike **TwiBot-20**, we observe a number of important features derived from statistics of node-features from neighbouring nodes of each vertex. This indicates that identifying bots using node-features requires more than local properties of nodes. This further supports the use of embeddings, since many embedding algorithms can learn global properties of nodes.

We conclude this section by emphasizing that the performance of each embedding on this dataset

Table 2: Bot classification performance summary for the **Italian-Election** dataset.

Feature-Set	Accuracy	Precision	Recall	f1-Score	MCC
Node Features	0.66	0.65	0.66	0.66	0.31
All Embeddings	0.69	0.70	0.64	0.67	0.38
Node2Vec	0.61	0.62	0.57	0.59	0.22
DeepWalk	0.62	0.62	0.59	0.60	0.23
LSME	0.64	0.64	0.65	0.64	0.29
Struc2Vec	0.63	0.66	0.55	0.60	0.27
GraphWave	0.64	0.65	0.61	0.63	0.29
Role2Vec	0.67	0.68	0.61	0.65	0.34

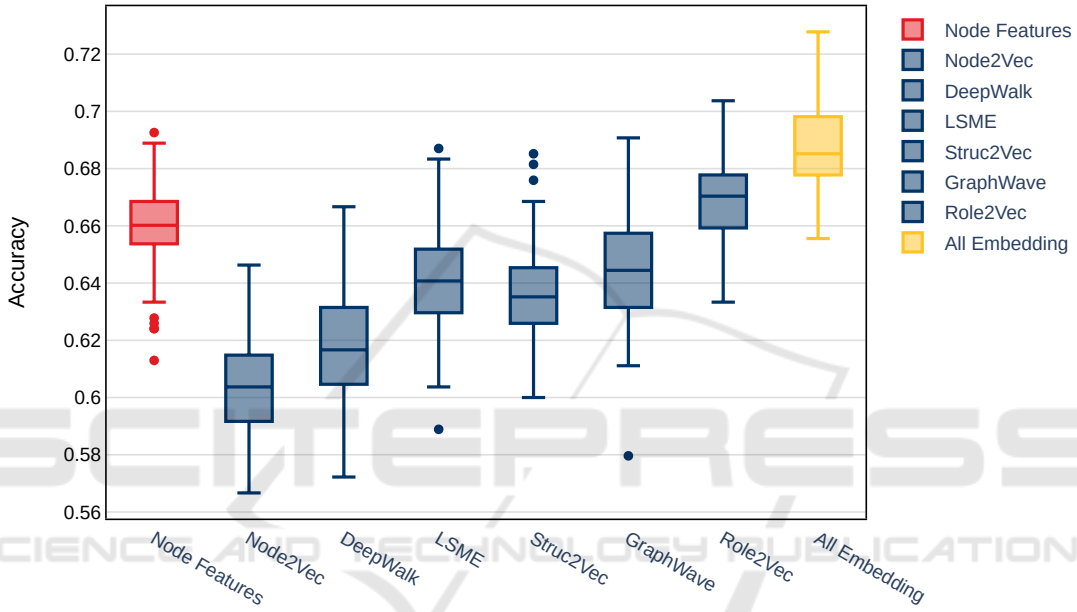

Figure 1: Accuracy for bot classification for the **Italian-Election** dataset. Distributions are over 10 runs of each embedding algorithm and 10 runs of classifier. Accuracy is measured using sample size 100 runs.

(**Italian-Election**) is not an indication that they perform in similar range if applied to other datasets. In general, embeddings are very application specific. This can also be said about hand-designed node features. Given the diversity amongst datasets, one should aim to use techniques that learn a wide range of representation of nodes in an unsupervised way. For this reason, an embedding technique (or a combination of them) could be power tool that could generalize well across various applications.

6 PREDICTIVE STABILITY

To understand the impact of data-noise on the quality of features extracted by embedding algorithms, we performed simulations, where we artificially introduced noise into the **Italian Election** dataset while measuring the accuracy of bot classifier models. As a benchmark, we compared all results against a model built using node-features. Based on our simulations, we found that features learned via embeddings are as resilient to noise as node-features. These results are highlighted in Figure 2, where we compared the performance of each embedding algorithm (blue) against a model built using node-features (red) and a model built using the combined embedding features (yellow).

7 CONCLUSION

In this work, we examined four distinct feature-sets extracted from the Twitter social network for identifying bot accounts. We divided the features into those captured directly from the Twitter network, NLP and

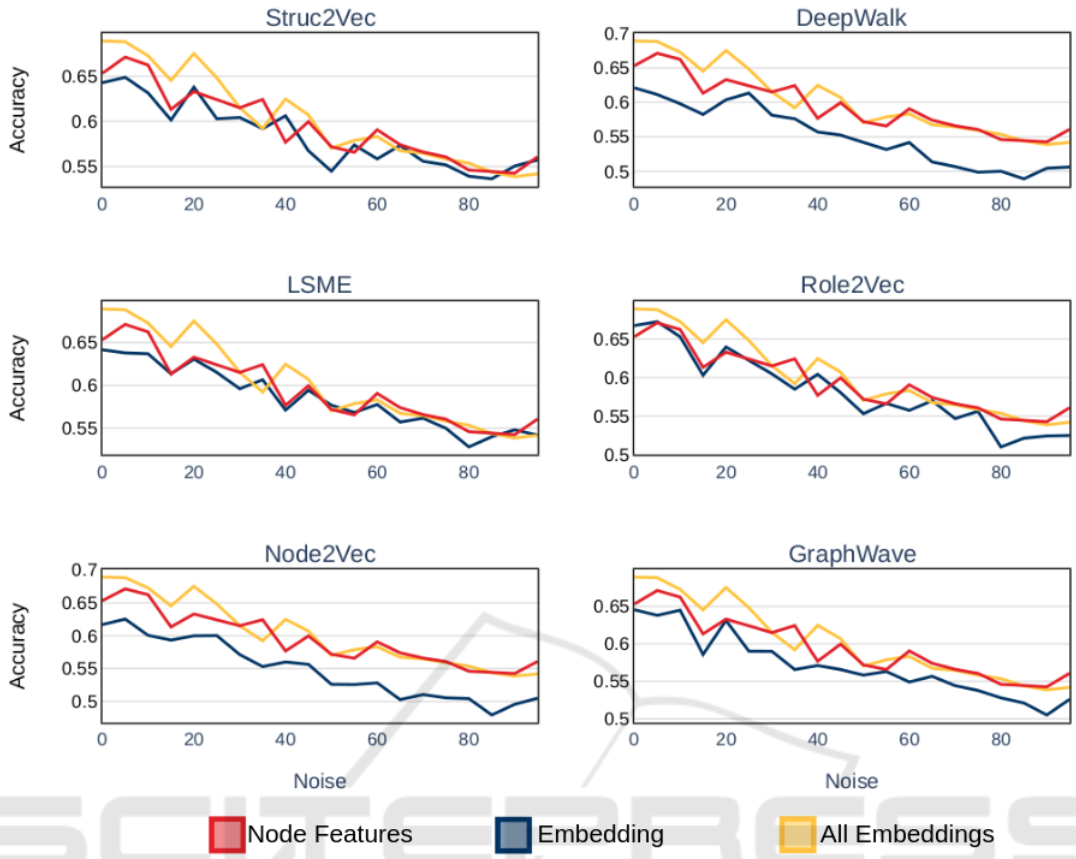

Figure 2: Stability of accuracy score: accuracy of node features (red), accuracy of specific embedding (blue), and accuracy of all embeddings combined (yellow).

user-profile data, and those derived from the underlying network structure, node-features and embeddings. We showed that NLP and user profile features have strong predictive power, however they suffer from the lack of generalizability. For example, language models trained for identifying bot accounts in a English speaking region can not be used to other regions. Similarly, clever bot accounts can modify their user profiles to appear more natural (non-bot like), and avoid being detected as bots. A much more difficult task however is altering the topology of the social network surrounding an account to appear as if it was created organically. Building on this intuition, we showed that the features extracted by mining graph structures indeed holds predictive power in helping us identify bots.

We analyzed features extracted from two Twitter datasets, one (**Twibot-20**) built using the follower as well as following relationships between users, and the other (**Italian Election**) constructed based on the interactions between users. We saw that in both networks, features mined from the underlying graph, either through node-feature engineering or learned in an unsupervised way via embedding algorithms, have predictive power. This hints at the fact that bot behaviour in a social network is distinguishable from non-bot users in both how they (bots) build relationships with other users and how they interact with them. An interesting future research question would be to study the impact of combining features from various networks, for example one built on follower/following and another on user-user interaction, for identifying bots.

Lastly, we showed that features captured using unsupervised embedding techniques are as predictive for bot classification as those built using node-feature engineering. One major advantage of using embedding algorithms is that they can learn a wide spectrum of properties about the proximity and structural properties of nodes. We also showed that one could combine features learned from various node and structural embeddings to a hybrid feature set.

REFERENCES

Ahmed, N. K., Rossi, R. A., Lee, J. B., Willke, T. L., Zhou, R., Kong, X., and Eldardiry, H. (2019). role2vec: Role-based network embeddings. In *Proc. DLG KDD*, pages 1–7.

Aiello, L. M., Deplano, M., Schifanella, R., and Ruffo, G. (2012). People are strange when you're a stranger: Impact and influence of bots on social networks. In *Sixth International AAAI Conference on Weblogs and Social Media*.

Antenore, M., Camacho Rodriguez, J. M., and Panizzi, E. (2022). A comparative study of bot detection techniques with an application in twitter covid-19 discourse. *Social Science Computer Review*, page 08944393211073733.

Bail, C. A., Guay, B., Maloney, E., Combs, A., Hillygus, D. S., Merhout, F., Freelon, D., and Volfovsky, A. (2020). Assessing the russian internet research agency's impact on the political attitudes and behaviors of american twitter users in late 2017. *Proceedings of the national academy of sciences*, 117(1):243–250.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Cai, H., Zheng, V. W., and Chang, K. C.-C. (2018). A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering*, 30(9):1616–1637.

Carter, D. (2016). Hustle and brand: The sociotechnical shaping of influence. *Social Media+ Society*, 2(3):2056305116666305.

Cha, M., Haddadi, H., Benevenuto, F., and Gummadi, K. (2010). Measuring user influence in twitter: The million follower fallacy. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 4.

Chavoshi, N., Hamooni, H., and Mueen, A. (2016). Debot: Twitter bot detection via warped correlation. In *Icdm*, pages 817–822.

De Domenico, M. and Altmann, E. G. (2020). Unraveling the origin of social bursts in collective attention. *Scientific reports*, 10(1):1–9.

Dong, G. and Liu, H. (2018). *Feature engineering for machine learning and data analytics*. CRC Press.

Donnat, C., Zitnik, M., Hallac, D., and Leskovec, J. (2018). Learning structural node embeddings via diffusion wavelets. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1320–1329.

Feng, S., Wan, H., Wang, N., Li, J., and Luo, M. (2021). Twibot-20: A comprehensive twitter bot detection benchmark. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 4485–4494.

Freelon, D., Bossetta, M., Wells, C., Lukito, J., Xia, Y., and Adams, K. (2020). Black trolls matter: Racial and ideological asymmetries in social media disinformation. *Social Science Computer Review*, page 0894439320914853.

Freitas, C., Benevenuto, F., Ghosh, S., and Veloso, A. (2015). Reverse engineering socialbot infiltration strategies in twitter. In *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 25–32. IEEE.

González-Bailón, S. and De Domenico, M. (2021). Bots are less central than verified accounts during contentious political events. *Proceedings of the National Academy of Sciences*, 118(11).

Goyal, P. and Ferrara, E. (2018). Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, 151:78–94.

Grover, A. and Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864.

Henderson, K., Gallagher, B., Eliassi-Rad, T., Tong, H., Basu, S., Akoglu, L., Koutra, D., Faloutsos, C., and Li, L. (2012). Rolx: structural role extraction & mining in large graphs. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1231–1239.

Hwang, T., Pearce, I., and Nanis, M. (2012). Socialbots: Voices from the fronts. *interactions*, 19(2):38–45.

Kamiński, B., Prałat, P., and Théberge, F. (2021). *Mining Complex Networks*. CRC Press.

Lehmann, J., Gonçalves, B., Ramasco, J. J., and Cattuto, C. (2012). Dynamical classes of collective attention in twitter. In *Proceedings of the 21st international conference on World Wide Web*, pages 251–260.

Matwin, S., Milios, A., Prałat, P., Soares, A., and Théberge, F. (2021). Survey of generative methods for social media analysis. *arXiv preprint arXiv:2112.07041*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Monti, F., Frasca, F., Eynard, D., Mannion, D., and Bronstein, M. M. (2019). Fake news detection on social media using geometric deep learning. *arXiv preprint arXiv:1902.06673*.

Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710.

Ribeiro, L. F., Saverese, P. H., and Figueiredo, D. R. (2017). struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 385–394.

Rossi, R. A. and Ahmed, N. K. (2014). Role discovery in networks. *IEEE Transactions on Knowledge and Data Engineering*, 27(4):1112–1131.

Sayyadiharikandeh, M., Varol, O., Yang, K.-C., Flammini, A., and Menczer, F. (2020). Detection of novel social bots by ensembles of specialized classifiers. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2725–2732.

Stella, M., Cristoforetti, M., and De Domenico, M. (2019). Influence of augmented humans in online interactions during voting events. *PloS one*, 14(5):e0214210.

Woolley, S. C. and Howard, P. N. (2018). *Computational propaganda: political parties, politicians, and political manipulation on social media*. Oxford University Press.

Yang, K.-C., Varol, O., Davis, C. A., Ferrara, E., Flammini, A., and Menczer, F. (2019). Arming the public with artificial intelligence to counter social bots. *Human Behavior and Emerging Technologies*, 1(1):48–61.