SimilCatch : Enhanced social spammers detection on Twitter using Markov Random Fields

Nour El-Mawass, Paul Honeine, Laurent Vercouter

SimilCatch: Enhanced Social Spammers Detection on Twitter using Markov Random Fields

Nour El-Mawass[a,*], Paul Honeine[a], Laurent Vercouter[b]

[a]*Normandie Univ, UNIROUEN, LITIS, 76000, Rouen, France*
[b]*Normandie Univ, INSA Rouen, LITIS, 76000, Rouen, France*

Abstract

The problem of social spam detection has been traditionally modeled as a supervised classification problem. Despite the initial success of this detection approach, later analysis of proposed systems and detection features has shown that, like email spam, the dynamic and adversarial nature of social spam makes the performance achieved by supervised systems hard to maintain. In this paper, we investigate the possibility of using the output of previously proposed supervised classification systems as a tool for spammers discovery. The hypothesis is that these systems are still highly capable of detecting spammers reliably even when their recall is far from perfect. We then propose to use the output of these classifiers as prior beliefs in a probabilistic graphical model framework. This framework allows beliefs to be propagated to similar social accounts. Basing similarity on a who-connects-to-whom network has been empirically critiqued in recent literature and we propose here an alternative definition based on a bipartite users-content interaction graph. For evaluation, we build a Markov Random Field on a graph of similar users and compute prior beliefs using a selection of state-of-the-art classifiers. We apply Loopy Belief Propagation to obtain posterior predictions on users. The proposed system is evaluated on a recent Twitter dataset that we collected and manually labeled. Classification results show a significant increase in recall and a maintained precision. This

*This is the corresponding author.
Email address: nour.el-mawass@etu.univ-rouen.fr (Nour El-Mawass)

validates that formulating the detection problem with an undirected graphical model framework permits to restore the deteriorated performances of previously proposed statistical classifiers and to effectively mitigate the effect of spam evolution.

Keywords: Social Spam detection, Online Social Networks, Twitter, Supervised Learning, Markov Random Field, Cybersecurity

1. Introduction

Stories of abuse of Online Social Networks have been frequently surfacing in the news scene. The Facebook Cambridge Analytica crisis and the repetitive bot-based manipulations of political elections on Twitter have brought to light the extent to which Online Media, and in particular, Online Social Networks (OSNs) can be abused. While the media involvement is relatively recent, there has been a decade long research effort to characterize, detect and control the ever increasing and proliferating forms of abuse on OSNs. Manifestations of spam and abusive behavior range from opinion manipulation and popularity inflation through to spammy advertisement, phishing, and malware dissemination. Attacks have been often undertaken by coordinated armies of fake (or sybil) accounts and occasionally by compromised accounts.

On Twitter, the 500 million tweets posted daily are impossible to monitor and filter manually, thus making the development and deployment of machine-learned detection and filtering systems a pressing necessity. The task is aggravated by the uniquely pronounced use of automation, which is now a major component of the abuse scene on Twitter.

The most commonly used paradigm to detect spammers on social networks is the supervised classification approach, which builds statistical classifiers of social accounts (or messages) based on features extracted from their profile, content, behavior and social network. Many early studies have shown that supervised classifiers were indeed able to yield high detection performance Benevenuto et al. (2010); Stringhini et al. (2010); McCord and Chuah (2011). Later

works Yang et al. (2011); Cresci et al. (2017a) have shown, however, that the supervised learning methodology falls short in keeping up with the complex and ever-changing social spam characteristics. The dynamic and evolving nature of social spam renders the rigid definitions and methods of the supervised paradigm especially ineffective.

Of special interest to this discussion is the concept of spam evolution. This is the process through which spammers change their characteristics and behavior with the goal of evading detection systems. As part of the population drifts away from the known pattern of spammers, the recall of machine learning systems is usually asymmetrically impacted Barreno et al. (2006). Other spammers do not drastically change their characteristics and they remain reliably detected. We use this asymmetrical deterioration in performance to motivate a change in the perception of supervised systems. Instead of detection, they can be seen as tools for discovering spammers in the wild. Discovery can furthermore be used for seeding, a concept that has been successfully exploited in unsupervised systems (e.g. in Leas, Youtube's unsupervised detection system where seeds are used to start localized graph clustering Li et al. (2016)).

The system we propose further exploits two main assumptions, namely that accounts similarity implies class homophily, and that predictions of supervised classifiers can be used to initialize beliefs about social accounts. This combined belief/similarity framework is an excellent candidate to a probabilistic graphical model. We use here the Markov Random Field, a simple yet versatile framework commonly used for belief modeling over dependent variables. This is an undirected graphical model, that models joint probability over a graph of dependent random variables. Social accounts are modeled as random variables, similarity is modeled as edges, and predictions of accounts classes by supervised classifiers as prior beliefs.

The classification problem can then be cast as an inference over dependent variables, and learning would correspond to finding the MRF parameters that minimise the classification loss.

Accounts networks on OSNs have traditionally been based on the who-

connects-to-whom network (called social graphs). Since using these as similarity graphs would require a problematic "strong trust" assumption Ghosh et al. (2012), we propose here a new content-based similarity measure. This is motivated by the observation that complicit spammers need to share the same (or similar) content. We show that content-based similarity offers an excellent way to construct graphs that are homophilic with respect to users classes.

We show that the proposed similarity measure successfully captures the concept of class homophily between social accounts.

The proposed system can be re-formulated and adapted to any online social network. We focus on the implementation of the system on Twitter since a substantial literature addresses the development of Twitter-based statistical detection features for the supervised learning paradigm.

In order to construct the bipartite content-users graph and evaluate the proposed model, we collect and manually label a recent dataset of Twitter accounts. This dataset, which we made public in a form that respects Twitter's rules, can be used by fellow researchers to reproduce the work in this paper and for further development and evaluation of similar models. Results demonstrate that, compared to individual prior predictions by state-of-the-art supervised classification systems, the probabilistic formulation leads to a significant increase in recall while maintaining high precision.

This work shows that traditional account-based supervised models, despite being inaccurate and lacking graceful degradation, can be effectively exploited in the context of a probabilistic framework. This is an important step towards the goal of exploiting scarce, inaccurate and biased predictions from variable sources (including biased statistical classifiers) and towards building detection systems that are more robust to features variations.

The remaining of this paper is structured as follows. Section 2 overviews related work on social spam detection by comparing and contrasting the existing detection approaches. Section 3 introduces the proposed system and explains its main components and parameters. Section 4 describes data collection and labeling techniques. Section 5 introduces the proposed content-based graph con-

struction mechanism and describes its implementation on Twitter. The Markov Random Field notation is introduced in Section 6. Section 7 presents the experimental evaluation and compares the results with existing approaches. Section 8 concludes the paper.

Research Contributions

The main implications of this work are summarized as follows:

- This paper shows that undirected graphical models can be used to model the problem of social spam detection.
- The Markov Random Fields formalism allows a hybrid social spam detection model that exploits both users features and their content-based similarity.
- A robust measure of similarity between users can be defined in terms of common content published by these users.
- The results validate that biased and inaccurate prior predictions on users classes can be effectively used in the context of probabilistic graphical models as demonstrated by the significant increase in recall obtained by the proposed approach.

2. Background and Related Work

2.1. Supervised Detection of Spam on OSNs

The first mention of social spam in a research work was in Yardi et al. (2009), where authors analyzed the behavior and characteristics of early forms of spamming accounts, namely accounts that posted unwanted URLs to trending Twitter topics. The rising problem of spam on OSNs was later formally addressed by several works Benevenuto et al. (2010); Stringhini et al. (2010); Lee et al. (2010); Chu et al. (2010); Jiang et al. (2016); Inuwa-Dutse et al. (2018). These works undertook the mission of characterizing and identifying spammers

using supervised learning tools and models. Their application spanned Twitter, Facebook and MySpace and their main contributions were focused on "features engineering". The proposed (and sometimes overlapping) sets of features were extracted from users' profiles, content, behavior and social network to characterize and identify spammers. They form the relatively constant core of later contributions.

One way to generally organize existing work is to classify the proposed systems based on the detection objective. We define the objective of the detection as the particular instance of the social platform the detection system wishes to identify or to label as abusive/spam-related. There are three distinct instances found in the literature: the social account, the social post (e.g. a tweet), and the URL. The first instance refers to a single profile on the platform, while a post represents the atomic unit of content. A URL is also an instance of content that can be found in text-based components, such as posts and about-me sections.

This classification platform can also be applied to subcategories of abusive behavior on social media. An example is the detection of fake followers Stringhini et al. (2013); Cresci et al. (2015) (sybil accounts following other accounts in order to inflate their popularity) which can be considered as a subcategory of spam accounts detection. Similarly, the task of detecting opinion manipulation and political propaganda Thomas et al. (2012); Ferrara (2017); Kušen and Strembeck (2020) can be considered a subcategory of the broader spam messages detection Wang et al. (2015); Clark et al. (2016). Although opinion manipulation is related to the more recent studies on fake news detection Zhang and Ghorbani (2019); Bondielli and Marcelloni (2019); Meel and Vishwakarma (2019), it is generally discussed in the context of bots-operated large scale manipulation, while fake news can propagate organically by legitimate accounts. Trends poisoning Lee et al. (2012), also known as hashtag hijacking, is the practice of linking one's content to trending topics by including these topics hashtags and keywords in the post text. It aims at diverting the public attention towards the hijackers content. This can be studied as a text Lee et al. (2012) or an account Benevenuto et al. (2010) detection problem. Note that URLs are usu-

ally detected in the context of spamvertising Zhang et al. (2014), and phishing and malware dissemination Aggarwal et al. (2012); Chhabra et al. (2011); Javed et al. (2019) but they can fit into the message detection category as well.

The problem of social spam detection is usually modelled under a supervised classification framework and the majority of papers consider that the task is to detect individual accounts. Few works deviate from this paradigm either by modeling the task as one of community classification (using community-based features) Bhat and Abulaish (2013) or by using an unsupervised platform Beutel et al. (2013); Cao et al. (2014); Li et al. (2016).

Along with detection systems, some works have attempted to quantify and qualify the mechanisms and dynamics that controlled the underground of malicious and abusive behavior on social media. The work of Thomas et al. Thomas et al. (2011, 2013) and Stringhini et al. Stringhini et al. (2013, 2012) on the spam underground and communities are notable in this domain.

The adversarial nature of the problem meant that spammers benefited from changing their characteristics in order to evade detection. Yang et al. Yang et al. (2011) demonstrate early evidence of spam evolution by underlining a change in spammers characteristics. Other studies Cresci et al. (2017a); El-Mawass and Alaboodi (2016) have evaluated the performance of state-of-the-art classification systems on recent datasets and have also conjectured that results indicate a spam evolution. A few recent articles have explored adversarial and proactive ways to predict spam evolution Cresci et al. (2018, 2019a,b); Washha et al. (2019).

2.2. Graph-based Detection of Social Spam

In contrast to the previously discussed approaches which constitute the bulk of the community contributions, a more recent paradigm is centered around the graphical representation of the problem by exploiting a major loophole in the spam strategy. The guiding assumption of this paradigm is that, in order to be effective, malicious attacks need to be at least loosely coordinated or synchronized. This results in sybil accounts being linked, either through the

social graph structure or through some form of similarity. This assumption is used to construct what can be called, depending on the application, a social Yu et al. (2008); Yu et al. (2008), interaction Li et al. (2016); Beutel et al. (2013) or similarity graph Cao et al. (2014); El-Mawass et al. (2018). Detection is therefore executed either by means of graph clustering (or cutting) Cao et al. (2014), or is modeled as a search for abnormally dense subgraphs Beutel et al. (2013). Detection models based on social graphs can be attacked by engineering social links with legitimate accounts. This issue is addressed in Fraudar Hooi et al. (2017, 2016), a next generation graph-based detection system designed to detect fraudsters in the presence of camouflage.

Some of these graph-based works are completely unsupervised Beutel et al. (2013); Cao et al. (2014), while some, especially works based on graph cutting, assume the presence of at least one label to help associate the identified clusters Danezis and Mittal (2009).

While most of the unsupervised detection approaches are graph-based, a notable exception is the work of Cresci et al. on DNA fingerprinting Cresci et al. (2017b). The approach assumes that collusive spammers have similar activity dynamics and proposes a DNA-like alphabet to model account's activity. Spammers are then detected by clustering similar DNA profiles.

2.3. Probabilistic Graphical Models for Online Abuse Detection

Previous applications of probabilistic graphical models in the context of anomaly detection on online platforms include Netprobe Pandit et al. (2007) and FraudEagle Akoglu et al. (2013); Rayana and Akoglu (2015), which target fraudulent accounts on online markets (e.g. Amazon and eBay) and review fraud, respectively. A notable example in the domain of malware detection is Semantic Norton's Polonium system Chau et al. (2011) which implements belief propagation over a large-scale bipartite graph of machines and files. Machines are assigned a proprietary reputation belief and the belief propagation helps identify malware files. Despite some similarities with the problem of online social spam detection, the context and formulation of these models are quite

different, making it impossible to transfer them directly to the setting of social spam detection.

On a closer front, applications of Markov Random Field (MRF) to spam detection have followed the traditional model of earlier graph-based approaches (e.g. SybilGuard Yu et al. (2006) and SybilLimit Yu et al. (2008)) by basing the users graph on the social structure of the network (the who-follows-whom graph). SybilBelief Gong et al. (2014) is a system that propagates known labels of users over the social structure using a MRF model. The system is tested on synthetic and real-world social graphs including the social graph of Facebook. SybilFrame Gao et al. (2015) is based on a similar idea but uses a probabilistic representation of users based on their perceived labels. The proposed system is evaluated on synthetic data and the social structure of Twitter. SybilBelief and SybilGuard are direct extensions of the established graph-based detection community which traditionally bases detection on the social structure network. The leading assumption is that links between users are based on a relationship of trust, an assumption that has been shown to be questionable on real online social networks including Twitter Ghosh et al. (2012).

This paper brings a completely different approach to the use of MRF for social spam detection. We position our work as an extension to the work of the machine learning community. Moreover, the proposed graph presentation does not use the social structure of the network, choosing instead to base the graph on the similarity between users. We thus avoid the notion of strong-trust that is assumed in structure-based contributions. This choice is also in line with the more recent graph-based contributions from Facebook and Youtube Li et al. (2016); Cao et al. (2014), which define a graph of interaction or similarity between users. On a practical note, constructing the graph with content-based similarity makes our work more easily reproducible and generalizable than works using proprietary social graph information.

3. Proposed System

3.1. Problem Formulation

Setting and Input. Assume we have a set of social accounts U, where each user $u \in U$ has an associated vector of numerical characteristics $\mathbf{x}_u \in X$ and a set of messages $M(u)$ that represents the content posted by this user. For each user u, we would like to assign a class y_u in $L = \{0, 1\}$ where 0 denotes a legitimate account and 1 denotes a malicious account. We also have access to an inaccurate oracle (e.g. a classification system or a blacklist) that predicts, for each account u, the probability $p(y_u = 1)$ that u is malicious based on $\mathbf{x}_u$ (or $M(u)$ in the case of a URL blacklist).

Domain knowledge indicates that if users have content in common, they are likely to share the same class, i.e. be both legitimate or both malicious. By defining a measure of similarity between users, it is possible to define $\mathcal{N}(u)$ the neighborhood of user u containing all nodes connected to u in the defined similarity graph.

Goal: We would like to know if, given the biased individual predictions offered by the oracle, it is possible to reach a better individual prediction for each user by taking into consideration its similarities to other users. Similarity is formulated here in terms of content and the leading assumption is that similarity can be used to calibrate the bias in the oracle's predictions.

Formal Definition: We formulate the problem of assigning a class to each user u as a classic classification problem where the goal is to find a mapping from the user's representation $\mathbf{x}_u$ to the set of labels L.

To take into account the similarity between accounts, we rely on the Markov Random Field formalism, which allows us to define dependencies between predictions of similar users. The model represents the class of each user u as a random variable Z_u and the relations between these variables as a graph $G(V, E)$ where $V = \{Z_u\}_{u \in \text{Users}}$ is the set of users predictions and E is the set of edges linking similar users.

Dependency between users is simplified by assuming the Markov property,

defined by a node being independent of all other nodes given its neighbors. On the defined undirected graph, the local Markov property is formally stated as: $P(Z_u \mid Z_{V \setminus u}) = P(Z_u \mid Z_{\mathcal{N}(u)})$ where $\mathcal{N}(u)$ is the neighborhood of u.

3.2. System Design

The problem we formulated above and the general system we propose to improve the classification performance are both platform-agnostic. The details of the implementation in this paper are specific to Twitter but the solution can be adapted to any social network platform in which a similar problem can be defined (e.g. Facebook, Instagram, etc...). Figure 1 illustrates the data flow and general architecture of the proposed system. The data flow can be summarized as follows:

1. **Data Crawling:** Accounts and content information is first crawled from the online social platform (Section 6).
2. **Features Extraction:** Distinguishing behavioral, social and content-based characteristics are extracted from accounts content and profiles, and a numerical features vector is assigned to each account (Section 7.1).
3. **Priors Computing:** A *prior* probability is assigned to each account given its numerical features vector (in Section 7.1, we obtain priors via state-of-the-art supervised classifiers but other sources can also be used to assign a prior belief to the class prediction).
4. **Graph Construction:** To construct the users similarity graph, a bipartite graph of users and messages is created to identify accounts that have identical or very similar content (Section 4).
5. **Posteriors computing**: Joint optimization of labels is finally applied using Loopy Belief Propagation over the constructed Markov Random Field (defined in Section 5) . Once the propagation converges, the most probable configuration of labels is inferred from the resulting posterior probabilities (Section 7.4).

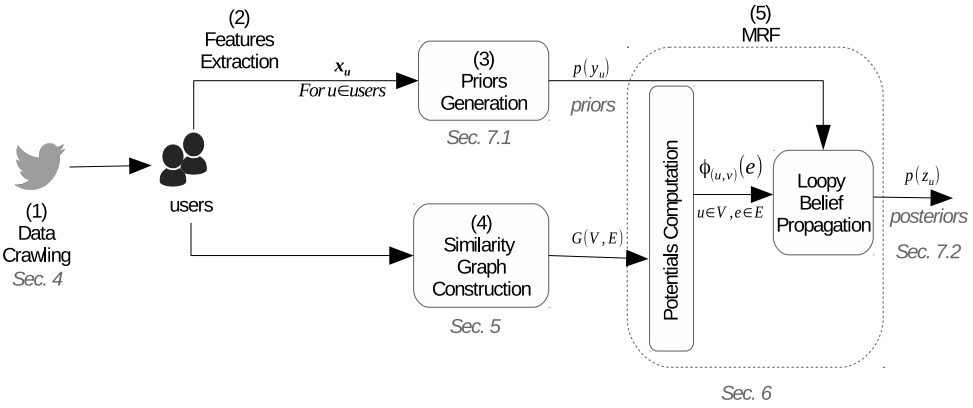

Figure 1: General architecture of the proposed system.

4. Users Similarity Graph

Accounts networks on OSNs have traditionally been based on the who-connects-to-whom network in bidirectional networks (e.g. the friendship network on Facebook) or on who-follows-whom networks in unidirectional networks (e.g. on Twitter). These networks can be used as similarity graphs by assuming "strong trust" between connected accounts. The "strong trust" assumption is the assumption that friendship between two accounts means that there is a bidirectional endorsement between these two accounts. Recent empirical analysis, however, suggests that the strong trust assumption is violated on unidirectional social networks. This is especially the case for Twitter Ghosh et al. (2012).

Recent unsupervised detection systems on Facebook (CopyCatch Beutel et al. (2013) and SynchroTrap Cao et al. (2014)) and Youtube (LEAS Li et al. (2016)) have used interaction graphs instead of social graphs. We propose a similar idea for Twitter where we base similarity on a bipartite content-users graph. The assumption here is that complicit spammers need to share the same content for better coverage. Shared content is also a more significant complicity signal than an unsolicited following link on Twitter. The users similarity graph consists of a graph where nodes are users and edges represent similarity between

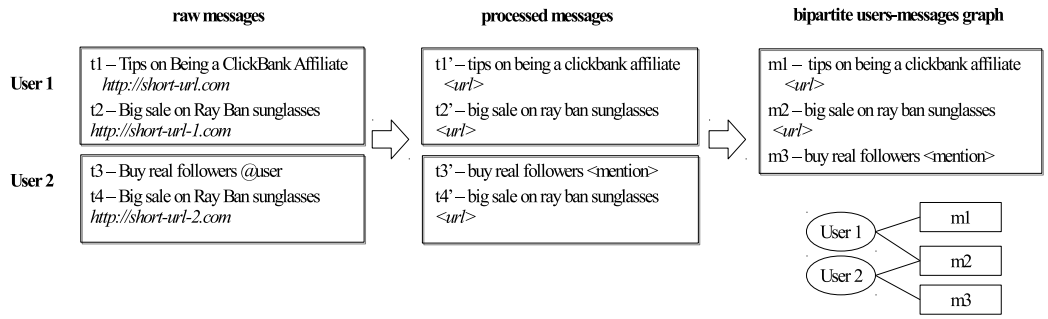

Figure 2: Construction of the bipartite users-messages graph on a toy example.

users.

The proposed definition of similarity relies therefore on the assumption that accounts that have common content tend to belong to the same class of users (see Figure 6 for a tweet that was shared across many profiles on Twitter). Specifically, spammers belonging to the same or similar spam campaigns tend to have similar content. We start by defining a bipartite users to messages graph and then collapse this bipartite graph into a users similarity graph. We also describe some special considerations to take into account when applying this general mechanism to Twitter.

4.1. Users-messages graph

To construct the bipartite users to messages graph, we start by processing the text messages published by each account and create an edge between accounts and the processed text of their messages. The process is illustrated in Figure 2.

Text pre-processing. Adding spurious characters, obfuscating final urls and varying the user mentioned in messages are all techniques that abusers are known to use to avoid their messages being detected as exact duplicates. Text pre-processing is therefore a vital part of the users-messages graph construction pipeline. In this instance, it consists of lowercasing and tokenizing texts and removing punctuation. Urls and users mentions are replaced by place holders (i.e. <url>and <mention>respectively). We do not replace hashtags by place

holders since they are frequently used as an integral part of the message text (due to the limitation on the number of characters per tweet). The described text processing is done once for each message by each user in the dataset, and the complexity of this step is linear in the number of messages.

Short texts containing less than three non place holders tokens (e.g. "Hi <mention>") are discarded to avoid creating false connections between users. The remaining exact replicates are merged and the set of resulting unique messages $\mathcal{M}$ forms the messages in the bipartite users-messages graph.

4.2. Users similarity graph

The users graph is generated by collapsing the messages in the bipartite graph. This is done by creating an edge between every pair of users that are connected to the same message as detailed in algorithm 1. Table 1 explains the notation used in the algorithm. This is a time consuming process since the complexity of generating all users pairs is quadratic in the number of edges between each message and its associated users. The number of users pairs is specifically equal to:

$$\frac{1}{2}\sum_{m\in\mathcal{M}} n_m(n_m-1), \tag{1}$$

where $\mathcal{M}$ is the set of processed messages and $n_m =\mid \{(u,m)\in E_{UM}$ for $u\in U\}\mid$ is the set of edges in the bipartite graph that link to message m.

Note that despite its high computational load, the generation of a users similarity graph is a prerequisite of many unsupervised detection models Li et al. (2016); Cao et al. (2014). It can be partially parallelized by assigning pairs generation of each message to a different process. A discussion of the parallelized implementation of a similarity graph on Facebook (based on login information and IP addresses) is provided in Cao et al. (2014). A threshold can be implemented in order to prevent the generation of pairs for highly popular content (messages in our case) Li et al. (2016); Cao et al. (2014). These popular messages are usually associated with legitimate content and the number of users linked to them would yield a significant computational load. Since the number

Table 1: Notation used for bipartite and similarity graphs.

Symbol	Description
U	set of users
$\mathcal{M}$	set of processed messages
u, v	users in U
m	a message in $\mathcal{M}$
n_m	number of users that posted message m
E_{UM}	set of bipartite user-message edges
(u, m)	a user-message edge in the bipartite graph
E_U	set of user-user edges in the users similarity graph
$G(U, E_U)$	users similarity graph
$G(U \cup \mathcal{M}, E_{UM})$	bipartite users-messages graph

of users associated with messages in our dataset is reasonable (most popular message is shared by 24 users, see Table 5), we did not have to implement a similar measure.

4.3. Twitter-specific considerations

The main assumption regarding the resulting bipartite graph described above is that created edges encode homophily: a malicious account creates spam messages while a legitimate account creates legitimate messages. While applying the general graph construction mechanism described above to Twitter, however, we became aware of two special cases in which an edge between a user and a message can be used to falsify credibility.

1. Content copying: A malicious account can engage in legitimate content copying. This leads to legitimate content (endorsed by links from legitimate accounts) being linked to malicious accounts, thus boosting the credibility of these accounts.
2. Compromising legitimate accounts: This has the opposite effect of content copying. When a spammer gains control of a legitimate account, any

Algorithm 1: Users pairs generation from the bipartite users - messages graph

Input: The set of processed messages $\mathcal{M}$ and the set of bipartite edges E_{UM}

Output: $E_U = (u, v, w)$ the set of weighted edges in the users similarity graph where $u, v \in U$ and w is an integer weight.

```
1  D ← new hash map
2  for m ∈ M do
3      U_m ← array of {u ∈ U | (u, m) ∈ E_UM}
4      for i = 1 to |U_m|−1 do
5          for j = i + 1 to |U_m| do
6              u ← U_m[i]
7              v ← U_m[j]
8              s ← min(u, v)
9              t ← max(u, v)
10             if (s, t) ∈ D then
11                 D[(s, t)] ← D[(s, t)] + 1
12             else
13                 D[(s, t)] ← 1
14 E_U ← {u, v, D[(u, v)]} for (u, v) ∈ D
```

malicious or spam content he publishes using this account will be endorsed by the previously established legitimacy of the compromised account.

Both problems can be solved using the notion of application on Twitter. An application, also known as the tweet source, is a term used to coin the software that published the tweet. Each application has a unique text identifier. We introduce the following changes on the algorithm described above.

1. Content copying: we identify a unique message by both its processed text and the source that published it. Thus, even if two messages share the same text, they are considered as different entities if they were not published by the same source. This restriction is reasonable since most legitimate accounts use the web interface or Twitter's mobile applications (e.g. Twitter for Android and Twitter for iPhone), while automated accounts use content management applications (e.g. dlvr.it and buffer) or custom scripts. Moreover, malicious campaigns often work in bursts to accomplish maximum visibility, and thus the shared content is usually published by the same application.

2. Exploiting compromised accounts: we introduce the notion of applications profiles. These are computed by extracting the application used to post each tweet and computing, for each account, the normalized proportion of tweets posted by each application. Since temporarily compromised accounts are often quickly restored, we expect that a compromised account that has a malicious message in common with a malicious account, will nonetheless have a significantly different application profile compared to this latter. To quantify this difference, we compute the cosine similarity between application profiles. For two users u and v, this is defined as the normalized inner product of the normalized applications vectors A_u and A_v (as illustrated in Figure 3). The similarity is defined as:

$$Sim(u,v) = \cos(A_u, A_v) = \frac{A_u^T A_v}{\|A_u\| \, \|A_v\|}, \text{ where } \|.\| \text{ is the Euclidean norm.} \tag{2}$$

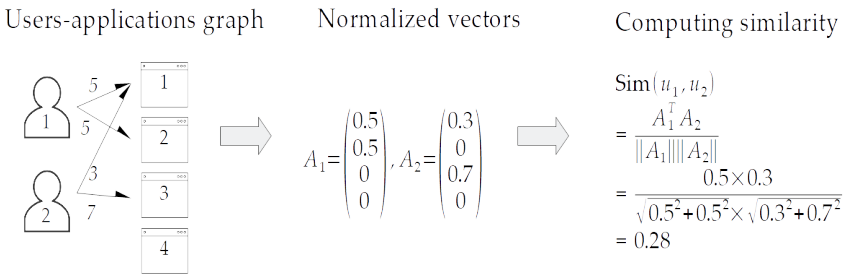

Figure 3: A toy example showing the computation of similarity based on applications profile.

5. Markov Random Field

An MRF (G, Ψ) is a probabilistic graphical model that allows joint inference over dependent random variables. It consists of a graph $G(V, E)$, where $V = \{Z_u\}_{u \in \mathrm{U}}$ is the set of random variables corresponding to users ($Z_u \in \{0, 1\}$), and $E = \{(u, v)\}$ is the set of edges denoting a dependency between two random variables Z_u and Z_v. A set Ψ of potential functions governs the relationships between random variables. Potentials are factors defined over cliques of nodes. In this work, we propose to use the pairwise MRF model which allows defining two types of potentials: edge (or pairwise) potentials and node (or unary) potentials.

5.1. MRF Potentials

Edge potentials are defined over edges in E. They ensure that the model responds to the smoothness criteria between connected variables in V, and generally direct the model towards predicting the same class for connected nodes. Unary potentials, on the other hand, are defined over individual nodes. They make it possible to take into consideration the features vector of each account by penalizing discrepancy between an observation vector $\mathbf{x_u}$ and the predicted class Z_u of user u. We construct these potentials as follows:

(i) A unary potential ϕ_u is a local function that quantifies how favorable a class is for node Z_u given its features vector $\mathbf{x_u}$. We define the unary poten-

tial here as a function that for each user $u \in U$ and class in L, associates a probability[1].

$$\phi_u : U \times L \rightarrow [0, 1]. \tag{3}$$

The unary potential is thus defined as a vector of two probabilities, the value of which is the system's prior belief about the class. This probability can be obtained from multiple sources including a supervised classification model trained on users features. This permits to indirectly incorporate the features information into the proposed MRF model as follows:

$$\phi_u(Z_u) = \begin{cases} 1 - p_u & \text{if } Z_u = 0 \\ p_u & \text{if } Z_u = 1 \end{cases} \quad \text{where} \quad p_u = p(y_u = 1 \mid \mathbf{x_u}) \in [0, 1] \tag{4}$$

(ii) An edge connects two nodes, Z_u and Z_v, if the corresponding users u and v are connected in the constructed similarity graph. Each edge is associated with a pairwise potential $\phi_{u,v}(Z_u, Z_v)$. In the current context, the edge potential is a function that represents compatibility between labels. Formally, edge potentials are defined as functions that for every realization of a pair of labels (in L), associates a real-valued factor quantifying its likelihood. Note that in this implementation, the edge potential is the same for all edges and is not conditional on the observations.

$$\phi_{u,v} : L \times L \rightarrow \mathbb{R}^+. \tag{5}$$

Specifically, we define the edge potentials as follows:

$$\begin{aligned} \phi_{u,v}(Z_u, Z_v) &= \exp(f(Z_u, Z_v)) \\ &= \exp \begin{array}{c} \begin{array}{cc} Z_v = 0 & Z_v = 1 \end{array} \\ \begin{pmatrix} w_0 & w_2 \\ w_1 & w_3 \end{pmatrix} \end{array} \begin{array}{l} Z_u = 0 \\ Z_u = 1 \end{array} \end{aligned}$$

where $w_{0-3} \in \mathbb{R}$ and $Z_{u\ (resp.\ v)} = 1$ if u (resp. v) is a spammer. (6)

[1]Note that in the general case, an MRF potential can take any value in the set of real numbers

$$\begin{pmatrix} 1-\epsilon & \epsilon \\ \epsilon & 1-\epsilon \end{pmatrix} \qquad \begin{pmatrix} e^{w} & 1 \\ 1 & e^{\alpha.w} \end{pmatrix}$$

(a) (b)

Figure 4: The edge potential matrix in (a) a symmetric special case, (b) an asymmetric case where inter-spammers and inter-legitimate connections are assigned different strengths.

We set $w_1 = w_2$ since they both designate a connection between a spammer and a legitimate user. The connection between two legitimate users and two spammers are governed by w_0 and w_3 respectively. There are two distinct cases that are generally used to model edge potentials:

- A symmetric MRF where the edge potential is the same for edges connecting spammers (w_3) and edges connecting legitimate users (w_0). A common setting is to set $\exp w_1 = \exp w_2 = \epsilon = 0.1$ and $\exp w_0 = \exp w_3 = 1 - \epsilon = 0.9$ (see the associated matrix in Figure 4a).

- An asymmetric MRF (Figure 4b), defining a flexible relation between parameters. We set $w_0 = w$ and $w_3 = \alpha.w$ where α is a positive tunable parameter. As we demonstrate in Section 7, this gives our model a greater expressiveness and allows it to more accurately capture the empirical relationships in the dataset. Since the model is over-parametrized, we set $e^{w_1} = 1$.

5.2. Computing Marginal Probabilities by Loopy Belief Propagation

Our goal is to obtain posterior class probabilities over nodes, given the node and edge potentials of the defined MRF. Exact computation of the marginal probabilities over the random variables requires summing the joint probability defined in eq. 7 over all possible labels permutations and is intractable for large graphs. Additionally, since the graph contains loops, efficient inference

algorithms designed for trees and chains are not applicable.

$$P(Z) = \frac{\widetilde{P}(Z)}{\sum_{Z' \in L^N} \widetilde{P}(Z')} \quad \text{where } \widetilde{P}(Z) = \prod_{u \in V} \phi_u(Z_u) \prod_{(u,v) \in E} \phi_{u,v}(Z_u, Z_v) \quad (7)$$

The Loopy Belief Propagation (LBP) algorithm Murphy et al. (1999) is an iterative message-passing algorithm that is frequently used to solve the inference problem on MRFs with general graph structure (e.g. in Computer Vision applications Freeman et al. (2000)). For graphs containing loops, LBP provides an approximate solution to the inference problem. LBP is considered linear in the number of edges. Its time complexity is $O(d\,|E|)$, where d is the number of iterations required until convergence and $|E|$ is the number of edges. The outline of the algorithm is provided in Algorithm 2. Although the algorithm does not offer convergence guarantees, it converges in practice after few iterations Murphy et al. (1999). Convergence is reached when beliefs converge (the inter-iterations difference is below a defined threshold, usually the machine epsilon).

Figure 5 illustrates how the predictions of a weak local classifier can be exploited to enhance the detection performance. It shows belief propagation on a cluster of 3 spammers. Edge potentials are computed for $\alpha = 2$ and $w = 0.6$ (see the next paragraph for more details on the choice of α and w). Because it is linked to two spammers, user 3 is correctly classified by MRF as a spammer. Moreover, users 1 and 2, being both linked and initially believed to be spammers, reinforce the prediction of each other, thus the probability of predicting the spammer class increases.

6. Dataset Collection and Labeling

A number of datasets containing annotated Twitter accounts has been introduced by previous work (e.g. in Benevenuto et al. (2010); Yang et al. (2011); Cresci et al. (2017a) and on the bot repository[2].). Due to Twitter's terms of

[2]The Bot Repository `https://botometer.iuni.iu.edu/bot-repository/datasets.html`.

Algorithm 2: Loopy Belief Propagation algorithm

Input: MRF (G, Ψ), $\mathcal{E}$

Ψ : $\phi_u(Z_u)$, $\phi_{u,v}(Z_u, Z_v)$.

$\mathcal{E} = \{(u,v),(v,u)\}$ for $(u,v) \in E$.

Output: Posterior marginal probabilities, $p(Z_u)$ for $u \in V$

`/* Initialization */`

$m_{u\to v}(Z_v) = 1$ for $(u,v) \in \mathcal{E}$ `/* Messages uniformly initialized */`

$b_u(Z_u) = 1$ `/* Beliefs of all nodes initialized to 1 */`

repeat

  `/* Update messages */`

  for $(u,v) \in \mathcal{E}$ **do**

    $m_{u\to v}(Z_v) = \sum_{Z_u} \left(\phi_u(Z_u)\phi_{u,v}(Z_u, Z_v) \prod_{i \in Ne(u)\setminus v} m_{i\to u}(Z_u) \right)$

  `/* Compute node beliefs */`

  for $u \in V$ **do**

    $b_u(Z_u) \propto \phi_u(Z_u) \prod_{v \in Ne(u)\setminus v} m_{v\to u}(Z_u)$

until *convergence*;

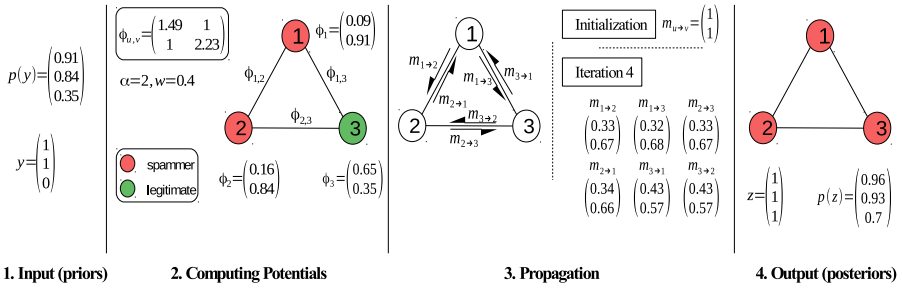

Figure 5: Loopy Belief Propagation illustrated on a cluster of 3 spammers. Input priors and potentials are shown on the left. The central frame shows the first and last iterations of belief propagation. The output frame shows posterior probabilities computed from the final values of messages. The algorithm in this instance converges after 4 iterations.

service, researchers are not allowed to share users' content of their datasets externally[3]. Most of the mentioned datasets therefore consist of users ids (and occasionally a vector of numerical features computed for each user e.g. in Benevenuto et al. (2010)). Moreover, Since Twitter executes regular purges of suspicious accounts, content of suspended and deleted accounts in the aforementioned datasets is no longer accessible[4].

Given that our model requires computation on users content for both features extraction and users graph construction, we collected and manually labeled a custom ground-truth Twitter dataset[5] to evaluate the model. The dataset contains 767 users[6] divided over four categories of users: verified accounts, normal users, hashtag hijackers and promoters. The first two categories belong to legitimate accounts and constitute 83% of the dataset, while the other two categories constitute the remaining 17% and exhibit an abusive behavior that violates Twitter terms of service[7]. Table 2 summarizes the general characteristics of the ground-truth dataset. For each of these users, Twitter's *Rest API*[8] was used to crawl users profiles and tweets. These were subsequently used to extract relevant content and behavioral features.

We explain hereafter the techniques used to collect and label Twitter accounts. Note that to obtain some users (e.g. users in the Verified category as well as some human users and promotional spambots), we needed to first collect a large dataset of random accounts and tweets. For this, we used the Developer *Streaming API*[8] in the period between 5 and 21 October 2017 and obtained a

[3]Twitter's policy on research use cases `https://twittercommunity.com/t/policy-update-clarification-research-use-cases/87566`

[4]See Bastos and Mercea (2019) for unique insights on the volatile nature of manipulative content on Twitter.

[5]The dataset (users ids, features and users graph) is available via `https://nourmawass.wordpress.com/datasets/`.

[6]The number of users in our dataset is comparable to other datasets obtained via manual labeling e.g. 759 users in RTbust Mazza et al. (2019), 62 and 529 accounts in Yang et al. (2019) and 1065 accounts in Benevenuto et al. (2010).

[7]Twitter terms of service `https://twitter.com/en/tos`

[8]Twitter developers API `https://developer.twitter.com/en/docs`

Table 2: Characteristics of the ground-truth dataset

Group Designation	**Class**	**Users**	**Tweets**
Verified Users	Legitimate	500	100 108
Human Users	Legitimate	130	56 663
Trends Hijackers	Spammer	51	22 586
Promotional Spambots	Spammer	86	31 404
Total		767	210 761

random sample of $20M$ tweets from $12M$ active users. For the remaining accounts in the ground-truth dataset, the collection targeted trending hashtags, the content of which was collected via the *Search API*.

6.1. Verified Accounts

Since automation on Twitter can be used by both legitimate and sybil users, it is important that the dataset comprises automated users from both categories. Verified users often belong to companies, celebrities or public figures, and are often operated by dedicated or generic content management applications[9]. They exhibit a behavior typical of what has come to be known in the literature as a "cyborg" account. These accounts may therefore have different features from those of normal human-based accounts and it is important to include them in the dataset to prevent the classifier from learning that every automated behavior is abusive.

Verified users are easy to identify (their profiles are marked with a blue tick mark and their crawled profiles include a "verified" flag). We randomly selected 500 users among 43k verified users appearing in the dataset and we included these 500 users in the ground-truth dataset.

[9]Examples of generic content management applications include *TweetDeck* and *dlvr*.

Figure 6: A screenshot of a compromised verified account posting a tweet containing a phishing link.

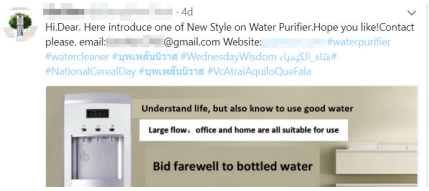

Figure 7: An example of trend-hijacking spam on Twitter.

6.2. Human users

The remaining 134 legitimate users in the ground-truth dataset were normal human-operated accounts. These users were identified by manually investigating a sample of active accounts from the initial dataset. This required a careful examination of the account in question, its tweets, profile and behavioral characteristics, and has therefore a small throughput. We elaborate on the pitfalls and advantages of manual labeling in the next paragraphs.

6.3. Promoters

The blacklisted links heuristic is a well-known heuristic that is commonly used to identify spammers in email and social media Aggarwal et al. (2012); Lee and Kim (2012). It consists of identifying users that post links to malicious webpages by verifying links appearing on social media against a continuously updated database of malicious webpages such as Google Safe Browsing[10] and Phishtank[11].

We applied this heuristic to the crawled dataset. For this, we first started by extracting all $3.8M$ links in the $20M$ crawled tweets. We subsequently wrote a program that follows the redirection chain of each link and returns the final landing webpage[12]. We then used Google Safe Browsing API to identify suspi-

[10]Google Safe Browsing API: `https://developers.google.com/safe-browsing/`

[11]The Phishtank database `https://www.phishtank.com/`

[12]To detect dynamic redirection, we used the *selenium* Python package to open each URL in a browser window.

cious URLs. Only 156 URLs were identified as being phishing or malware URLs. We extracted all users IDs that posted any of these malicious URLs and then proceeded to the manual verification of the resulting accounts. Surprisingly, a significant number of these accounts were actually legitimate accounts that were temporarily compromised[13] by a malicious posting mechanism[14]. Consequently, we could not rely on this labeling heuristic alone to obtain malicious accounts as it yielded a high false negative rate. Alternatively, for the users that were found to be genuinely malicious, we extracted the text associated with the blacklisted URLs. We then searched Twitter for users that posted the same text, and were able to identify several communities of spammers. We obtained 86 users in total, most of them engaged in promotional and site referral activity.

6.4. Trends hijackers

Trend hijacking is a type of collective-attention spam Lee et al. (2012) that is particularly ubiquitous on social media. It consists of poisoning trending topics (which typically offer high visibility and attract a large audience) with unrelated posts, often to promote a particular product or service (see Figure 7 for an example) or to manipulate public opinion Ratkiewicz et al. (2011); Thomas et al. (2012).

We obtained 47 trends hijackers by reading the tweets of a trending sport-related hashtag and manually identifying suspect tweets. This was followed by a manual investigation consisting of reading the recent tweets of suspect profiles and cross-examining different profiles for similar patterns and content. This process is similar to the one described in Cresci et al. (2016); El-Mawass

[13]Compromise is fairly common on social media. We used a variation of the Compa system described in Egele et al. (2013) for identifying and excluding compromised accounts among identified suspicious accounts. Compa builds statistical profiles for users and identifies compromise by comparing recent posts with the previously built profile.

[14]In one instance of these compromise campaigns, the "Rayban sale" scam, one verified account was found to retweet the same malicious URL dozens of times before the malicious behavior stops and the account restarts its normal behavior (see Figure 6).

and Alaboodi (2016). Manual labeling is different from mainstream labeling techniques described in the literature in that it is time consuming and requires an annotator that is familiar with current spam techniques and tricks[15].

7. Experimental Evaluation on Twitter

One of the leading motivations of this work is to establish that the extensive literature on supervised classification, which attempts to characterize spammers based solely on their behavioral, content and social network attributes, can still be leveraged . The main drawback of supervised approaches is that their performance degrades over the time as has been discussed in previous works Yang et al. (2011); Cresci et al. (2017a). We show that, even with beliefs produced by a weak supervised classifier, the MRF model can leverage the prior predictions of the supervised classifiers and output improved predictions. In the following, we define the statistical features and classifiers that we used to produce prior predictions for users in the groundtruth dataset.

We evaluate the performance of the proposed MRF-based model over the ground-truth Twitter dataset. We compare these results to the baseline performance yielded by state-of-the-art supervised classifiers and discuss their significance and implications. We use the 157 accounts belonging to the graph constructed in 7.2 to evaluate the MRF model with the prior probabilities predicted by the supervised classifiers. The remaining 610 accounts form the training dataset for the supervised classifiers (80% of the ground-truth dataset).

Section 7.1 discusses obtaining prior predictions via traditional supervised classifiers. Section 7.2 presents the implementation of the graph construction over the ground-truth dataset. Section 7.3 introduces the details of MRF and

[15] Previous work that uses manual labeling such as Benevenuto et al. (2010) relies on crowd-sourced annotation of individual hashtag tweets. While we think that this method could have yielded trustworthy annotation back when spam was less complicated and more straightforward, recent empirical evidence Freeman (2017); Cresci et al. (2017a) suggests that non-initiated human annotators fail to identify the new generation of spam on social media.

LBP implementation. The results are presented and discussed in Section 7.4.

7.1. Generating Prediction Priors

For each user u, with features vector denoted $\mathbf{x}_u$, the classifier predicts a class y_u with a probability $p(y_u)$. This probability quantifies the classifier confidence of its prediction and is the prior prediction probability used in LBP.

7.1.1. Statistical account features

We select 28 features from different previous works Benevenuto et al. (2010); Stringhini et al. (2010); McCord and Chuah (2011); Lee et al. (2010) and compute their values for accounts in the dataset. A list of these features along with their description is presented in Table 3. This set captures a wide range of information including aspects related to the accounts behavior, social network, content and social profile. We also specifically reproduce the works in Benevenuto et al. (2010) and Stringhini et al. (2010) (denoted hereafter as *Benevenuto* and *Stringhini* respectively) which represent subsets of the larger set of features. These were chosen based on self-reported performance, wide acceptance in the community, and reproducibility. The latter is defined by the possibility of reproducing the model with accessible account information and without the need for internal information such as IP addresses or the social graph[16].

Figure 8 shows cumulative distribution functions (CDFs) of the top 9 individually relevant features selected by the mutual information method. The curves distinguish three types of users: verified, humans and spammers. They confirm that verified users are indeed different from humans and sometimes exhibit behavior closer to spammers. Some of the cumulative distributions associated with spammers have more than one inflection point. This suggests that

[16]While it is certainly possible to use Twitter's Rest API to obtain a user's social graph, the imposed API rate limit makes it prohibitive and impractical to require this information in a large-scale model. Models using such information (e.g. Yang et al. (2011)) are hard to reproduce with a normal-level API access.

Table 3: Description of features used in this work

	Feature	Description	Our features	Benevenuto	Stringhini
Profile	age of the account	time since the account was created	✓	✓	
	statuses count	Total number of tweets posted by the account	✓		✓
S. Network	followers count	Number of users following the account	✓	✓	
	friends count	Number of users followed by the account	✓	✓	✓
	followers per followees	Ratio of followers to followees (friends)	✓	✓	
	followees per squared followers	Ratio of friends per the squared number of followers			✓
Content	replicates	Number of identical posts in the account timeline	✓		
	similarity	Average similarity between the account's posts	✓		✓
	fraction of tweets with urls	Fraction of tweets containing a link	✓	✓	✓
	fraction of tweets with hashtags	Fraction of tweets containing a hashtag	✓		
	fraction of replies	Fraction of replies in the user's posts	✓	✓	
	fraction of retweets	Fraction of retweets in the user's posts	✓		
	mean nb hashtags per tweet	Average number of hashtags in a tweet	✓	✓	
	mean nb urls per tweet	Average number of links in a tweet	✓	✓	
	urls used on average	Average number of times a single link is used	✓		
Behavioral	avg intertweet interval	Average time interval between consecutive tweets	✓		
	std intertweet interval	Standard deviation of time intervals between consecutive tweets	✓		
	min intertweet interval	Minimum time interval between two consecutive tweets	✓		
	nb followees per day	Average number of followees the account follows in a day	✓		
	nb followers per day	Average number of users following the account in a day	✓		
	active tweeting frequency per day	Average number of tweets posted by the account on a daily basis (computed based on the 200 most recent tweets)	✓		
	distribution of tweets in temporal bins	8 features corresponding to the proportion of the account's tweets contained in temporal bins of 3 hours each.	✓		

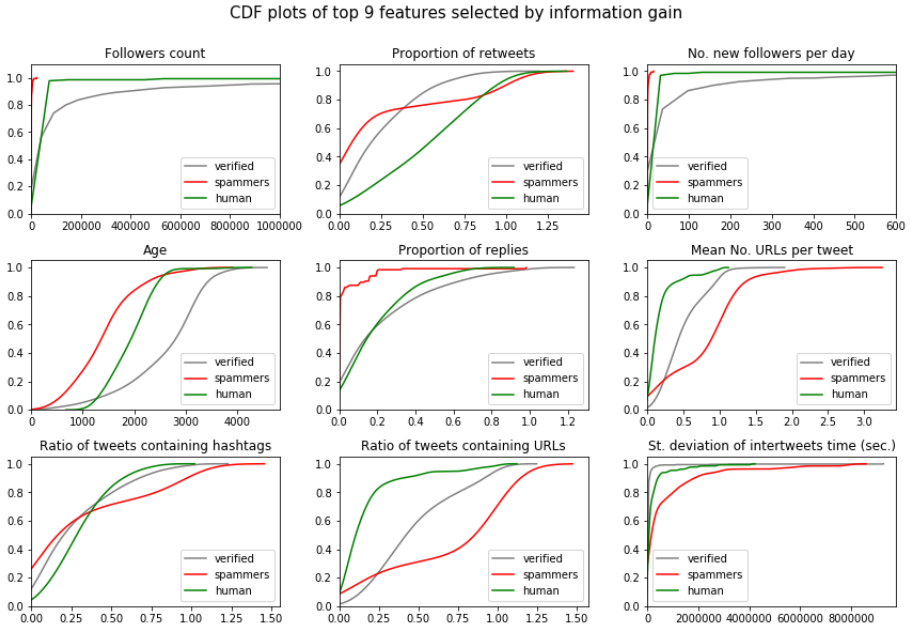

Figure 8: The CDF plots of the top 9 relevant features in the ground-truth dataset as selected by the mutual information method.

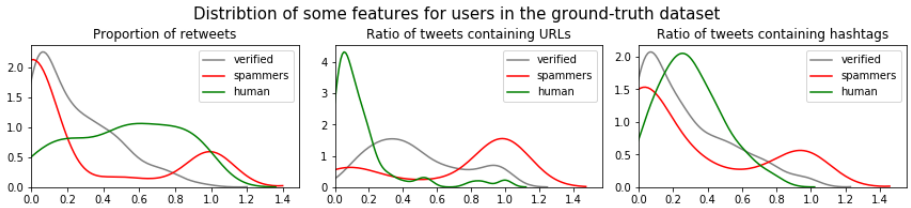

Figure 9: Distribution plots of features showacasing a dual spammer behavior.

the spammers population has a dual distribution with respect to these features (e.g. proportion of retweets, ratio of tweets containing URLs). This empirical observation generally supports the proposition that spammers are not a homogeneous population. The distribution of some of these features are shown in Figure 9. These show that spammers' behavior can closely mimic that of verified or human users.

7.1.2. Choosing a suitable classification model

We train and evaluate the described set of features using the following discriminative models: Support Vector Machines (SVM) Hearst et al. (1998), Logistic Regression (LR) Kleinbaum and Klein (2010) and Random Forests (RF) Breiman (2001). We do not to use generative learning models (e.g. Naive Bayes) to avoid learning a joint distribution $p(x, y)$ over the input and output spaces. This is done because the collection of Twitter ground-truth datasets introduces a selection bias and the resulting dataset does not offer a true distribution $p(x)$ over the input space[17]. We only want therefore to learn the conditional probability $p(y|x)$. The probabilities $p(y_u|\mathbf{x}_u)$ predicted by the classification models are used as priors in our MRF model[18]. Note that, although deep neural networks can also be used as a local supervised classifier, the limited size of the available datasets does not allow an effective deep implementation.

7.1.3. Implementation

We used the *sickit-learn* library Pedregosa et al. (2011) in Python to train the SVM, Logistic Regression (LR) and Random Forests (RF) classifiers. The SVM classifier used the RBF kernel, and its parameters C and γ were obtained using a grid search[19]. For LR, we compared results with $L1$ and $L2$ regularization.

[17]The selection bias introduced by the methods used to collect labeled instances is more amply discussed in El-Mawass and Alaboodi (2017).

[18]In our implementation, we used the *predict_proba* function of the *scikit-learn* Python package to compute the probability each supervised model assigns to its prediction.

[19]The values for C and γ obtained via cross-validation are $(10^3, 10^{-2})$ for our features, $(10^2, 10^{-1})$ for Benevenuto features, and $(10^3, 10^{-3})$ for Stringhini features.

Table 4: Most used applications in the groundtruth dataset (in terms of the number of unique messages.)

Application	**No. messages**	**%**	**Application type**
Twitter for iPhone	71,755	39.1	Twitter affiliated
Twitter Web Client	27,397	14.9	Twitter affiliated
Twitter for Android	17,866	9.7	Twitter affiliated
TweetDeck	11,198	6.1	Content management (Twitter affiliated)
Done For You Traffic	7,963	4.3	Content management
dlvr.it	7,079	3.9	Content management
IFTTT	6,343	3.5	Content management
Hootsuite	4,994	2.7	Content management
Google	3,572	1.9	Content Referral
Facebook	2,204	1.2	Content Referral

We evaluated and compared the classifiers over the three previously discussed sets of features, namely our selected set of state-of-the-art features and the sets of features proposed in Benevenuto and Stringhini. All features were normalized before training.

7.2. Constructing the Similarity Graph

We implemented the revised version of the graph construction algorithm by defining messages as a tuple that contains both the processed text and the source application (Table 4 lists the top 10 applications generating 87% of the messages in the ground-truth dataset). We then filtered edges in the resulting users graph according to the pairwise similarity of applications profiles. We removed edges that have a content weight of one (one common tweet) or an application similarity rate of less than 0.9. This choice is taken so that an edge represents real complicity between users. It also decreases the probability of linking two users based on text that is falsely identified as similar.

The resulting similarity graph is a sparse graph with 157 nodes and 549 edges.Edges represent 4.5% of the number of edges in a fully connected graph

Table 5: Processed texts and applications of the top five tweets in the dataset (in terms of number of users sharing the tweet).

Processed text message	**Application**	**No. users**
"<url>the daunting risks of laparoscopic obesity surgery"	Done For Your Traffic	24
(translation) "<mention>: to al nasr fans [sports team], I was honored this evening to be one of the world championship players..."	Twitter for iPhone	7
"<url>15 reasons to join affiliate programs"	Done For Your Traffic	7
"<url>8 ways to improve your affiliate marketing strategies"	Done For Your Traffic	7
"<url>finding the perfect product at clickbank"	Done For Your Traffic	7

with 157 nodes.

The similarity graph in Figure 10 illustrates the labels homophily captured by the similarity measure. It shows that linked users generally belong to the same class. Legitimate users and spammers also tend to form their own respective clusters. Among the 30 identified clusters, only 2 contain spammers and legitimate nodes simultaneously. This validates that the proposed similarity measure does indeed result in users of the same class being linked together. The high modularity (0.873) and average clustering coefficient (0.795) of the graph also demonstrate that users tend to cluster in communities of mutually similar users that are quite distinct and disassociated from the rest of the graph. Moreover, the graph clearly shows that the assumption that spammers form one connected community, which forms the basis of many previous works, does not hold.

7.3. MRF Classification

We implemented MRF using the *UGM* library Schmidt (2007) in Matlab. For inference, we used the library's implementation of LBP. Starting from the priors computed on each node, we applied LBP over the graph and updated classes beliefs according to the defined edge potentials. The MRF predictions are associated with posterior prediction probabilities obtained on LBP con-

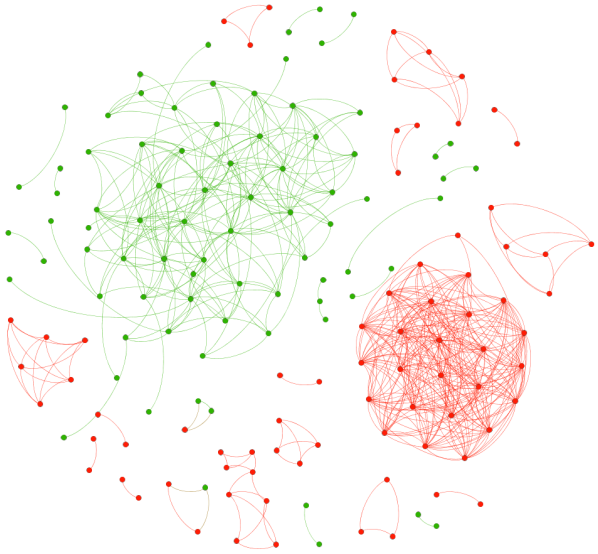

Figure 10: The similarity graph of connected users in the ground-truth dataset. Legitimate users are represented in green while spammers are shown in red.

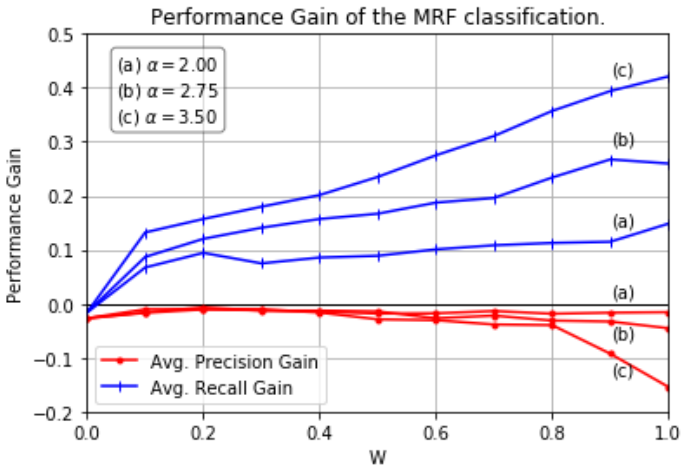

Figure 11: Performance gain of the MRF classification as a function of edge potentials for α ranging from 2 to 3.5.

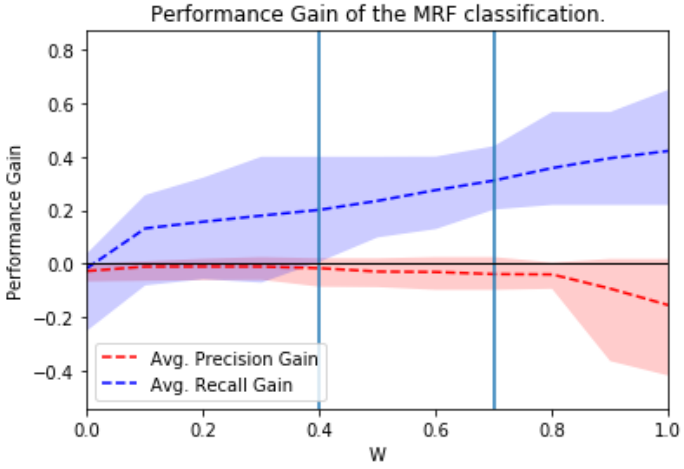

Figure 12: Average absolute gain in performance as a function of edge potentials for $\alpha = 3.5$. Upper and lower limits correspond to the maximum and minimum gain at each value of w.

vergence. Note that the comparison between the MRF performance and the performance of baseline classifiers is only meaningful for connected nodes. The results are thus obtained and compared over connected nodes only. In the case of a singleton (a node that is not connected to other nodes), the node does not have an associated edge potential. Its prediction probability is therefore solely governed by the prior predicted by the baseline classifier. The output of the MRF model for singletons is thus equivalent to the output of the traditional classifier.

7.4. Results

Classification results of the baseline supervised classifiers, symmetric and asymmetric MRF models are shown in Table 6. The highest values for each metric are highlighted in bold. The performance is evaluated in terms of accuracy, precision, recall and F1-measure, where the F1-measure is defined as follows:

$F1\text{-}measure = 2 \times Precision \times Recall/(Precision + Recall)$.

Results demonstrate that traditional classifiers have a high precision but a

Table 6: Classification performance, evaluated over the test dataset, of the baseline supervised classifiers, the symmetric MRF classifier ($e^{w_0} = e^{w_3} = 0.9$ and $e^{w_1} = e^{w_2} = 0.1$) and the asymmetric MRF classifier ($w = 0.6, \alpha = 2.5$).

		All features			**Benevenuto features**			**Stringhini features**		
		Sup.	Sym. MRF	Asym. MRF	Sup.	Sym. MRF	Asym. MRF	Sup.	Sym. MRF	Asym. MRF
SVM	Precision	0.891	0.917	**0.919**	0.939	**0.955**	0.930	0.941	**1.0**	0.966
	Recall	0.598	0.571	**0.883**	0.756	0.273	**0.857**	0.195	0.143	**0.364**
	F1	0.715	0.704	**0.901**	0.838	0.424	**0.892**	0.323	0.25	**0.528**
	Accuracy	0.752	0.764	**0.904**	0.847	0.637	**0.898**	0.573	0.58	**0.682**
LR L1	Precision	0.865	0.86	**0.89**	0.961	**1.0**	0.924	**1.0**	**1.0**	**1.0**
	Recall	0.549	0.558	**0.844**	0.598	0.545	**0.792**	0.159	0.325	**0.506**
	F1	0.672	0.677	**0.867**	0.737	0.706	**0.853**	0.274	0.49	**0.672**
	Accuracy	0.72	0.739	**0.873**	0.777	0.777	**0.866**	0.561	0.669	**0.758**
LR L2	Precision	**0.956**	0.9	0.933	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
	Recall	0.524	0.468	**0.727**	0.317	0.182	**0.714**	0.159	0.325	**0.481**
	F1	0.677	0.615	**0.818**	0.481	0.308	**0.833**	0.274	0.49	**0.649**
	Accuracy	0.739	0.713	**0.841**	0.643	0.599	**0.86**	0.561	0.669	**0.745**
RF	Precision	**0.955**	0.924	0.902	**1.0**	0.933	0.928	0.96	**1.0**	0.925
	Recall	0.78	0.792	**0.961**	0.585	0.727	**0.831**	0.585	0.532	**0.805**
	F1	0.859	0.853	**0.931**	0.738	0.818	**0.877**	0.727	0.695	**0.861**
	Accuracy	0.866	0.866	**0.930**	0.783	0.841	**0.885**	0.771	0.771	**0.873**

generally low recall[20]. Although the models have the advantage of being trained on the ground-truth accounts we recently collected, the obtained performance is significantly lower than the performance reported in the works in which these models were originally presented. This is especially the case for features proposed in Stringhini et al. (2010) that have a very low recall of contemporary spammers.Since features are still able to identify part of the spammers population with a relatively high precision, it can be argued that the deterioration in recall is due to some spammers succeeding at evading detection, thus driving the recall down. The limited efficiency of these features in keeping up with spam evolution validate the need to explore alternative approaches to traditional

[20]Supervised classifiers outperform a random classifier that classifies 85% of the population as legitimate. On the test dataset, a random classifier has a 52% precision, a 15% recall, a 6% f1-measure and a 48.6% accuracy. The 85% rate is based on estimates of the ratio of spammers to the total population of social accounts on Twitter (ranging between 4% in Benevenuto et al. (2010) and 15% in Varol et al. (2017)).

supervised classification.

As for MRF results, the average gain in performance (recall and precision) achieved by the asymmetric MRF classification compared to the baseline local classifiers is shown in Figure 11 for α ranging between 2 and 3.5. Gain is defined as the absolute change between the MRF performance and the performance of the local classifier on which it is based. The evolution of performance is computed as a function of edge potentials ($w_0 = w$ and $w_3 = \alpha \times w$). The figure clearly shows that MRF classification consistently increases the recall while maintaining precision around its baseline level.

Figure 12 shows the performance gain for $\alpha = 3.5$. Note that the best performance is obtained by setting w between 0.4 and 0.7. This yields a positive increase in recall (20 to 27% on average) while maintaining original precision (average decrease of 1.6 to 3%).

The reported values of α and w are in agreement with the empirical characteristics of the dataset for several reasons:

- First, it is to be expected that w_0 and w_3 (strength of connection between legitimate users and spammers respectively) should be bigger than w_1 (a spammer to legitimate user connection): users having the same class are more likely to be connected than users of different classes (only 4 edges in the similarity graph are inter-classes edges).

- Second, two observations confirm that w_3 (spammer to spammer connection) is expected to be higher than w_0 (legitimate to legitimate connection): (1) Spammers are more densely connected than legitimate users and (2) classifiers are always more confident about their spam labels (see the precision of the spam class in Table 6) making a spam label prediction more trustworthy than a legitimate label prediction.

7.5. Discussion and Generalization Insights

We present in this section the generalization insights gained from the results of the model above and discuss practical implementation issues that are faced

in a large-scale implementation.

7.5.1. Modularity vs. Homophily

For the MRF to work, edges should generally indicate a relationship of homophily: connected nodes have the same class. This assumption is vital to ensure that belief can be propagated on the users similarity graph. As discussed in section 4, the graph construction mechanism is tuned so that the number of edges connecting accounts from opposite classes is minimized. Although the resulting graph is modular, modularity in itself is not seeked: if two spam (resp. legitimate) clusters become connected through an edge, the model will become even more certain about its posterior predictions. In this case, an edge offers additional information.

The opposite case is problematic. If a legitimate account or cluster of accounts become connected to a spam cluster, belief will be propagated between the two clusters, leading to a decreased certainty in both the spam and legitimate class predictions. It is therefore important for the graph construction mechanism to result in homophilic edges and for the similarity measure to connect accounts having the same class.

A practical obstacle that would be faced in the case of a large-scale implementation is represented by "quotes apps". These are applications the generate automated sayings and posts and are generally subscribed for by both legitimate and spam accounts. The latter benefit from these applications in keeping their accounts active and posting. Tweets posted by these applications on behalf of subscribing accounts will inevitably result in edges created between spam and legitimate users. A simple solution would be to filter posts generated by these applications. This is feasible as these applications have a large throughput and are usually easy to identify when aggregating content from a large collection of users.

7.5.2. Time Complexity of Graph Construction

We have discussed in section 4 the time complexity of of generating users edges from a list of users associated with a post. In practice, there are two facets associated with the computational load of generating the graph:

- Quantifying the computational load of creating a similarity graph.
- Fixing a threshold to the number of users associated with a post (a unit of content).

Computational load of generating a similarity graph. We start by answering the first part: assessing the time required to build a similarity graph. For that we need two information:

- An empirical distribution of the number of users per post (text/unit of content).
- An empirical estimation of the time required to expand a list of users as a function of the number of users in the list.

Recall that expanding a list of users refers to creating an edge between all pairs of users. The complexity of this is $O(n^2)$, where n is the number of users in the list. To assess the expansion time, we conducted a simulation on an intel i7 machine with a 2.5 GHz clock and 16 GB RAM machine and measured the time required to list all edges associated with a given number of users. We used a user ID format comparable to that of Twitter to mimic realistic memory usage. The results are shown in Figure 13 and confirm that the time increases exponentially with the number of users. Figure 14 shows the evolution of the logarithmic time as a function of the number of edges. It further confirms that the time is proportional to the squared number of users. The processing time reaches one minute around 10000 users (or 10^8 edges). This means that it becomes 10 minutes for $31k$ users and 100 minutes for $100k$ users. Note that for expanding less than 1000 users, the processing time is negligible (< 1 second).

To assess the time required to build the similarity graph, we also need to assess the empirical distribution of the number of users to expand per post.

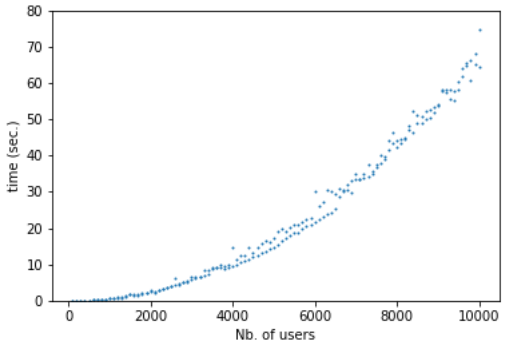

Figure 13: Time required to expand a list of users into edges.

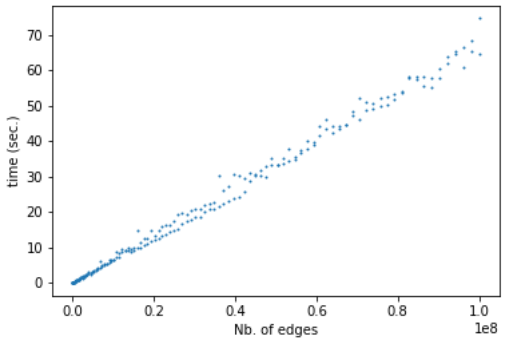

Figure 14: Time required to generate users edges.

For that, we consider the real case scenario where our system is used to detect spammers in trending topics. Trending topics are typically related to current and controversial topics and events and are characterized by a large audience. This makes them a particularly interesting target for spammers and opinion manipulators. We consider a dataset of trending topics we previously collected. The dataset covers trending topics in KSA in the period between 19/3/2015 and 1/4/2015, and contains $1,124,926$ unique tweets.

We construct the bipartite users-content graph and plot the distribution of users per text (processed post) for each trending topic as shown for example in Figure 15. All the distributions for the studied trending topics represent a similar pattern where every post (unit of content) is predominantly shared by a small number of users. This means that the associated users edges should be obtained in milliseconds. Only a few texts are shared by thousands of users. Specifically, a typical trending topic does not have more than 5 messages counting more than one thousand users. The most shared text in the studied dataset has been shared by 14k users and associated edges should therefore be produced in around one minute.

Note that the construction of the graph is completely parallelizable in that every list of users (associated with a given post) can be assigned to a different machine core or mapper (in a MapReduce framework). The total time is therefore bounded by the time required to process the largest list of users. Even when the processing is serialized, the total time is dominated by the same value as the distribution is usually biased towards less popular posts.

Choosing a threshold. The question of where the threshold should be placed depends on the application and the available processing power. A threshold is the number of users beyond which a list of users is not expanded (not transformed into edges). We empirically assessed the upper bounds of content-based aggregation of users in trending topics and deduced that they can be processed in reasonable time. By interpolating the plot obtained in Figure 14, we can safely consider that a threshold of around $30k$ users is a reasonable restriction.

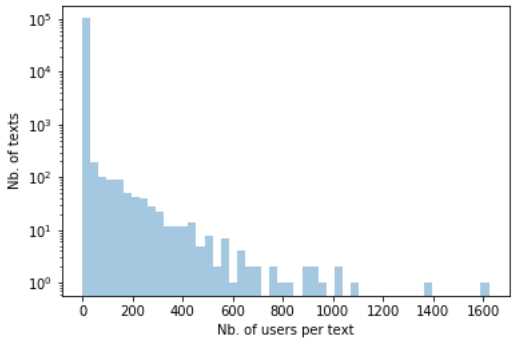

Figure 15: Histogram of the distribution of identical texts in a trending topic.

Posts with this degree of popularity are usually initiated by celebrity profiles, and we can safely consider that a (text, application) tuple having $30k$ accounts is an indication of an organic legitimate sharing activity.

7.5.3. Effect of baseline recall and precision

All compared baseline classifiers have a relatively high precision and can therefore be reliably used to detected seeds of spam accounts. Table 6 shows that even when baseline recall is lower than 50% (Stringhini features), beliefs can be effectively propagated, and the MRF model can increase the recall while maintaining precision. This can be explained by two reasons. The first is that the edge potentials matrix favors spammer-spammer edges. When a spammer is identified, connections are more likely to be predicted by the model as spammers. The second reason is that accounts features are not randomly distributed among users. Connected users are more likely to have similar features and therefore to have similar prior predictions. In other terms, when seeds are discovered, they are more likely to be clustered together than to be randomly distributed among spammers clusters. Thus, even when the baseline recall is low, the concentration of seeds in particular clusters ensure that other spammers in those clusters will be correctly identified.

7.5.4. Role of the edge potentials matrix

MRF is a generative model. The potentials can be used to quantify the likelihood of incidence of a particular edge configuration. Higher values of w would therefore indicate that a spammer-spammer edge is much more likely than other configurations. This blocks belief propagation across the graph as inference becomes dominated by the edge potential. Figure 11 shows a general trend of decreasing precision when w reaches higher values. This can be explained by the edge potential becoming significantly higher than the node potentials[21]. Thus, the model becomes mostly equivalent to an MRF with no observations on the nodes and assumes that most edges are statistically associated with spammers. This is similar to the case where a traditional classifier assumes that all or the majority of classified instances belong to a certain class, resulting in a perfect recall and a low precision.

Spam clusters are typically denser and are therefore associated with more edges. This makes asymmetric edge potentials matrices better at capturing the distribution of edges in a users graph. However, Lower values of w should be preferred to avoid the scenario discussed above.

Finally, results confirm that similarity can be used to improve the performance of a weak local classifier. The local information synthesized as a belief can be propagated throughout the graph to correct misclassified instances and mitigate the effect of spam evolution. Compared to local classifiers, our model consistently improves recall over several sets of features, and generally maintains high precision. The notion of a weak local classifier is therefore effectively exploitable in the context of probabilistic graphical inference.

8. Conclusion and Future Work

In this paper, we tackle the problem of the deteriorating classification performance of state-of-the-art supervised classifiers by proposing a system based

[21] For $w = 1$ and $\alpha = 3.5$: $\phi_u \in [0, 1]$, $\phi_{u,v}(1, 1) \approx 33$, $\phi_u \ll \phi_{u,v}(1, 1)$

on the Markov Random Field framework. We propose a solution where the predictions of supervised classifiers are considered prior beliefs on the classes of social users. We then propagate these beliefs on a users graph to obtain more accurate posterior beliefs. To construct a homophilic users graph, we define similarity based on an interaction graph instead of the commonly used social graph. We evaluate this hybrid features/graph framework on a ground-truth Twitter dataset. The performance deterioration of state-of-the-art supervised classifiers evaluated on this dataset corroborates similar results reported in the literature. The implementation of the proposed system on this dataset validates that it is indeed possible to exploit accounts similarity in a probabilistic framework. The Markov Random Field framework restores classification performance of the mentioned classifiers by increasing recall while simultaneously maintaining precision.

As a future work, we would like to bypass some of the limitations of the current work. Specifically, we would like to explore forms of graphical models that are more expressive than the commonly used but relatively rigid representation of the Markov Random Field. Several aspects of similarity for example can be more accurately represented through conditional edge potentials. This may result in a model that is more accurate, more expressive and less data-dependent. Additionally, we would like to assess the implications of the obtained results on a full-fledged in-the-wild application. This could represent an interesting and potentially useful venue for future exploration.

References

Aggarwal, A., Rajadesingan, A., Kumaraguru, P., 2012. PhishAri: Automatic realtime phishing detection on twitter, in: eCrime Researchers Summit (eCrime), 2012, IEEE. pp. 1–12. URL: https://ieeexplore.ieee.org/abstract/document/6489521.

Akoglu, L., Chandy, R., Faloutsos, C., 2013. Opinion Fraud Detection in Online

Reviews by Network Effects., in: proceedings of the international conference on weblogs and social media ICWSM, pp. 2–11.

Barreno, M., Nelson, B., Sears, R., Joseph, A.D., Tygar, J.D., 2006. Can machine learning be secure?, in: Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security, ACM, New York, NY, USA. pp. 16–25.

Bastos, M.T., Mercea, D., 2019. The brexit botnet and user-generated hyperpartisan news. Social Science Computer Review 37, 38–54.

Benevenuto, F., Magno, G., Rodrigues, T., Almeida, V., 2010. Detecting spammers on twitter, in: Collaboration, electronic messaging, anti-abuse and spam conference (CEAS), p. 12.

Beutel, A., Xu, W., Guruswami, V., Palow, C., Faloutsos, C., 2013. CopyCatch: stopping group attacks by spotting lockstep behavior in social networks, in: Proceedings of the 22nd international conference on World Wide Web, International World Wide Web Conferences Steering Committee. pp. 119–130.

Bhat, S.Y., Abulaish, M., 2013. Community-based features for identifying spammers in online social networks, in: Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining - ASONAM '13, pp. 100–107.

Bondielli, A., Marcelloni, F., 2019. A survey on fake news and rumour detection techniques. Information Sciences 497, 38 – 55. URL: http://www.sciencedirect.com/science/article/pii/S0020025519304372.

Breiman, L., 2001. Random forests. Machine learning 45, 5–32.

Cao, Q., Yang, X., Yu, J., Palow, C., 2014. Uncovering Large Groups of Active Malicious Accounts in Online Social Networks, in: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security - CCS '14, ACM Press, New York, New York, USA. pp. 477–488. URL: http://dl.acm.org/citation.cfm?id=2660267.2660269.

Chau, D.H.P., Nachenberg, C., Wilhelm, J., Wright, A., Faloutsos, C., 2011. Polonium: Tera-scale graph mining and inference for malware detection, in: Proceedings of the 2011 SIAM International Conference on Data Mining, SIAM. pp. 131–142.

Chhabra, S., Aggarwal, A., Benevenuto, F., Kumaraguru, P., 2011. Phi.sh/$oCiaL: The Phishing Landscape through Short URLs, in: Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference on - CEAS '11, ACM Press. pp. 92–101. URL: http://dl.acm.org/citation.cfm?id=2030376.2030387.

Chu, Z., Gianvecchio, S., Wang, H., Jajodia, S., 2010. Who is tweeting on twitter: Human, bot, or cyborg?, in: Proceedings of the 26th Annual Computer Security Applications Conference, ACM, New York, NY, USA. pp. 21–30.

Clark, E.M., Williams, J.R., Jones, C.A., Galbraith, R.A., Danforth, C.M., Dodds, P.S., 2016. Sifting robotic from organic text: A natural language approach for detecting automation on twitter. Journal of Computational Science 16, 1 – 7. URL: http://www.sciencedirect.com/science/article/pii/S1877750315300363.

Cresci, S., Di Pietro, R., Petrocchi, M., Spognardi, A., Tesconi, M., 2015. Fame for sale: Efficient detection of fake twitter followers. Decision Support Systems 80, 56–71.

Cresci, S., Di Pietro, R., Petrocchi, M., Spognardi, A., Tesconi, M., 2016. DNA-inspired online behavioral modeling and its application to spambot detection. IEEE Intelligent Systems 31, 58–64.

Cresci, S., Di Pietro, R., Petrocchi, M., Spognardi, A., Tesconi, M., 2017a. The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race, in: Proceedings of the 26th international conference on World Wide Web companion, International World Wide Web Conferences Steering Committee. pp. 963–972.

Cresci, S., Di Pietro, R., Petrocchi, M., Spognardi, A., Tesconi, M., 2017b. Social fingerprinting: detection of spambot groups through dna-inspired behavioral modeling. IEEE Transactions on Dependable and Secure Computing 15, 561–576.

Cresci, S., Petrocchi, M., Spognardi, A., Tognazzi, S., 2018. From reaction to proaction: Unexplored ways to the detection of evolving spambots., in: WWW (Companion Volume), pp. 1469–1470.

Cresci, S., Petrocchi, M., Spognardi, A., Tognazzi, S., 2019a. Better safe than sorry: An adversarial approach to improve social bot detection, in: Proceedings of the 11th ACM Conference on Web Science, WebSci 2019, Boston, MA, USA. pp. 47–56.

Cresci, S., Petrocchi, M., Spognardi, A., Tognazzi, S., 2019b. On the capability of evolved spambots to evade detection via genetic engineering. Online Social Networks and Media 9, 1 – 16. URL: http://www.sciencedirect.com/science/article/pii/S246869641830065X.

Danezis, G., Mittal, P., 2009. SybilInfer : Detecting Sybil Nodes using Social Networks, in: Network & Distributed System Security Symposium (NDSS).

Egele, M., Stringhini, G., Kruegel, C., Vigna, G., 2013. COMPA: Detecting Compromised Accounts on Social Networks., in: Network & Distributed System Security Symposium (NDSS).

El-Mawass, N., Alaboodi, S., 2016. Detecting Arabic Spammers and Content Polluters on Twitter, in: 6th International Conference on Digital Information Processing and Communications (ICDIPC'16), IEEE, Beirut, Lebanon.

El-Mawass, N., Alaboodi, S., 2017. Data quality challenges in social spam research. J. Data and Information Quality 9, 4:1–4:4.

El-Mawass, N., Honeine, P., Vercouter, L., 2018. Supervised Classification of Social Spammers using a Similarity-based Markov Random Field Approach,

in: Proceedings of the 5th Multidisciplinary International Social Networks Conference on - MISNC '18, ACM Press, Saint-Etienne, France. pp. 1–8.

Ferrara, E., 2017. Disinformation and Social Bot Operations in the Run Up to the 2017 French Presidential Election, in: SSRN Electronic Journal. URL: https://www.ssrn.com/abstract=2995809.

Freeman, D.M., 2017. Can You Spot the Fakes?: On the Limitations of User Feedback in Online Social Networks, in: Proceedings of the 26th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee. pp. 1093–1102.

Freeman, W.T., Pasztor, E.C., Carmichael, O.T., 2000. Learning low-level vision. International journal of computer vision 40, 25–47.

Gao, P., Gong, N.Z., Kulkarni, S., Thomas, K., Mittal, P., 2015. Sybilframe: A defense-in-depth framework for structure-based sybil detection. arXiv preprint arXiv:1503.02985 , 17URL: http://arxiv.org/abs/1503.02985, arXiv:1503.02985.

Ghosh, S., Viswanath, B., Kooti, F., Sharma, N.K., Korlam, G., Benevenuto, F., Ganguly, N., Gummadi, K.P., 2012. Understanding and combating link farming in the twitter social network, in: Proceedings of the 21st international conference on World Wide Web - WWW '12, ACM Press, New York, New York, USA. p. 61. URL: http://dl.acm.org/citation.cfm?id=2187836.2187846.

Gong, N.Z., Frank, M., Mittal, P., 2014. SybilBelief: A Semi-Supervised Learning Approach for Structure-Based Sybil Detection. IEEE Transactions on Information Forensics and Security 9, 976–987. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6787042.

Hearst, M.A., Dumais, S., Osuna, E., Platt, J., Scholkopf, B., 1998. Support vector machines. volume 13. IEEE. URL: http://ieeexplore.ieee.org/document/708428/.

Hooi, B., Shin, K., Song, H.A., Beutel, A., Shah, N., Faloutsos, C., 2017. Graph-based fraud detection in the face of camouflage. ACM Transactions on Knowledge Discovery from Data 11, 44:1–44:26.

Hooi, B., Song, H.A., Beutel, A., Shah, N., Shin, K., Faloutsos, C., 2016. Fraudar: Bounding graph fraud in the face of camouflage, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM. pp. 895–904. URL: https://www.kdd.org/kdd2016/subtopic/view/fraudar-bounding-graph-fraud-in-the-face-of-camouflage.

Inuwa-Dutse, I., Liptrott, M., Korkontzelos, I., 2018. Detection of spam-posting accounts on twitter. Neurocomputing 315, 496 – 511. URL: http://www.sciencedirect.com/science/article/pii/S0925231218308798.

Javed, A., Burnap, P., Rana, O., 2019. Prediction of drive-by download attacks on twitter. Information Processing & Management 56, 1133 – 1145. URL: http://www.sciencedirect.com/science/article/pii/S0306457317305824.

Jiang, M., Cui, P., Faloutsos, C., 2016. Suspicious behavior detection: Current trends and future directions. IEEE Intelligent Systems 31, 31–39. URL: https://ieeexplore.ieee.org/document/7389913.

Kleinbaum, D.G., Klein, M., 2010. Logistic regression: a self-learning text. Springer Science & Business Media.

Kušen, E., Strembeck, M., 2020. You talkin' to me? exploring human/bot communication patterns during riot events. Information Processing & Management 57, 102–126. URL: http://www.sciencedirect.com/science/article/pii/S0306457319305370.

Lee, K., Caverlee, J., Kamath, K.Y., Cheng, Z., 2012. Detecting collective attention spam, in: Proceedings of the 2nd Joint WICOW/AIRWeb Workshop

on Web Quality - WebQuality '12, ACM Press, New York, New York, USA. p. 48.

Lee, K., Caverlee, J., Webb, S., 2010. Uncovering social spammers: Social honeypots + machine learning, in: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, New York, NY, USA. pp. 435–442.

Lee, S., Kim, J., 2012. WarningBird: Detecting Suspicious URLs in Twitter Stream., in: Network & Distributed System Security Symposium (NDSS).

Li, Y., Martinez, O., Chen, X., Li, Y., Hopcroft, J.E., 2016. In a World That Counts: Clustering and Detecting Fake Social Engagement at Scale, in: Proceedings of the 25th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee.

Mazza, M., Cresci, S., Avvenuti, M., Quattrociocchi, W., Tesconi, M., 2019. Rtbust: Exploiting temporal patterns for botnet detection on twitter, in: Proceedings of the 10th ACM Conference on Web Science, ACM, New York, NY, USA. pp. 183–192.

McCord, M., Chuah, M., 2011. Spam detection on Twitter using traditional classifiers. Lecture Notes in Computer Science 6906 LNCS, 175–186. URL: https://link.springer.com/chapter/10.1007/978-3-642-23496-5_13.

Meel, P., Vishwakarma, D.K., 2019. Fake news, rumor, information pollution in social media and web: A contemporary survey of state-of-the-arts, challenges and opportunities, in: Expert Systems with Applications. Elsevier.

Murphy, K.P., Weiss, Y., Jordan, M.I., 1999. Loopy belief propagation for approximate inference: An empirical study, in: Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence, Morgan Kaufmann Publishers Inc.. pp. 467–475.

Pandit, S., Chau, D.H., Wang, S., Faloutsos, C., 2007. Netprobe: a fast and scalable system for fraud detection in online auction networks, in: Proceedings

of the 16th international conference on World Wide Web - WWW '07, ACM Press, Banff, Alberta, Canada. pp. 201–210.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12, 2825–2830.

Ratkiewicz, J., Conover, M.D., Meiss, M., Gonc, B., Flammini, A., Menczer, F., Gonçalves, B., Flammini, A., Menczer, F., 2011. Detecting and Tracking Political Abuse in Social Media., in: International Conference on Weblogs and Social Media ICWSM, pp. 297–304. URL: http://www.aaai.org/ocs/index.php/ICWSM/ICWSM11/paper/viewFile/2850/3274.

Rayana, S., Akoglu, L., 2015. Collective opinion spam detection: Bridging review networks and metadata, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM. pp. 985–994.

Schmidt, M., 2007. UGM: A Matlab toolbox for probabilistic undirected graphical models. URL: http://www.cs.ubc.ca/{~}schmidtm/Software/UGM.html.

Stringhini, G., Egele, M., Kruegel, C., Vigna, G., 2012. Poultry markets: on the underground economy of twitter followers, in: Proceedings of WOSN'12, pp. 1–6. URL: http://dl.acm.org/citation.cfm?id=2342551.

Stringhini, G., Kruegel, C., Vigna, G., 2010. Detecting spammers on social networks, in: Proceedings of the 26th Annual Computer Security Applications Conference, ACM, New York, NY, USA. pp. 1–9.

Stringhini, G., Wang, G., Egele, M., Kruegel, C., Vigna, G., Zheng, H., Zhao, B.Y., 2013. Follow the green: growth and dynamics in twitter follower mar-

kets, in: Proceedings of the 2013 conference on Internet measurement conference, pp. 163–176.

Thomas, K., Grier, C., Paxson, V., 2012. Adapting Social Spam Infrastructure for Political Censorship, in: 5th USENIX Workshop on Large-Scale Exploits and Emergent Threats. URL: https://www.usenix.org/conference/leet12/workshop-program/presentation/thomas.

Thomas, K., Grier, C., Song, D., Paxson, V., 2011. Suspended accounts in retrospect: an analysis of twitter spam, in: Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference, pp. 243–258.

Thomas, K., Paxson, V., Mccoy, D., Grier, C., 2013. Trafficking Fraudulent Accounts : The Role of the Underground Market in Twitter Spam and Abuse. USENIX Security Symposium , 195–210.

Varol, O., Ferrara, E., Davis, C.A., Menczer, F., Flammini, A., 2017. Online human-bot interactions: Detection, estimation, and characterization, in: Eleventh international AAAI conference on web and social media.

Wang, B., Zubiaga, A., Liakata, M., Procter, R., 2015. Making the Most of Tweet-Inherent Features for Social Spam Detection on Twitter, in: Proceedings of the 5th Workshop on Making Sense of Microposts Microposts2015.

Washha, M., Qaroush, A., Mezghani, M., Sedes, F., 2019. Unsupervised collective-based framework for dynamic retraining of supervised real-time spam tweets detection model. Expert Systems with Applications 135, 129 – 152. URL: http://www.sciencedirect.com/science/article/pii/S0957417419303872.

Yang, C., Harkreader, R.C., Gu, G., 2011. Die free or live hard? empirical evaluation and new design for fighting evolving twitter spammers, in: Recent Advances in Intrusion Detection, Springer. pp. 318–337. URL: https://link.springer.com/chapter/10.1007/978-3-642-23644-0_17.

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

Yardi, S., Romero, D., Schoenebeck, G., Others, 2009. Detecting spam in a twitter network, in: First Monday. volume 15. URL: https://firstmonday.org/article/view/2793/2431.

Yu, H., Gibbons, P.B., Kaminsky, M., Xiao, F., 2008. SybilLimit: A Near-Optimal Social Network Defense against Sybil Attacks, in: 2008 IEEE Symposium on Security and Privacy, IEEE. pp. 3–17. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4531141.

Yu, H., Kaminsky, M., Gibbons, P.B., Flaxman, A., 2006. Sybilguard: defending against sybil attacks via social networks, in: ACM SIGCOMM Computer Communication Review, ACM. pp. 267–278.

Yu, H., Kaminsky, M., Gibbons, P.B., Flaxman, A.D., 2008. Sybilguard: Defending against sybil attacks via social networks. IEEE/ACM Transactions on Networking 16, 576–589.

Zhang, X., Ghorbani, A.A., 2019. An overview of online fake news: Characterization, detection, and discussion, in: Information Processing & Management. URL: http://www.sciencedirect.com/science/article/pii/S0306457318306794.

Zhang, Y., Ruan, X., Wang, H., Wang, H., 2014. What scale of audience a campaign can reach in what price on Twitter? INFOCOM, 2014 Proceedings IEEE , 1168–1176URL: https://ieeexplore.ieee.org/abstract/document/6848048.