#DEBATENIGHT: The Role and Influence of Socialbots on Twitter During the 1st 2016 U.S. Presidential Debate

Marian-Andrei Rizoiu[12] and **Timothy Graham**[1] and **Rui Zhang**[12] and **Yifei Zhang**[12] and **Robert Ackland**[1] and **Lexing Xie**[12]
[1]The Australian National University, [2]Data61 CSIRO, Canberra, Australia.

Abstract

Serious concerns have been raised about the role of 'socialbots' in manipulating public opinion and influencing the outcome of elections by retweeting partisan content to increase its reach. Here we analyze the role and influence of socialbots on Twitter by determining how they contribute to retweet diffusions. We collect a large dataset of tweets during the 1st U.S. presidential debate in 2016 and we analyze its 1.5 million users from three perspectives: user influence, political behavior (partisanship and engagement) and botness. First, we define a measure of user influence based on the user's active contributions to information diffusions, i.e. their tweets and retweets. Given that Twitter does not expose the retweet structure – it associates all retweets with the original tweet – we model the latent diffusion structure using only tweet time and user features, and we implement a scalable novel approach to estimate influence over all possible unfoldings. Next, we use partisan hashtag analysis to quantify user political polarization and engagement. Finally, we use the BotOrNot API to measure user *botness* (the likelihood of being a bot). We build a two-dimensional "polarization map" that allows for a nuanced analysis of the interplay between botness, partisanship and influence. We find that not only are socialbots more active on Twitter – starting more retweet cascades and retweeting more – but they are 2.5 times more influential than humans, and more politically engaged. Moreover, pro-Republican bots are both more influential and more politically engaged than their pro-Democrat counterparts. However we caution against blanket statements that software designed to appear human dominates politics-related activity on Twitter. Firstly, it is known that accounts controlled by teams of humans (e.g. organizational accounts) are often identified as bots. Secondly, we find that many highly influential Twitter users are in fact pro-Democrat and that most pro-Republican users are mid-influential and likely to be human (low botness).

1 Introduction

Socialbots are broadly defined as "software processes that are programmed to appear to be human-generated within the context of social networking sites such as Facebook and Twitter" (Gehl and Bakardjieva 2016, p.2). They have recently attracted much attention and controversy, with concerns that they infiltrated political discourse during the 2016 U.S. presidential election and manipulated public opinion at scale. Concerns were heightened with the discovery that an influential conservative commentator (*@Jenn_Abrams*, 70,000 followers) and a user claiming to belong to the Tennessee Republican Party (*@TEN_GOP*, 136,000 followers) – both retweeted by high-profile political figures and celebrities – were in fact Russian-controlled bots operated by the Internet Research Agency in St. Petersburg (Collins and Cox 2017; Timberg, Dwoskin, and Entous 2017).

There are several challenges that arise when conducting large-scale empirical analysis of political influence of bots on Twitter. The first challenge concerns estimating user influence from retweet diffusions, where the retweet relations are unobserved – the Twitter API assigns every retweet to the original tweet in the diffusion. Current state-of-the-art influence estimation methods such as ConTinEst (Du et al. 2013) operate on a static snapshot of the diffusion graph, which needs to be inferred from retweet diffusions using approaches like NetRate (Rodriguez, Balduzzi, and Schölkopf 2011). This workflow suffers from two major drawbacks: first, the algorithms for uncovering the diffusion graph do not scale to millions of users like in our application; second, operating on the diffusion graph estimates the "potential of being influential", but it loses information about user activity – e.g. a less well connected user can still be influential if they tweet a lot. The question is **how to estimate at scale the influence of millions of users from diffusion in which the retweet relation is not observed?** The second challenge lies in determining at scale whether a user is a bot and also her political behavior, as manually labeling millions of users is infeasible. The question is therefore **how to leverage automated bot detection approaches to measure the botness of millions of users? and how to analyze political behavior (partisanship and engagement) at scale?**

This paper addresses the above challenges using a large dataset (hereafter referred to as #DEBATENIGHT) of 6.5 million tweets authored by 1.5 million users that was collected on 26 September 2016 during the 1st U.S. presidential debate.

To address the first challenge, we introduce, evaluate, and apply a novel algorithm to estimate user influence based on retweet diffusions. We model the latent diffusion structure using only time and user features by introducing the *diffusion scenario* – a possible unfolding of a diffusion – and its

likelihood. We implement a scalable algorithm to estimate user influence over all possible diffusion scenarios associated with a diffusion. We demonstrate that our algorithm can accurately recover the ground truth on a synthetic dataset. We also show that, unlike simpler alternative measures–such as the number of followers, or the mean size of initiated cascades–our influence measure φ assigns high scores to both highly-connected users who never start diffusions and to active retweeters with little followership.

We address the second challenge by proposing three new measures (political polarization $\mathcal{P}$, political engagement $\mathcal{E}$ and botness ζ) and by computing them for each user in #DEBATENIGHT. We manually compile a list of partisan hashtags and we estimate political engagement based on the tendency to use these hashtags and political polarization based on whether pro-Democrat or pro-Republican hashtags were predominantly used. We use the BotOrNot API to evaluate botness and to construct four reference populations – `Human`, `Protected`, `Suspended` and `Bot`. We build a two-dimensional visualization – the *polarization map* – that enables a nuanced analysis of the interplay between botness, partisanship and influence. We make several new and important findings: (1) bots are more likely to be pro-Republican; (2) bots are more engaged than humans, and pro-Republican bots are more engaged than pro-Democrat bots; (3) the average pro-Republican bot is twice as influential as the average pro-Democrat bot; (4) very highly influential users are more likely to be pro-Democrat; and (5) highly influential bots are mostly pro-Republican.

The main contributions of this work include:

- We introduce a **scalable algorithm to estimate user influence** over all possible unfoldings of retweet diffusions where the cascade structure is not observed; the code is publicly accessible in a Github repository[1];
- We develop two **new measures of political polarization and engagement** based on usage of partisan hashtags; the list of partisan hashtags is also available in the repository;
- We measure the **botness of a very large population of users** engaged in Twitter activity relating to an important political event – the 2016 U.S presidential debates;
- We propose the **polarization map** – a novel visualization of political polarization as a function of user influence and botness – and we use it to gain insights into the influence of bots on the information landscape around the U.S. presidential election.

2 Related work

We structure the discussion of previous work into two categories: related work on the estimation of user influence and work concerning bot presence and behavior on Twitter.

Estimating user influence on Twitter. Aggregate measures such as the follower count, the number of retweets and the number of mentions have been shown to be indicative of user influence on Twitter (Cha et al. 2010; Kwak et al. 2010). More sophisticated estimates of user influence use eigenvector centrality to account for the connectivity of followers or retweeters; for example, TwitteRank (Weng et al. 2010) extends PageRank (Page et al. 1999) by taking into account topical similarity between users and network structure. Other extensions like Temporal PageRank (Rozenshtein and Gionis 2016) explicitly incorporate time into ranking to account for a time-evolving network. However, one limitation of PageRank-based methods is that they require a complete mapping of the social networks. More fundamentally, network centrality has the drawback of evaluating only the *potential* of a user to be influential in spreading ideas or content, and it does not account for the actions of the user (e.g. tweeting on a particular subject). Our influence estimation approach proposed in Sec. 3 is built starting from the user Twitter activity and it does not require knowledge of the social network.

Recent work (Yates, Joselow, and Goharian 2016; Chikhaoui et al. 2017) has focused on estimating user influence as the contribution to information diffusion. For example, ConTinEst (Du et al. 2013) requires a complete diffusion graph and employs a random sampling algorithm to approximate user influence with scalable complexity. However, constructing the complete diffusion graph might prove problematic, as current state-of-the-art methods for uncovering the diffusion structure (e.g. (Rodriguez, Balduzzi, and Schölkopf 2011; Simma and Jordan 2010; Cho et al. 2013; Li and Zha 2013; Linderman and Adams 2014)) do not scale to the number of users in our dataset. This is because these methods assume that a large number of cascades occur in a rather small social neighborhood, whereas in #DEBATENIGHT cascades occur during a short period of time in a very large population of users. Our proposed algorithm estimates influence directly from retweet cascades, without the need to reconstruct the retweet graph, and it scales cubically with the number of users.

Bot presence and behavior on Twitter. The 'BotOrNot' Twitter bot detection API uses a Random Forest supervised machine learning classifier to calculate the likelihood of a given Twitter user being a bot, based on more than 1,000 features extracted from meta-data, patterns of activity, and tweet content (grouped into six main classes: user-based; friends; network; temporal; content and language; and sentiment) (Davis et al. 2016; Varol et al. 2017)[2]. The bot scores are in the range $[0, 1]$, where 0 (1) means the user is very unlikely (likely) to be a bot. BotOrNot was used to examine how socialbots affected political discussions on Twitter during the 2016 U.S. presidential election (Bessi and Ferrara 2016). They found that bots accounted for approximately 15 % (400,000 accounts) of the Twitter population involved in election-related activity, and authored about 3.8 million (19 %) tweets. However, Bessi and Ferrara (2016) sampled the most active accounts, which could bias upwards their estimate of the presence of bots as activity volume is one of the features that is used by BotOrNot. They found that bots were just as effective as humans at *attracting retweets* from humans. Woolley and Guilbeault (2017) used BotOrNot to

[1]Code and partisan hashtag list is publicly available at `https://github.com/computationalmedia/cascade-influence`

[2]See: `https://botometer.iuni.iu.edu/#!/`

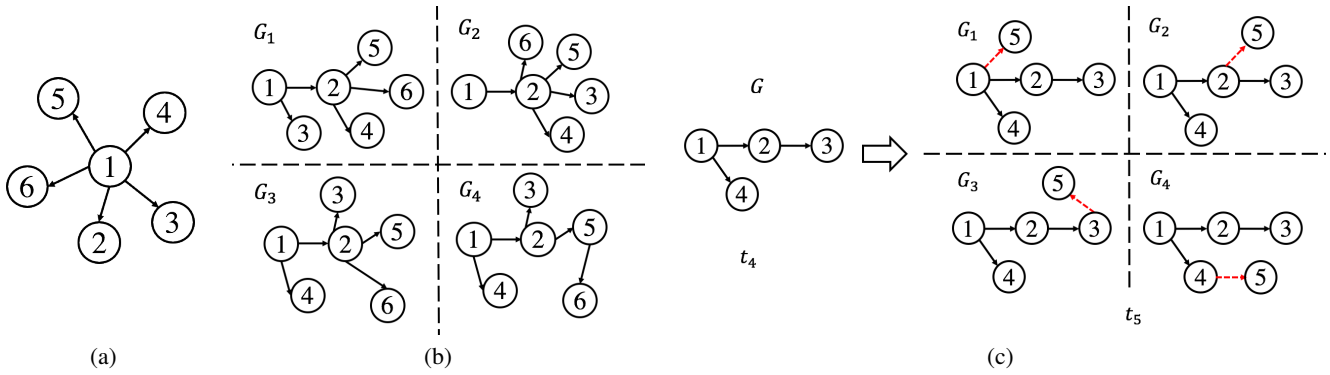


Figure 1: Modeling latent diffusions. **(a)** The schema of a retweet cascade as provided by the Twitter API, in which all retweets are attributed to the original tweet. **(b)** Four diffusion scenarios (out of 120 possible scenarios), associated with the retweet cascade in (a). **(c)** Intuition of the independent conditional model. A new node v_5 appears conditioned on one diffusion scenario G. Four new diffusion scenarios are generated as v_5 can attach to any of the existing nodes.

test 157,504 users randomly sampled from 1,798,127 Twitter users participating in election-related activity and found that over 10% were bots. Here we use BotOrNot to classify *all* 1.5 million users in our dataset to obtain a less biased approximation of their numbers and impact.

Previous work has studied the political partisanship of Twitter bots. Kollanyi, Howard, and Woolley (2016) analyzed candidate-oriented hashtag use during the 1st U.S. presidential debate and found that highly automated accounts (self-identified bots and/or accounts that post at least 50 times a day) were disproportionately pro-Trump. Bessi and Ferrara (2016) also studied political partisanship by identifying five pro-Trump and four pro-Clinton hashtags and assigning users to a particular political faction. The results suggested that both humans and bots were more pro-Trump in terms of hashtag partisanship. However, the above findings are limited to a comparison between humans and bots of frequency counts of tweets authored and retweets received, and they provide no insight into the importance of users in retweet diffusions. We overcome this limitation by modeling the latent structure of retweet diffusions and computing user influence over all possible scenarios.

3 Estimating influence in retweet cascades

An *information cascade* V of size n is defined as a series of messages v_i sent by user u_i at time t_i, i.e. $V = \{v_i = (u_i, t_i)\}_{i=1:n}$. Here $v_1 = (u_1, t_1)$ is the initial message, and $v_1, \ldots, v_n$ with $t_1 < \ldots < t_n$ are subsequent reposts or relays of the initial message. In the context of Twitter, the initial message is an original tweet and the subsequent messages are retweets of that original tweet (which by definition, are also tweets). A latent retweet diffusion graph $G = (V, E)$ has the set of tweets as its vertexes V, and additional edges $E = \{(v_i, v_j)\}$ that represent that the j^{th} tweet is a retweet of the i^{th} tweet, and respects the temporal precedence $t_i < t_j$. Web data sources such as the Twitter API provide cascades, but not the diffusion edges. Such missing data makes it challenging to measure a given user's contribution to the diffusion process.

3.1 Modeling latent diffusions

Diffusion scenarios. We focus on tree-structured diffusion graphs, i.e. each node v_j has only one incoming link (v_i, v_j), $i < j$. Denote the set of trees that are consistent with the temporal order in cascade C as $\mathcal{G}$, we call each diffusion tree a *diffusion scenario* $G \in \mathcal{G}$. Fig. 1a contains a cascade visualized as a star graph, attributing subsequent tweets to the first tweet at t_1. Fig. 1b shows four example diffusion scenarios consistent with this cascade. The main challenge here is to estimate the influence of each user in the cascade, taking into account all possible diffusion trees.

Probability of retweeting. For each tweet v_j, we model the probability of it being a direct descendant of each previous tweet in the same cascade as a weighted softmax function, defined by two main factors: firstly, users retweet *fresh* content (Wu and Huberman 2007). We assume that the probability of retweeting decays exponentially with the time difference $t_j - t_i$; secondly, users prefer to retweet locally influential users, known as preferential attachment (Barabási 2005; Rizoiu et al. 2017). We measure the local influence m_i of a user u_i using her number of followers (Kwak et al. 2010; Cha et al. 2010). We quantify the probability that v_j is a direct retweet of v_i as:

$$p_{ij} = \frac{m_i e^{-r(t_j - t_i)}}{\sum_{k=1}^{j-1} m_k e^{-r(t_j - t_k)}} \quad (1)$$

where r is a hyper-parameter controlling the temporal decay. It is set to $r = 6.8 \times 10^{-4}$, tuned using linear search on a sample of 20 real retweet cascades (details in the supplement (sup 2018, annex D)).

3.2 Tweet influence in a retweet cascade

We additionally assume retweets follow *independent conditional diffusions* within a cascade. This is to say that conditioned on an existing partial cascade of $j-1$ retweets denoted as $V^{(j-1)} = \{v_k\}_{k=1}^{j-1}$ whose underlying diffusion scenario is $G^{(j-1)}$, the j^{th} retweet is attributed to any of the

$k = 1, \ldots, j-1$ prior tweets according to Eq. 1, and is independent of the diffusion scenario $G^{(j-1)}$. For example, the 5^{th} tweet in the cascade will incur four valid diffusion trees for each of the diffusion scenarios for 4 tweets – this is illustrated in Fig. 1c. This simplifying assumption is reasonable, as it indicates that each user j makes up his/her own mind about whom to retweet, and that the history of retweets is available to user j (as is true in the current user interface of Twitter). It is easy to see that under this model, the total number of valid diffusion trees for a 5-tweet cascade is $1 \cdot 2 \cdot 3 \cdot 4 = 24$, and that for a cascade with n tweets is $(n-1)!$.

The goal for influence estimation for each cascade is to compute the contribution $\varphi(v_i)$ of each tweet v_i *averaging* over all independent conditional diffusion trees consistent with cascade V and with edge probabilities prescribed by Eq. 1. Enumerating all valid trees and averaging is clearly computationally intractable, but the illustration in Fig. 1c lends itself to a recursive algorithm.

Tractable tweet influence computation We introduce the *pair-wise influence score* m_{ij} which measures the influence of v_i over v_j. v_i can influence v_j both directly when v_j is a retweet of v_i, and indirectly when a path exists from v_i to v_j in the underlying diffusion scenario. Let v_k be a tweet on the path from v_i to v_j ($i < k < j$) so that v_j is a direct retweet of v_k. m_{ik} can be computed at the k^{th} recursion step and it measures the influence of v_i over v_k over all possible paths starting with v_i and ending with v_k. Given the above independent diffusions assumption, the m_{ij} can be computed using m_{ik} to which we add the edge (v_k, v_j). User u_j can chose to retweet any of the previous tweets with probability $p_{kj}, k < j$, therefore we further weight the contribution through v_k using p_{ij}. We consider that a tweet has a unit influence over itself ($m_{ii} = 1$). Finally, we obtain that:

$$m_{ij} = \begin{cases} \sum_{k=i}^{j-1} m_{ik} p_{kj}^2 & , i < j \\ 1 & , i = j \\ 0 & , i > j. \end{cases} \quad (2)$$

Naturally, $\varphi(v_i)$ the total influence of node v_i is the sum of $m_{ij}, j > i$ the pair-wise influence score of v_i over all subsequent nodes v_j. The recursive algorithm has three steps.

1. **Initialization.** $m_{ij} = 0$ for $i, j = 1, \ldots, n, j \neq i$, and $m_{ii} = 1$ for $i = 1, \ldots, n$;
2. **Recursion.** For $j = 2, \ldots, n$;
   (a) For $k = 1, \ldots, j-1$, compute p_{kj} using Eq. (1);
   (b) For $i = 1, \ldots, j-1$, $m_{ij} = \sum_{k=i}^{j-1} m_{ik} p_{kj}^2$;
3. **Termination.** Output $\varphi(v_i) = \sum_{k=i+1}^{n} m_{ik}$, for $i = 1, \ldots, n$.

We exemplify this algorithm on a 3-tweet toy example. Consider the cascade $\{v_1, v_2, v_3\}$. When the first tweet v_1 arrives, we have $m_{11} = 1$ by definition (see Eq. (2)). After the arrival of the second tweet, which must be retweeting the first, we have $m_{12} = m_{11} p_{12}^2 = 1$, and $m_{22} = 1$ by definition. The third tweet can be a retweet of the first or the second, therefore we obtain:

$$\begin{aligned} m_{13} &= m_{11} p_{13}^2 + m_{12} p_{23}^2 \; ; \\ m_{23} &= m_{22} p_{23}^2 \; ; \\ m_{33} &= 1 \; . \end{aligned}$$

The second term of m_{13} accounts for the indirect influence of v_1 over v_3 through v_2. This is the final step for a 3-node cascade.

The computational complexity of this algorithm is $O(n^3)$. There are n recursion steps, and calculating p_{ij} at sub-step (a) needs $O(n)$ units of computation, and sub-step (b) takes $O(n^2)$ steps. In real cascades containing 1000 tweets, the above algorithm finishes in 34 seconds on a PC. For more details and examples, see the online supplement (sup 2018, annex B).

3.3 Computing influence of a user

Given $\mathcal{T}(u)$ – the set of tweets authored by user u –, we define the user influence of u as the mean tweet influence of tweets $v \in \mathcal{T}(u)$:

$$\varphi(u) = \frac{\sum_{v \in \mathcal{T}(u)} \varphi(v)}{|\mathcal{T}(u)|}, \mathcal{T}(u) = \{v | u_v = u\} \quad (3)$$

To account for the skewed distribution of user influence, we mostly use the normalization – percentiles with a value of 1 for the most influential user our dataset and 0 for the least influential – denoted $\varphi(u)\%$.

4 Dataset and measures of political behavior

In this section, we first describe the #DEBATENIGHT dataset that we collected during the 1st U.S. presidential debate. Next, we introduce three measures for analyzing the political behavior of users who were active on Twitter during the debate. In Sec. 4.1, we introduce *political polarization* $\mathcal{P}$ and *political engagement* $\mathcal{E}$. In Sec. 4.2 we introduce the *botness score* ζ and we describe how we construct the reference bot and human populations.

The #DEBATENIGHT dataset contains Twitter discussions that occurred during the 1st 2016 U.S presidential debate between Hillary Clinton and Donald Trump. Using the Twitter Firehose API[3], we collected all the tweets (including retweets) that were authored during the two hour period from 8.45pm to 10.45pm EDT, on 26 September 2016, and which contain at least one of the hashtags: `#DebateNight`, `#Debates2016`, `#election2016`, `#HillaryClinton`, `#Debates`, `#Hillary2016`, `#DonaldTrump` and `#Trump2016`. The time range includes the 90 minutes of the presidential debate, as well as 15 minutes before and 15 minutes after the debate. The resulting dataset contains 6,498,818 tweets, emitted by 1,451,388 twitter users. For each user, the Twitter API provides aggregate information such as the number of followers, the total number (over the lifetime of the user) of emitted tweets, authored retweets, and favorites. For individual tweets, the API provides the timestamp and, if it is

[3] Via the Uberlink Twitter Analytics Service.

Figure 2: Wordclouds of partisan hashtags in #DEBATENIGHT: Democrat **(top)** and Republican **(bottom)**. Hashtags sizes are scaled by their frequency.

a retweet, the original tweet that started the retweet cascade. The #DEBATENIGHT dataset contains 200,191 retweet diffusions of size 3 and larger.

4.1 Political polarization $\mathcal{P}$ and engagement $\mathcal{E}$

Protocol. Content analysis (Kim and Kuljis 2010) was used to code the 1000 most frequently occurring hashtags according to their political polarity. More specifically, we used Directed Content Analysis (Hsieh and Shannon 2005) to contextually analyse hashtags and code them according to their political polarity (or not, denoted as 'neutral' and subsequently excluded from analysis). This approach has been used in previous work to study hashtags on Twitter in a manner that is valid, reliable and replicable (Small 2011). There were two previous studies of Twitter activity during the 2016 U.S. presidential election that informed the development of our coding schema. Firstly, Bessi and Ferrara (2016) devised a binary classification scheme that attributed political partisanship to a small set of key hashtags as either 'Trump-supporting' (#donaldtrump, #trump2016, #neverhillary, #trumppence16, #trump) or 'Clinton-supporting' (#hillaryclinton, #imwithher, #nevertrump, #hillary). Secondly, in studying Twitter activity during the 1st U.S. presidential debate, Kollanyi, Howard, and Woolley (2016) developed a coding schema that categorized tweets into seven categories based on the hashtags that occurred within the tweet. However, the authors found that three 'exclusive' categories ('Pro-Trump', 'Pro-Clinton', and 'Neutral') accounted for the majority (88.5%) of observations.

Given the findings of previous research, we developed a code book with three categories: 'Pro-Trump', 'Pro-Clinton', and 'Neutral'. To ensure that hashtags were analyzed within context, our content analysis methodology focussed on three units of analysis (following the approach developed by Small (2011)). The first is hashtags, comprised of a set of the 1000 most frequently occurring hashtags over all tweets in our dataset. The second unit of analysis was individual tweets that contained these hashtags. In order to gain a more nuanced and 'situated' interpretation of hashtag usage, for each hashtag we referred to a small random sample of tweets in our dataset that contained each given hashtag. In some instances the polarity (or neutrality) was clear and/or already determined from previous studies, which helped to speed up the analysis of tweets. The third unit of analysis was user profiles, which we referred to in situations where the polarity or neutrality of a given hashtag was unclear from the context of tweet analysis. For example, #partyoflincoln was used by both Republican and Democrat Twitter users, but an analysis of both tweets and user profiles indicated that this hashtag was *predominantly* used by Pro-Trump supporters to positively align the Republican Party with the renowned historical figure of President Abraham Lincoln, who was a Republican. The content analysis resulted in a subset of 93 pro-Democrat and 86 pro-Republican hashtags (see the wordcloud visualization in Fig. 2), whilst the remaining 'neutral' hashtags were subsequently excluded from further analysis. The resulting partisan hashtag list contains hashtags indicating either strong support for a candidate (e.g., `#imwithher` for Clinton and `#trump2016` for Trump), or opposition and/or antagonism (e.g., `#nevertrump` and `#crookedhillary`). The complete list of partisan hashtags is publicly available in the Github repository.

Two measures of political behavior. We identify 65,031 tweets in #DEBATENIGHT that contain at least one partisan hashtag (i.e., one of hashtags in the reference set of partisan hashtags constructed earlier). 1,917 tweets contain partisan hashtags with both polarities: these are mostly negative tweets towards both candidates (e.g., "Let's Get READY TO RUMBLE AND TELL LIES. #nevertrump #neverhillary #Obama") or hashtag spam. We count the number of occurrences of partisan hashtags for each user, and we detect a set of 46,906 politically engaged users that have used at least one partisan hashtag. Each politically engaged user u_i has two counts: dem_i the number of Democrat hashtags that u_i used, and rep_i the number of Republican hashtags. We measure the *political polarization* as the normalized difference between the number of Republican and Democrat hashtags used:

$$\mathcal{P}(u_i) = \frac{rep_i - dem_i}{rep_i + dem_i}. \quad (4)$$

$\mathcal{P}(u_i)$ takes values between -1 (if u_i emitted only Democrat partisan hashtags) and 1 (u_i emitted only Republican hashtags). We threshold the political polarization to construct a population of Democrat users with $\mathcal{P}(u) \leq -0.4$ and Republican users with $\mathcal{P}(u) \geq 0.4$. In the set of politically engaged users, there are 21,711 Democrat users, 22,644 Republican users and 2,551 users with no polarization ($\mathcal{P}(u) \in (-0.4, 0.4)$). We measure the *political engagement* of users using the total volume of partisan hashtags included in their tweets $\mathcal{E}(u_i) = rep_i + dem_i$.

Table 1: Tabulating population volumes and percentages of politically polarized users over four populations: `Protected`, `Human`, `Suspended` and `Bot`.

	All	`Prot.`	`Human`	`Susp.`	`Bot`
All	1,451,388	45,316	499,822	10,162	17,561
Polarized	44,299	1,245	11,972	265	435
Democrat	21,676	585	5,376	111	185
Republican	22,623	660	6,596	154	250
Dem. %	48.93%	46.99%	44.90%	41.89%	42.53%
Rep. %	51.07%	53.01%	55.10%	58.11%	57.47%

4.2 Botness score ζ and bot detection

Detecting automated bots. We use the BotOrNot (Davis et al. 2016) API to measure the likelihood of a user being a bot for each of the 1,451,388 users in the #DEBATENIGHT dataset. Given a user u, the API returns the botness score $\zeta(u) \in [0, 1]$ (with 0 being likely human, and 1 likely non-human). Previous work (Varol et al. 2017; Bessi and Ferrara 2016; Woolley and Guilbeault 2017) use a botness threshold of 0.5 to detect socialbots. However, we manually checked a random sample of 100 users with $\zeta(u) > 0.5$ and we found several human accounts being classified as bots. A threshold of 0.6 decreases misclassification by 3%. It has been previously reported by Varol et al. (2017) that organizational accounts have high botness scores. This however is not a concern in this work, as we aim to detect 'highly automated' accounts that behave in a non-human way. We chose to use a threshold of 0.6 to construct the `Bot` population in light of the more encompassing notion of account *automation*.

Four reference populations. In addition to the `Bot` population, we construct three additional reference populations: `Human` $\zeta(u) \leq 0.2$ contains users with a high likelihood of being regular Twitter users. `Protected` are the users whose profile has the access restricted to their followers and friends (the BotOrNot system cannot return the botness score); we consider these users to be regular Twitter users, since we assume that no organization or broadcasting bot would restrict access to their profile. *Suspended* are those users which have been suspended by Twitter between the date of the tweet collection (26 September 2016) and the date of retrieving the botness score (July 2017); this population has a high likelihood of containing bots. Table 1 tabulates the size of each population, split over political polarization.

5 Evaluation of user influence estimation

In this section, we evaluate our proposed algorithm and measure of user influence. In Sec 5.1, we evaluate on synthetic data against a known ground truth. In Sec. 5.2, we compare the $\varphi(u)$ measure (defined in Sec. 3.3) against two alternatives: the number of followers and the mean size of initiated cascades.

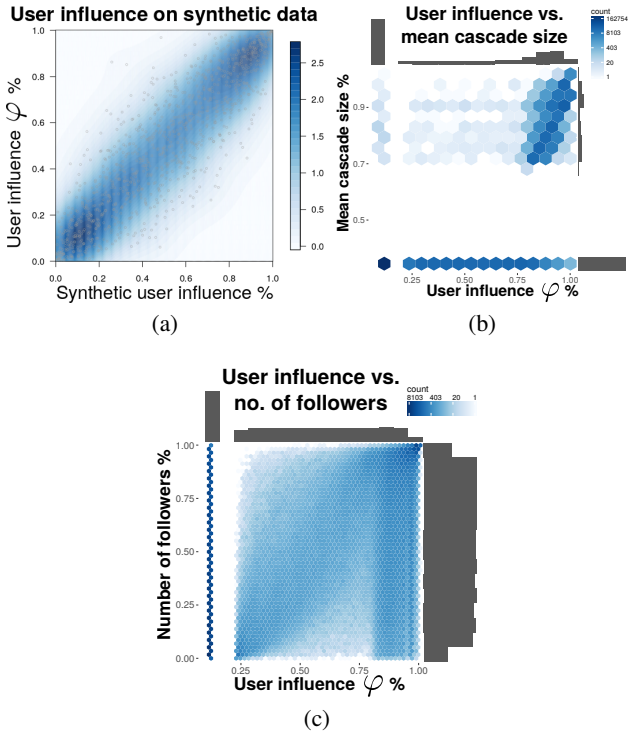
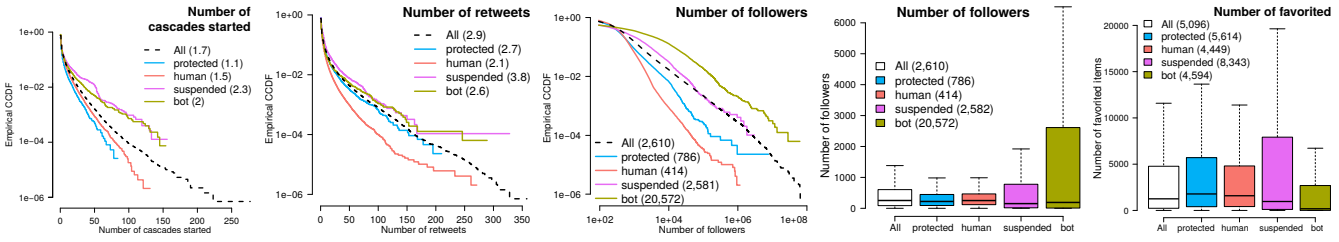


Figure 3: Evaluation of the user influence measure. **(a)** 2D density plot (shades of blue) and scatter-plot (gray circles) of user influence against the ground truth on a synthetic dataset. **(b)(c)** Hexbin plot of user influence percentile (x-axis) against mean cascade size percentile (b) and the number of followers (c) (y-axis) on #DEBATENIGHT. The color intensity indicates the number of users in each hex bin. 1D histograms of each axis are shown using gray bars. Note 72.3% of all users that initiate cascades are never retweeted.

5.1 Evaluation of user influence

Evaluating user influence on real data presents two major hurdles. The first is the lack of ground truth, as user influence is not directly observed. The second hurdle is that the diffusion graph is unknown, which renders impossible comparing to state-of-the-art methods which require this information (e.g. ConTinEst (Du et al. 2013)). In this section, evaluate our algorithm against a known ground truth on a synthetic dataset, using the same evaluation approach used for ConTinEst.

Evaluation on synthetic data. We evaluate on synthetic data using the protocol previously employed in (Du et al. 2013). We use the simulator in (Du et al. 2013) to generate an artificial social network with 1000 users. We then simulate 1000 cascades through this social network, starting from the same initial user. The generation of the synthetic social network and of the cascades is detailed in the online supplement (sup 2018, annex C). Similar to the retweet cascades in #DEBATENIGHT, each event in the synthetic cascades has a timestamp and an associated user. Unlike the real retweet cascades, we know the real diffusion structure behind each



(a) (b) (c) (d) (e)

Figure 4: Profiling behavior of the `Protected`, `Human`, `Suspended` and `Bot` populations in the #DEBATENIGHT dataset. The numbers in parentheses in the legend are mean values. **(a)** CCDF of the number of Twitter diffusion cascades started. **(b)** CCDF of the number of retweets. **(c)(d)** CCDF (c) and boxplots (d) of the number of followers. **(e)** Number of items favorited.

synthetic cascade. For each user u, we count the number of nodes reachable from u in the diffusion tree of each cascade. We compute the influence of u as the mean influence over all cascades. ConTinEst (Du et al. 2013) has been shown to asymptotically approximate this synthetic user influence.

We use our algorithm introduced in Sec. 3.2 on the synthetic data, to compute the measure $\varphi(u)$ defined in Eq. 3. We plot in Fig. 3a the 2D scatter-plot and the density plot of the synthetic users, with our influence measure φ on the y-axis and the ground truth on the x-axis (both in percentiles). Visibly, there is a high agreement between the two measures, particularly for the most influential and the least influential users. The Spearman correlation coefficient of the raw values is 0.88. This shows that our method can output reliable user influence estimates in the absence of any information about the structure of the diffusions.

5.2 Comparison with other influence metrics

We compare the influence measure $\varphi(u)$ against two alternatives that can be computed on #DEBATENIGHT.

Mean size of initiated cascades (of a user u) is the average number of users reached by original content authored by u. It should be noted that this measure does not capture u's role in diffusing content authored by someone else. In the context of Twitter, mean size of initiated cascades is the average number of users who retweeted an original tweet authored by u: we compute this for every user in the #DEBATENIGHT dataset, and we plot it against $\varphi(u)$ in Fig. 3b. Few users have a meaningful value for mean cascade size: 55% of users never start a cascades (and they are not accounted for in Fig. 3b); out of the ones that start cascades 72.3% are never retweeted and they are all positioned at the lowest percentile (shown by the 1D histograms in the plot). It is apparent that the mean cascade size metric detects the influential users that start cascades, and it correlates with $\varphi(u)$. However, it misses highly influential users who never initiate cascades, but who participate by retweeting. Examples are user *@SethMacFarlane* (the actor and filmmaker Seth MacFarlane, 10.8 million followers) or user *@michaelianblack* (comedian Michael Ian Black, 2.1 million followers), both with φ in the top 0.01% most influential users.

Number of followers is one of the simplest measures of direct influence used in literature (Mishra, Rizoiu, and Xie 2016; Zhao et al. 2015). While being loosely correlated with $\varphi(u)$ (visible in Fig. 3c, Pearson $r = 0.42$), it has the drawback of not accounting for any of the user actions, such as an active participation in discussions or generating large retweet cascades. For example, user *@PoliticJames* (alt-right and pro-Trump, 2 followers) emitted one tweet in #DEBATENIGHT, which was retweeted 18 times and placing him in the top 1% most influential users. Similarly, user *@tiwtter1tr4_tv* (now suspended, 0 followers) initiated a cascade of size 58 (top 1% most influential). Interestingly, half of the accounts scoring on the bottom 1% by number of followers and top 1% by influence are now suspended or have very high botness scores.

6 Results and findings

In this section, we present an analysis of the interplay between botness, political behavior (polarization and engagement) and influence. In Sec. 6.1, we first profile the activity of users in the four reference populations; next, we analyze the political polarization and engagement, and their relation with the botness measure. Finally, in Sec. 6.2 we tabulate user influence against polarization and botness, and we construct *the polarization map*.

6.1 Political behavior of humans and bots

Twitter activity across four populations. We measure the behavior of users in the four reference populations defined in Sec. 4.1 using several measures computed from the Twiter API. The number of cascades started (i.e., number of original tweets) and the number of posted retweets are simple measures of activity on Twitter, and they are known to be long-tail distributed (Cha et al. 2010). Fig. 4a and 4b respectively plot the log-log plot of the empirical Complementary Cumulative Distribution Function (CCDF) for each of the two measures. It is apparent that users in the `Bot` and `Suspended` populations exhibit higher levels of activity than the general population, whereas the `Human` and `Protected` populations exhibit lower level. Fig. 4c and 4d plot the number of followers and present a more nuanced story: the average bot user has 10 times more followers than the average human user; however, bots have a median of 190 followers, less than the median 253 followers of human users. In other words, some bots are very highly followed,

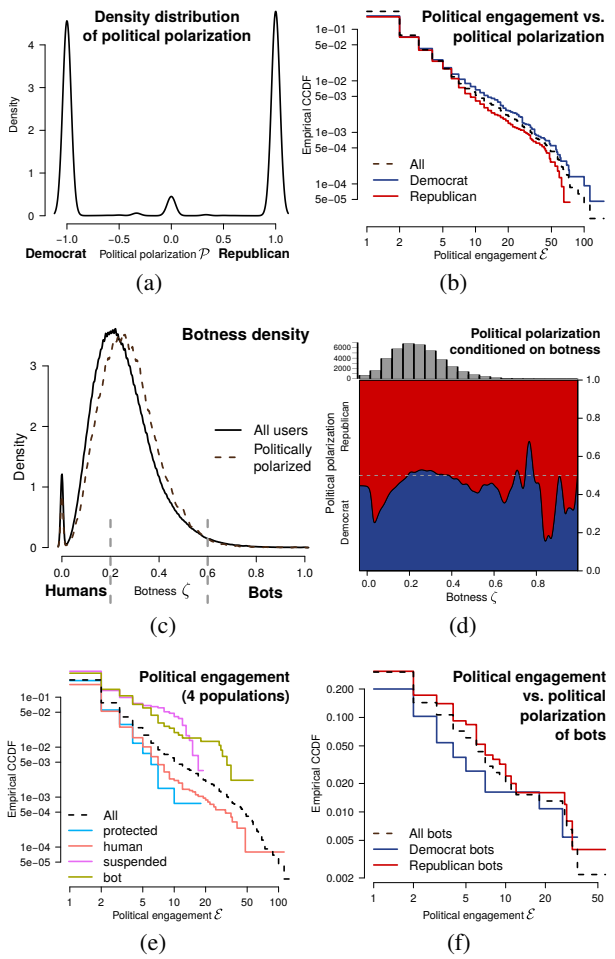


Figure 5: Political polarization, engagement and botness. **(a)** The density distribution of political polarization $\mathcal{P}$. **(b)** Log-log plot of the CCDF of political engagement $\mathcal{E}$ for the Democrat and Republican populations. **(c)** The density distribution of botness ζ for the entire population (solid line) and the politically polarized population (dashed line). **(d)** The conditional density of polarization conditioned on botness. The top panel shows the volumes of politically polarized users in 30 bins. **(e)(f)** CCDF of political engagement for the reference populations (e) and for the polarized `Bot` populations (f).

but most are simply ignored. Finally, Fig. 4e shows that bots favorite less than humans, indicating that their activity patterns differ from those of humans.

Political polarization and engagement. The density distribution of political polarization (Fig. 5a) shows two peaks at -1 and 1, corresponding to strongly pro-Democrat and strongly pro-Republican respectively. The shape of the density plot is consistent with the sizes of Republican and Democrat populations (Sec. 4.1), and the extreme bimodality can be explained by the clear partisan nature of the chosen hashtags and by the known political polarization of users on Twitter (Conover et al. 2011; Barberá et al. 2015), which will be greatly enhanced in the context of a political debate. Fig. 5b presents the log-log plot of the CCDF of the political engagement, which shows that the political engagement score is long-tail distributed, with *pro-Democrats slightly more engaged than pro-Republicans overall* (t-test significant, p-val $= 0.0012$).

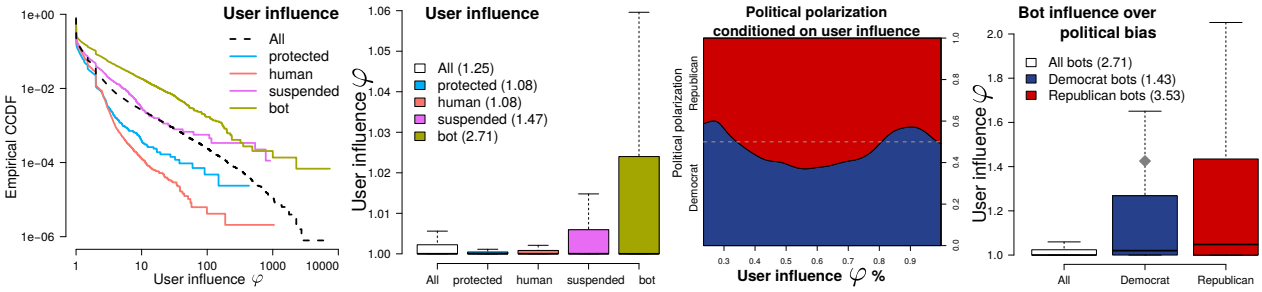
Botness and political polarization. The distribution of botness ζ exhibits a large peak around $[0.1, 0.4]$ and a long tail (Fig. 5c). The dashed gray vertical lines show the thresholds used in Sec. 4.2 for constructing the reference `Human` ($\zeta \in [0, 0.2]$) and `Bot` ($\zeta \in [0.6, 1]$) populations. Fig. 5d shows the conditional density of polarization conditioned on botness. For both high botness scores (i.e., bots) and low botness scores (humans) the likelihood of being pro-Republican is consistently higher than that of being pro-Democrat, while users with mid-range botness are more likely to be pro-Democrat. In other words, *socialbots accounts are more likely to be pro-Republican than to be pro-Democrat*.

Political engagement of bots. Fig. 5e shows the CCDF of political engagement of the four reference populations, and it is apparent that the `Bot` and `Suspended` populations exhibit consistently higher political engagement than the `Human` and `Protected` populations. Fig. 5f shows the CCDF of political engagement by the political partisanship of bots and we find that pro-Republican `Bot` accounts are more politically engaged than their pro-Democrat counterparts. In summary, *socialbots are more engaged than humans (p-val* $= 8.55 \times 10^{-5}$*), and pro-Republican bots are more engaged than their pro-Democrat counterparts (p-val = 0.1228)*.

6.2 User influence and polarization map

User influence across four populations. First, we study the distribution of user influence across the four reference populations constructed in Sec. 4.2. We plot the CCDF in Fig. 6a and we summarize user influence as boxplots in Fig. 6b for each population. User influence φ is long-tail distributed (shown in Fig. 6a) and it is higher for `Bot` and `Suspended` populations, than for `Human` and `Protected` (shown in Fig 6b). There is a large discrepancy between the influence of `Human` and `Bot` (p-val $= 0.0025$), with *the average bot having 2.5 times more influence than the average human.* We further break down users in the `Bot` population based on their political polarization. Fig. 6d aggregates as boxplots the influence of pro-Democrat and pro-Republican bots (note: not all bots are politically polarized). Notably, on a per-bot basis, pro-Republican bots are more influential than their pro-Democrat counterparts (p-val $= 0.0096$) – *the average pro-Republican bot is twice as influential as the average pro-Democrat bot*.

Political polarization and user influence. Next, we analyze the relation between influence and polarization. Fig. 6c plots the probability distribution of political polarization, conditioned on user influence $\varphi\%$. While for mid-range influential users ($\varphi\% \in [0.4, 0.8]$) the likelihood of being Republican is higher than being Democrat, we observe the inverse situation on the higher end of the influence scale. *Very highly influential users ($\varphi\% > 0.8$]) are more likely to be pro-Democrat*, and this is consistent with the fact that many



(a) (b) (c) (d)

Figure 6: Profiling influence, and linking to botness and political behavior. **(a)(b)** User influence $\varphi(u)$ for the reference populations, shown as log-log CCDF plot (a) and boxplots (b). **(c)** Probability distribution of polarization, conditional on $\varphi(u)\%$. **(d)** Boxplots of user influence for the pro-Democrat and pro-Republican `Bot` users. Numbers in parenthesis show mean values.

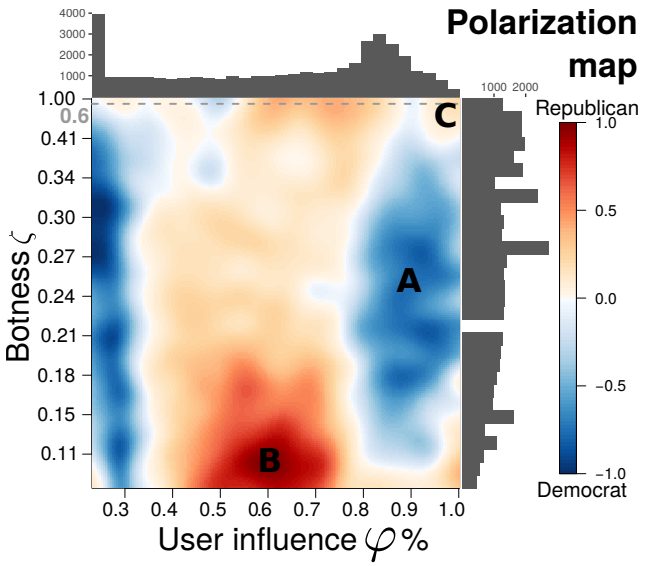


Figure 7: Political polarization by user influence $\varphi(u)\%$ (x-axis) and bot score ζ (y-axis). The gray dashed horizontal line shows the threshold of 0.6 above which a user is considered a bot. The color in the map shows political polarization: areas colored in bright blue (red) are areas where the Democrats (Republicans) have considerably higher density than Republicans (Democrats). Areas where the two populations have similar densities are colored white. Three areas of interest are shown by the letter **A**, **B** and **C**.

public figures were supportive of the Democrat candidate during the presidential campaign.

The polarization map. Finally, we create a visualization that allows us to jointly account for botness and user influence when studying political partisanship. We project each politically polarized user in #DEBATENIGHT onto the two-dimensional space of user influence $\varphi\%$ (x-axis) and botness ζ (y-axis). The y-axis is re-scaled so that an equal length interval around any botness value contains the same amount of users, This allows to zoom in into denser areas like $\zeta \in [0.2, 0.4]$, and to deal with data sparsity around high botness scores. We compute the 2D density estimates for the pro-Democrat and pro-Republican users (shown in the online supplement (sup 2018, annex E)). For each point in the space $(\varphi\%, \zeta)$ we compute a score as the log of the ratio between the density of the Republican users and that of the pro-Democrats, which is then renormalized so that values range from -1 (mostly Democrat) to +1 (mostly Republican). The resulting map – dubbed the *polarization map* – is shown in Fig. 7 and it provides a number of insights. Three areas of interest (**A**, **B** and **C**) are shown on Fig. 7. Area **A** is a pro-Democrat area corresponding to highly influential users (already shown in Fig. 6c) that spans across most of the range of botness values. Area **B** is the largest predominantly pro-Republican area and it corresponds to mid-range influence (also shown in Fig. 6c) and concentrates around small botness values – this indicates the presence of a large pro-Republican population of mainly human users with regular user influence. Lastly, we observe that the top-right area **C** (high botness and high influence) is predominantly red: In other words *highly influential bots are mostly pro-Republican.*

7 Discussion

In this paper, we study the influence and the political behavior of socialbots. We introduce a novel algorithm for estimating user influence from retweet cascades in which the diffusion structure is not observed. We propose four measures to analyze the role and user influence of bots versus humans on Twitter during the 1st U.S. presidential debate of 2016. The first is the user influence, computed over all possible unfoldings of each cascade. Second, we use the BotOrNot API to retrieve the botness score for a large number of Twitter users. Lastly, by examining the 1000 most frequently-used hashtags we measure political polarization and engagement. We analyze the interplay of influence, botness and political polarization using a two-dimensional map – the polarization map. We make several novel findings, for example: bots are more likely to be pro-Republican; the average pro-Republican bot is twice as influential as its pro-Democrat counterpart; very highly influential users are more likely to be pro-Democrat; and highly influential bots are

mostly pro-Republican.

Validity of analysis with respect to BotOrNot. The BotOrNot algorithm uses tweet content and user activity patterns to predict botness. However, this does not confound the conclusions presented in Sec. 6. First, political behavior (polarization and engagement) is computed from a list of hashtags specific to #DEBATENIGHT, while the BotOrNot predictor was trained before the elections took place and it has no knowledge of the hashtags used during the debate. Second, a loose relation between political engagement and activity patterns could be made, however we argue that engagement is the number of used partisan hashtags, not tweets – i.e. users can have a high political engagement score after emitting few very polarized tweets.

Assumptions, limitations and future work. This work makes a number of simplifying assumptions, some of which can be addressed in future work. First, the delay between the tweet crawling (Sept 2016) and computing botness (July 2017) means that a significant number of users were suspended or deleted. A future application could see simultaneous tweets and botscore crawling. Second, our binary hashtag partisanship characterization does not account for independent voters or other spectra of democratic participation, and future work could evaluate our approach against a clustering approach using follower ties to political actors (Barberá et al. 2015). Last, this work computes the expected influence of users in a particular population, but it does not account for the aggregate influence of the population as a whole. Future work could generalize our approach to entire populations, which would allow answers to questions like "Overall, were the Republican bots more influential than the Democrat humans?".

Acknowledgments. This research is sponsored in part by the Air Force Research Laboratory, under agreement number FA2386-15-1-4018.

References

[Barabási 2005] Barabási, A.-L. 2005. The origin of bursts and heavy tails in human dynamics. *Nature* 435(7039):207–11.

[Barberá et al. 2015] Barberá, P.; Jost, J. T.; Nagler, J.; Tucker, J. A.; and Bonneau, R. 2015. Tweeting from left to right: Is online political communication more than an echo chamber? *Psychological Science* 26(10):1531–1542.

[Bessi and Ferrara 2016] Bessi, A., and Ferrara, E. 2016. Social bots distort the 2016 u.s. presidential election online discussion. *First Monday* 21(11).

[Cha et al. 2010] Cha, M.; Haddadi, H.; Benevenuto, F.; and Gummadi, K. P. 2010. Measuring User Influence in Twitter: The Million Follower Fallacy. In *ICWSM '10*, volume 10, 10–17.

[Chikhaoui et al. 2017] Chikhaoui, B.; Chiazzaro, M.; Wang, S.; and Sotir, M. 2017. Detecting communities of authority and analyzing their influence in dynamic social networks. *ACM Trans. Intell. Syst. Technol.* 8(6):82:1–82:28.

[Cho et al. 2013] Cho, Y.-S.; Galstyan, A.; Brantingham, P. J.; and Tita, G. 2013. Latent self-exciting point process model for spatial-temporal networks. *Disc. and Cont. Dynamic Syst. Series B*.

[Collins and Cox 2017] Collins, B., and Cox, J. 2017. Jenna abrams, russia's clown troll princess, duped the mainstream media and the world. `Dailybeast.com`.

[Conover et al. 2011] Conover, M. D.; Ratkiewicz, J.; Francisco, M.; Goncalves, B.; Menczer, F.; and Flammini, A. 2011. Political polarization on Twitter. In *ICWSM'11*, 89–96. AAAI.

[Davis et al. 2016] Davis, C. A.; Varol, O.; Ferrara, E.; Flammini, A.; and Menczer, F. 2016. Botornot: A system to evaluate social bots. In *WWW Companion*, 273–274. ACM.

[Du et al. 2013] Du, N.; Song, L.; Gomez-Rodriguez, M.; and Zha, H. 2013. Scalable Influence Estimation in Continuous-Time Diffusion Networks. In *NIPS'13*, 3147–3155.

[Gehl and Bakardjieva 2016] Gehl, R., and Bakardjieva, M. 2016. *Socialbots and Their Friends: Digital Media and the Automation of Sociality*.

[Hsieh and Shannon 2005] Hsieh, H.-F., and Shannon, S. E. 2005. Three approaches to qualitative content analysis. *Qualitative Health Research* 15(9):1277–1288.

[Kim and Kuljis 2010] Kim, I., and Kuljis, J. 2010. Applying content analysis to web-based content. *Jour. of Comp. and Inf. Tech.* 18(4):369–375.

[Kollanyi, Howard, and Woolley 2016] Kollanyi, B.; Howard, P. N.; and Woolley, S. C. 2016. Bots and automation over Twitter during the first U.S. presidential debate: Comprop data memo 2016.1. Oxford, UK.

[Kwak et al. 2010] Kwak, H.; Lee, C.; Park, H.; and Moon, S. 2010. What is Twitter, a social network or a news media? In *WWW'10*, 591–600.

[Li and Zha 2013] Li, L., and Zha, H. 2013. Dyadic event attribution in social networks with mixtures of hawkes processes. In *CIKM'13*, 1667–1672. ACM.

[Linderman and Adams 2014] Linderman, S., and Adams, R. 2014. Discovering latent network structure in point process data. In *ICML'14*, 1413–1421.

[Mishra, Rizoiu, and Xie 2016] Mishra, S.; Rizoiu, M.-A.; and Xie, L. 2016. Feature driven and point process approaches for popularity prediction. In *CIKM'16*, 1069–1078.

[Page et al. 1999] Page, L.; Brin, S.; Motwani, R.; and Winograd, T. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

[Rizoiu et al. 2017] Rizoiu, M.-A.; Xie, L.; Sanner, S.; Cebrian, M.; Yu, H.; and Van Hentenryck, P. 2017. Expecting to be HIP: Hawkes Intensity Processes for Social Media Popularity. In *WWW'17*, 735–744.

[Rodriguez, Balduzzi, and Schölkopf 2011] Rodriguez, M. G.; Balduzzi, D.; and Schölkopf, B. 2011. Uncovering the Temporal Dynamics of Diffusion Networks. In *ICML'11*, 561–568.

[Rozenshtein and Gionis 2016] Rozenshtein, P., and Gionis, A. 2016. Temporal pagerank. In *ECML-PKDD'16*, 674–689. Springer.

[Simma and Jordan 2010] Simma, A., and Jordan, M. I. 2010. Modeling events with cascades of poisson processes. *UAI'10*.

[Small 2011] Small, T. A. 2011. WHAT THE HASHTAG? A content analysis of Canadian politics on Twitter. *Information, Communication & Society* 14(6):872–895.

[sup 2018] 2018. Appendix: #DebateNight: The role and influence of socialbots on Twitter during the 1st 2016 U.S. presidential debate. `https://arxiv.org/pdf/1802.09808.pdf#page=11`.

[Timberg, Dwoskin, and Entous 2017] Timberg, C.; Dwoskin, E.; and Entous, A. 2017. Russian Twitter account pretending to be tennessee gop fools celebrities, politicians. `Chicagotribune.com`.

[Varol et al. 2017] Varol, O.; Ferrara, E.; Davis, C. A.; Menczer, F.; and Flammini, A. 2017. Online human-bot interactions: Detection, estimation, and characterization. In *ICWSM'17*, 280–289. AAAI.

[Weng et al. 2010] Weng, J.; Lim, E.-P.; Jiang, J.; and He, Q. 2010. Twitterrank: finding topic-sensitive influential twitterers. In *WSDM'10*, 261–270. ACM.

[Woolley and Guilbeault 2017] Woolley, S. C., and Guilbeault, D. 2017. Computational propaganda in the united states of america: Manufacturing consensus online: Working paper 2017.5.

[Wu and Huberman 2007] Wu, F., and Huberman, B. A. 2007. Novelty and collective attention. *PNAS* 104(45):17599–601.

[Yates, Joselow, and Goharian 2016] Yates, A.; Joselow, J.; and Goharian, N. 2016. The news cycle's influence on social media activity. In *ICWSM'16*.

[Zhao et al. 2015] Zhao, Q.; Erdogdu, M. A.; He, H. Y.; Rajaraman, A.; and Leskovec, J. 2015. SEISMIC: A Self-Exciting Point Process Model for Predicting Tweet Popularity. In *KDD'15*.

Contents (Appendix)

A Derivation of the influence formula

In this section, we detail the calculation of the tweet influence $\varphi(v)$, proposed in Sec. 3. In Sec. A.1, we define the notion of diffusion scenario, and we compute its likelihood given an observed retweet cascade. In Sec. A.2, we compute the formula for tweet influence over all possible diffusion scenarios associated with the given cascade.

A.1 Diffusion scenarios

Diffusion trees. We can represent an online diffusion using a directed tree $G(V, E)$, in which each node has a single parent and the direction of the edges indicates the flow of the information. For retweet cascades, the nodes $v \in V$ are individual tweets and each directed edge $e \in E, e = \{v_a, v_b\}$ (showing the direction $v_a \longrightarrow v_b$) indicates that v_b is a *direct retweet* of v_a. A direct retweet means that u_b – the user that emitted the tweet v_b – clicked on the "Retweet" option under tweet v_a. The top panel of Fig. 8a shows an example of such a diffusion tree. Note that each node v has associated a time of arrival t_v and that the diffusion tree respects the order of the times of arrival – i.e. given the edge $e = \{v_a, v_b\}$, then $t_a < t_b$. The bottom panel of Fig. 8a shows the incremental construction of the diffusion tree shown in top panel: node v_1 is the root of the tree and the source of the information diffusion; at each time t_i, node v_i attaches to the previous tree constructed at time t_{i-1}.

Diffusion scenarios for retweet cascades. The diffusion tree is not observed for real Twitter retweet cascades, since the Twitter API does not expose the direct retweet relationships. Instead, it assigns every retweet in the cascade to the original tweet. Every retweet cascades constructed based on raw retweet information from the Twitter API resembles the graph in Fig. 1a. Due to this particular shape, we denote retweet cascades as *stars*. However, the API exposes the time of arrival of the retweets t_i. We denote as a *diffusion scenario* any valid diffusion tree that could be associated with the observed retweet star – i.e., the edges in the diffusion tree respects the order of arrival of retweets. Fig. 1b

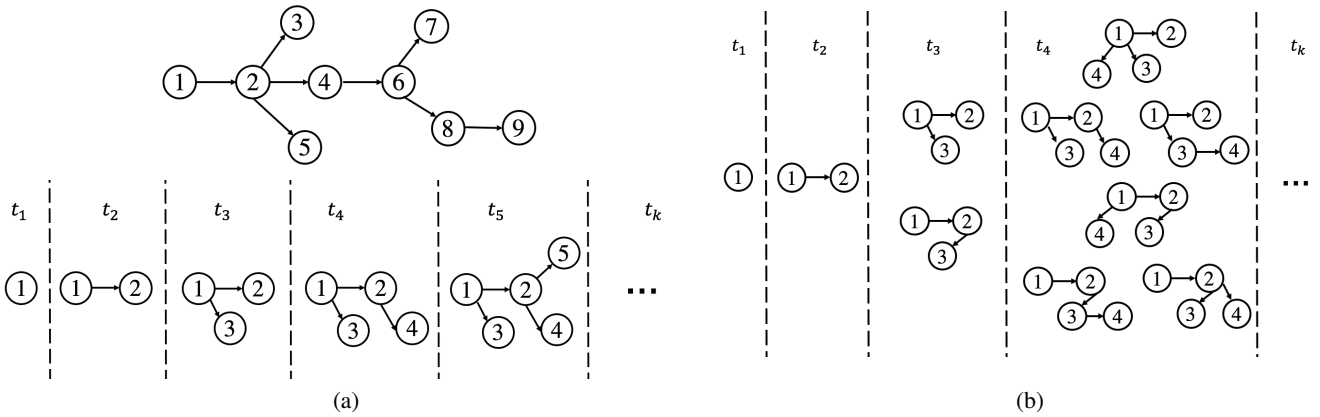


Figure 8: **(a)** An example of a diffusion tree **(top)** and its corresponding incremental diffusion process **(bottom)**. **(b)** Enumeration of all possible diffusion scenarios: each retweet v_k arrives at time t_k, and it can attach to any of the previous nodes, in any of the diffusion scenario constructed at time t_{k-1}. At time t_k there are $(k-1)!$ diffusion scenarios, each with k nodes.

shows four examples of diffusion scenarios associated with the star in Fig. 1a.

Constructing diffusion scenarios. Fig. 8b exemplifies a straight-forward method to enumerate all diffusion scenarios associated with the star in Fig. 1a. The node v_1 is the root node and it is published at time t_1; tweet v_2 occurs at time t_2 and it is undoubtedly a direct retweet of v_1 – a directed edge is drawn from v_1 to v_2. Tweet v_3 observed at t_3 can be a direct retweet of either tweet v_1 or tweet v_2. Therefore, at time t_3 there are two possible diffusion scenarios: G_1 with the edge set $E = \{\{v_1, v_2\}, \{v_1, v_3\}\}$ and G_2 with the edge set $E = \{\{v_1, v_2\}, \{v_2, v_3\}\}$. Similarly, v_4 can be a direct retweet of v_1, v_2 or v_3, in either G_1 or G_2. Consequently, at time t_4 there are 6 possible diffusion scenarios. The process continues until all nodes have been attached. For an observed star of size k, there are $(k-1)!$ associated diffusion scenarios.

Probability of a diffusion scenario. Given that in retweet cascades individual edges are not observed, we define the probability of an edge $\mathbb{P}(\{v_a, v_b\})$ as the likelihood that u_b emitted tweet v_b as a direct retweet of v_a. We model two factors in the likelihood of retweeting: firstly, users retweet *fresh* content (Wu and Huberman 2007). The probability of the edge e decays exponentially with the time difference $t_b - t_a$; secondly, users prefer to retweet locally influential users, also known as preferential attachment (Barabási 2005). We measure the local influence of a user using his number of followers (Kwak et al. 2010; Cha et al. 2010; Rizoiu et al. 2017). We quantify the probability of an edge as:

$$\mathbb{P}(\{v_a, v_b\}) = m_a e^{-r(t_b - t_a)} \tag{5}$$

where m_a is the number of followers of the user of u_a and r controls the temporal decay of the probability.

Under the assumption that retweeting events (i.e. edges) occur independently one from another, we obtain the probability of a diffusion scenario as:

$$\mathbb{P}(G) = \prod_{\{v_a, v_b\} \in E} \mathbb{P}(\{v_a, v_b\}) \tag{6}$$

Note that the above assumption of independence of retweet events is a strong assumption. Current state-of-the-art approaches (Mishra, Rizoiu, and Xie 2016; Zhao et al. 2015) for modeling retweet cascades employ self-exciting point processes, in which the arrival of one event increases the probability of future events. However, for our application of estimating the probability of a diffusion scenario it is the simplest assumption. Additional arguments in its favor are also that we are studying networks of events (in which each event is identified with unique user), not networks of users. The interdependence often observed in socially generated networks (like triadic closure) can be ignored in edge formation within a particular retweet cascade.

Table 2: Summary of notations.

Notation	Interpretation
$G(V, E)$	diffusion tree. In case the tree is unobserved, G is a diffusion scenario.
$v \in V$	node in the diffusion tree (i.e. retweets).
$e = \{v_a, v_b\} \in E$	directed edge in the diffusion tree, tweet v_b is a direct retweet of v_a.
t_v	time of arrival of node v (timestamp of the tweet).
u_v	user that has emitted tweet v.
m_v	local influence (i.e. number of followers) of user u_v.

A.2 Computing tweet influence

(Du et al. 2013) define user influence of a user u as the average number of users in the social network who get in contact with the content emitted by u. For retweet cascades the diffusion tree is not observed; it is impossible to directly measure user influence, apart from the root user. We define

the *tweet influence* over a retweet cascade as the expected number of time it is retweeted – direct retweets or descendants in a diffusion scenario –, over all possible diffusion scenarios associated with the given star. Finally, we compute the influence of user u as the sum of the influences of the tweets that u authored. We see that the definition in (Du et al. 2013) is a special case of our definition, in which the diffusion tree is observed.

Tweet influence over one diffusion scenario. Let $z(v_i, v_k) \in G$ be a path in the diffusion scenario G – i.e a sequence of nodes which starts with v_i and ends with v_k. $\mathbb{P}(z(v_i, v_k))$ is the probability of reaching v_k from v_i. $\varphi(v_i|G)$ is the influence of v_i and it is computed as the expected number of users reached from v_i using a model of independent binomials to decide whether or not to take each hop in each path that starts with v_i. Formally:

$$\begin{aligned}\varphi(v_i|G) &= \mathbb{E}\left[\sum_{v_k \in V(G)} \mathbb{1}\left\{z(v_i, v_k|G)\right\}\right] \\ &= \sum_{v_k \in V(G)} \mathbb{E}\left[\mathbb{1}\left\{z(v_i, v_k|G)\right\}\right] \\ &= \sum_{v_k \in V(G)} \mathbb{P}(z(v_i, v_k|G)) \qquad (7) \\ &= \sum_{v_k \in V(G)} \prod_{\{v_a, v_b\} \in z(v_i, v_k)} \mathbb{P}\big(\{v_a, v_b\}\big) \qquad (8)\end{aligned}$$

where $\mathbb{1}\left\{z(v_i, v_k|G)\right\}$ is a function that takes the value 1 when the path from v_i to v_k exists in G.

Tweet influence over a retweet cascade. We compute the influence of tweet v_i over VG – all possible diffusion scenarios associated with a retweet cascade – as:

$$\begin{aligned}\varphi(v_i) &= \sum_{G \in VG} \mathbb{P}(G)\varphi(v_i|G) \\ &= \sum_{G \in VG} \mathbb{P}(G) \sum_{v_k \in V(G)} \mathbb{P}\big(z(v_i, v_k|G)\big) \qquad (9)\end{aligned}$$

It is intractable to directly evaluate Eq. (9), plugged in with Eq. (6) and (8), particularly due to the factorial number of diffusion scenarios in VG. For example, there are 10^{156} diffusion scenarios for a cascade of 100 retweet. We develop, in the next section, an efficient linear time algorithm to compute the influence of all tweets in a retweet cascade.

B Efficient tweet influence computation

The key observation is that each tweet v_k is added simultaneously at time t_k to all diffusion scenarios constructed at time t_{k-1}. v_k contributes only once to the tweet influence of each tweet that is found on the branch it attached to. This process is exemplified in Fig. 1c. Node v_5 is added to a given diffusion scenario, generating 4 new diffusion scenarios at time t_5. We color in red the nodes whose influence increases as a result of adding node v_5. This allows to compute the tweet influence incrementally, by updating $\varphi(v_i), i < k$ at each time t_k. We denote by $\varphi^k(v_i)$ the value of tweet influence of v_i after adding node v_k. As a result, we only keep track of how tweet influence increases over time steps and we do not require to construct all diffusion scenarios.

B.1 Complete derivation of the recursive influence formula

Incremental construction of diffusion scenarios. Let $G^- \in VG^{k-1}$ be a diffusion scenario constructed at time t_{k-1}, with the set of nodes $V^- = \{v_1, v_2, \cdots, v_{k-1}\}$. When v_k arrives, it can attach to any node in V^-, generating $k-1$ new diffusion scenarios G_j^+, with $V_j^+ = V^- \cup v_k$ and $E_j^+ = E^- \cup \{v_j, v_k\}$. This process is exemplified in Fig. 1c. We can write the set of scenarios at time t_k as:

$$VG^k = \left\{G_j^+ = G^- \cup \{v_j, v_k\} \middle| \forall j < k, \forall G^- \in VG^{k-1}\right\} \qquad (10)$$

We write the tweet influence of v_i at time k as:

$$\begin{aligned}\varphi^k(v_i) &= \sum_{G^+ \in VG^k} \mathbb{P}(G^+)\varphi(v_i|G^+) \\ &\overset{cf.\ (10)}{=} \sum_{G^- \in VG^{k-1}} \sum_{j=1}^{k-1} \mathbb{P}(G_j^+)\varphi(v_i|G_j^+) \qquad (11)\end{aligned}$$

Note that in Eq. (11) we explicitly make use of how diffusion scenarios at time k are constructed based on the diffusion scenarios at time $k-1$.

Attach a new node v_k. We concentrate on the right-most factor in Eq. (11) – the tweet influence in scenario G_j^+. We observe that the terms in Eq. (7) can be divided into two: the paths from v_i to all other nodes except v_k and the path from $z(v_i, v_k)$. We obtain:

$$\varphi(v_i|G_j^+) = \sum_{\substack{v_l \in G_j^+ \\ l > i, l \neq k}} \mathbb{P}(z(v_i, v_l|G_j^+)) + \mathbb{P}(z(v_i, v_k|G_j^+)).$$

Note that a path that does not involve v_k has the same probability in G_j^+ and in its parent scenario G^-:

$$\mathbb{P}(z(v_i, v_l|G_j^+)) = \mathbb{P}(z(v_i, v_l|G^-)), \text{ for } l > i, l \neq k$$

we obtain:

$$\begin{aligned}\varphi(v_i|G_j^+) &= \sum_{\substack{v_l \in G^- \\ l > i}} \mathbb{P}(z(v_i, v_l|G^-)) + \mathbb{P}(z(v_i, v_k|G_j^+)) \\ &\overset{cf.\ (7)}{=} \varphi(v_i|G^-) + \mathbb{P}(z(v_i, v_k|G_j^+)) \qquad (12)\end{aligned}$$

Combining Eq. (11) and (12), we obtain:

$$\begin{aligned}\varphi^k(v_i) &= \sum_{G^- \in VG^{k-1}} \sum_{j=1}^{k-1} \mathbb{P}(G_j^+)\left[\varphi(v_i|G^-) + \mathbb{P}(z(v_i, v_k|G_j^+))\right] \\ &= \underbrace{\sum_{G^- \in VG^{k-1}} \varphi(v_i|G^-) \sum_{j=1}^{k-1} \mathbb{P}(G_j^+)}_{A} \\ &\quad + \underbrace{\sum_{G^- \in VG^{k-1}} \sum_{j=1}^{k-1} \mathbb{P}(G_j^+)\mathbb{P}(z(v_i, v_k|G_j^+))}_{M_{ik}}. \qquad (13)\end{aligned}$$

Tweet influence at previous time step t_{k-1}. Given the definition of G_j^+ in Eq. (10), we obtain that $\mathbb{P}(G_j^+) = \mathbb{P}(G^-)\mathbb{P}(\{v_j, v_k\})$. Consequently, part A in Eq. (13) can be written as:

$$\begin{aligned} A &= \sum_{G^- \in VG^{k-1}} \varphi(v_i|G^-) \sum_{j=1}^{k-1} \mathbb{P}(G^-)\mathbb{P}(\{v_j, v_k\}) \\ &= \sum_{G^- \in VG^{k-1}} \varphi(v_i|G^-)\mathbb{P}(G^-) \sum_{j=1}^{k-1} \mathbb{P}(\{v_j, v_k\}) \\ &= \sum_{G^- \in VG^{k-1}} \mathbb{P}(G^-)\varphi(v_i|G^-) \\ &\overset{cf.\ (9)}{=} \varphi^{k-1}(v_i) \end{aligned} \tag{14}$$

Consequently, A is the tweet influence of v_i at the previous time step t_{k-1}. Note that $\sum_{j=1}^{k-1} \mathbb{P}(\{v_j, v_k\}) = 1$ because v_k is necessarily the direct retweet of one of the previous nodes $v_j, j < k$ of the retweet cascade.

Contribution of v_k. With A being the influence of v_i at the previous time step, intuitively M_{ik} is the contribution of v_k to the influence of v_i. Knowing that:

$$\begin{aligned} \mathbb{P}(G_j^+) &= \mathbb{P}(G^-)\mathbb{P}(\{v_j, v_k\}) \text{ and} \\ \mathbb{P}\{z(v_i, v_k|G_j^+)\} &= \mathbb{P}(z(v_i, v_j|G_j^+))\mathbb{P}(\{v_j, v_k\}) \\ &= \mathbb{P}(z(v_i, v_j|G^-))\mathbb{P}(\{v_j, v_k\}) \end{aligned}$$

we write M_{ik} as:

$$\begin{aligned} M_{ik} &= \sum_{G^- \in VG^{k-1}} \sum_{j=1}^{k-1} \mathbb{P}(G^-)\mathbb{P}^2(\{v_j, v_k\})\mathbb{P}(z(v_i, v_j|G^-)) \\ &= \sum_{j=1}^{k-1} \mathbb{P}^2(\{v_j, v_k\}) \underbrace{\sum_{G^- \in VG^{k-1}} \mathbb{P}(G^-)\mathbb{P}(z(v_i, v_j|G^-))}_{M_{ij}} \\ &= \sum_{j=1}^{k-1} \mathbb{P}^2(\{v_j, v_k\})M_{ij} \end{aligned} \tag{15}$$

An alternative interpretation for M_{ik} is the influence of v_i over v_k. Eq. (15) can be intuitively understood as the expected influence of v_i over a newly attached node v_k is proportional to the influence of v_i over each already attached node v_j multiplied with the likelihood that v_k attached itself to v_j. We obtain the formula of the the expected influence of a node v_i over another node v_k as:

$$M_{ik} = \begin{cases} \sum_{j=1}^{k-1} M_{ij}\mathbb{P}^2(\{v_j, v_k\}) & , i < k \\ 1 & , i = k \\ 0 & , i > k. \end{cases} \tag{16}$$

Recursive computation of tweet influence. Using Eq (13) and (14), we can recursively compute the influence of v_i at time t_k as:

$$\begin{aligned} \varphi^k(v_i) &= \varphi^{k-1}(v_i) + M_{ik} \\ &= \varphi^{k-2}(v_i) + M_{i(k-1)} + M_{ik} \\ &= \sum_{j=1}^{k} M_{ij}. \end{aligned} \tag{17}$$

with M_{ij} as defined in Eq. (16). Intuitively, the influence of v_i at time t_k is the sum of the expected influence over each of the nodes in the diffusion scenario.

B.2 Tweet influence algorithm

We define two matrices: matrix $P = [P_{ij}]$, with $P_{ij} = \mathbb{P}^2(\{v_i, v_j\})$. Element P_{ij} of the matrix P is the square probability that tweet v_j is a direct retweet of tweet v_i; matrix $M = [M_{ij}]$, with M_{ij} defined in Eq. (16). Element M_{ij} of matrix M is the contribution of v_j to the influence of v_i. Alternatively, M_{ij} can be interpreted as the influence of v_i on v_j.

From Eq. (16) follows the formula for computing iteratively the columns of matrix M:

$$M_{.j} = \begin{bmatrix} M_{[1..j-1,1..j-1]} \times P_{[1:j-1,j]} \\ 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{18}$$

with the value 1 occurring on line j. For each column j, we compute the first $j-1$ elements by multiplying the submatrix $M_{[1..j-1,1..j-1]}$ with the first $j-1$ elements on the j^{th} column of matrix P. The computation of matrix M finishes in linear time, after n steps, where n is the total number of retweets in the retweet cascade. Fig. 9 demonstrated the computation of the first four columns in M. Algorithm **??** gives an overview of the efficient influence computation algorithm.

C Generation of synthetic data

This section completes and details the results concerning the evaluation on synthetic data, presented in the main text Sec. 5.1. This section details the construction of a synthetic random social graph (Sec. C.1) and the sampling of synthetic cascades (Sec. C.2). The purpose is to construct a synthetic dataset of cascades, in which the user influence ground truth is known. Both the graph and cascade generators described here below reproduce closely the synthetic experimental setup described in (Du et al. 2013).

C.1 Generation of random graphs

In this section, we describe the construction of a synthetic social graph, with n nodes, specified by its adjacency matrix M. Each node corresponds to a synthetic user, and the edges correspond to synthetic follow relations. We follow the below steps:

- Given the number of nodes n in the graph (here 1000), create an null (all zero) adjacency matrix M of size $n \times n$;
- Randomly choose the number of edges $|E|$, between $n/2$ to n^2.
- Randomly choose i and j between 1 to n and set M_{ij} to 1. Iterate this step until $|E|$ different edges are generated.
- The adjacency matrix M defines the final random graph.

$$\underbrace{\begin{bmatrix} 1 & M_{12} & M_{13} & M_{14} & \cdots & M_{1n} \\ 0 & 1 & M_{23} & M_{24} & \cdots & M_{2n} \\ 0 & 0 & 1 & M_{34} & \cdots & M_{3n} \\ 0 & 0 & 0 & 1 & \cdots & M_{4n} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}}_{\mathbf{M}} \underbrace{\begin{bmatrix} 1 & P_{12} & P_{13} & P_{14} & \cdots & P_{1n} \\ 0 & 1 & P_{23} & P_{24} & \cdots & P_{2n} \\ 0 & 0 & 1 & P_{34} & \cdots & P_{3n} \\ 0 & 0 & 0 & 1 & \cdots & P_{4n} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}}_{\mathbf{P}}$$

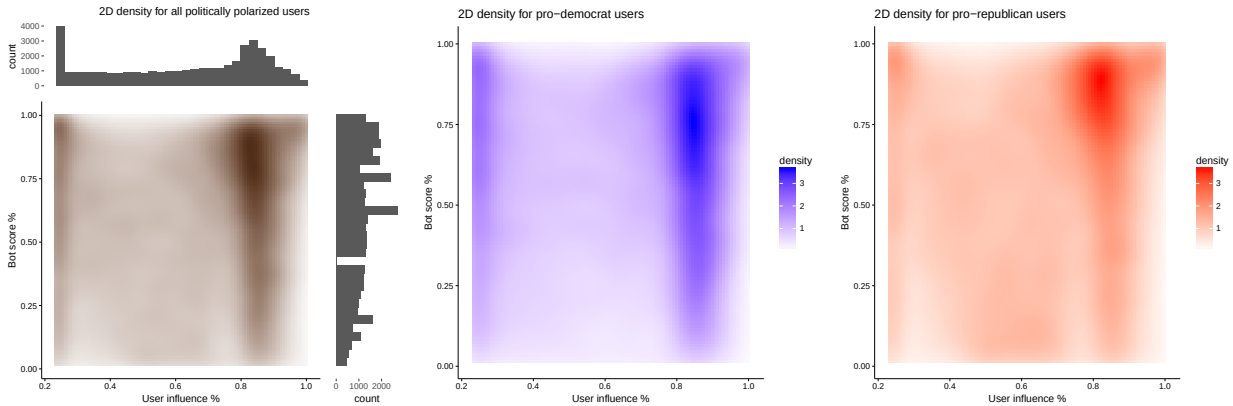
$$M_{11} = 1 \qquad k = 1$$

$$M_{12} = P_{12}M_{11} \qquad k = 2 \quad [P_{12}]\,[M_{11}] = [M_{12}]$$

$$\left.\begin{aligned} M_{13} &= P_{13}M_{11} + P_{23}M_{12} \\ M_{23} &= P_{13}M_{21} + P_{23}M_{22} \end{aligned}\right\} \quad k = 3 \quad \begin{bmatrix} 1 & M_{12} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} P_{13} \\ P_{23} \end{bmatrix} = \begin{bmatrix} M_{13} \\ M_{23} \end{bmatrix}$$

$$\left.\begin{aligned} M_{14} &= P_{14}M_{11} + P_{24}M_{12} + P_{34}M_{13} \\ M_{24} &= P_{14}M_{21} + P_{24}M_{22} + P_{34}M_{23} \\ M_{34} &= P_{14}M_{31} + P_{24}M_{32} + P_{34}M_{33} \end{aligned}\right\} k = 4 \quad \begin{bmatrix} 1 & M_{12} & M_{13} \\ 0 & 1 & M_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P_{14} \\ P_{24} \\ P_{34} \end{bmatrix} = \begin{bmatrix} M_{14} \\ M_{24} \\ M_{34} \end{bmatrix}$$

Figure 9: Exemplification of the efficient computation of the first four columns of matrix M. Each k^{th} column vector of matrix M is colored correspondingly with the column vector in P used in the multiplication.



(a) (b) (c)

Figure 10: 2D density plots for all politically polarized users in #DEBATENIGHT **(a)**, for Democrat users **(b)** and for Republican users **(c)**.

C.2 Sampling synthetic cascades

In this section we describe how to construct synthetic cascades, given a synthetic social graph G constructed as shown in Sec. C.1. To generate one cascades, to each edge in G we associate an exponentially distributed waiting time and we construct a shortest path tree. We detail this procedure in the following steps:

- Similar to previous work (ConTinEst (Du et al. 2013)), for each edge $\{v_i, v_j\} \in G$ draw a transmission rate r_{ij} from a Weibull distribution of shape $k = 2$.
- Given the transmission rate r_{ij}, we draw an Exponentially distributed waiting time τ_{ij} using the inverse transform sampling:
$$\tau_{ij} = -\frac{ln(s)}{r_{ij}}$$
where s is draw uniformly from $U(0, 1)$.
- Set τ_{ij} as the weight of edge $\{v_i, v_j\}$.
- Starting from a source node v_s, construct the shortest path tree from v_s to all the other nodes in G;
- For each node v_k compute two measures: t_k – its time of occurrence as the total waiting time along the path from v_s to v_k – and c_k – the number of reachable nodes from v_k in the shortest path tree;
- The generated cascade is $\{t_1, t_2, t_3 \cdots t_n\}$ where $(t_1 < t_2 < t_3 < \cdots t_n)$;
- The ground truth influence of node v_k is the mean c_k over multiple random graphs (here 100).

D Choosing the temporal decay parameter

The temporal decay parameter r shown in Eq. (1) is determined by linear search. Eq. (1) measures the probabilities of edges in the retweeting (diffusion) network, however the real diffusions are not observed. We make the assumption that diffusions occur along edges in the underlying social graph (follower relation). We measure the fitness of edge probability by the likelihood of uncovering the ground truth follower graph. In other words, if the edge $\{v_i, v_j\}$ exists in the follower graph (i.e. v_j is a follower of v_i) then the edge $\{v_i, v_j\}$ has the highest probability in the diffusion tree than the edge $\{v_l, v_j\}$ which does not exist in the following graph. In other words, we are using the retweeting probability to predict the existence of edges in the social graph. We randomly select 20 cascades, and we crawl the following list of every user appearing the the diffusions (the following list for a user u_j consists of users u_i followed by u_j). We use this information as ground truth for the following prediction exercise: given a user u_j who emitted tweet v_j in a particular cascade, we want to predict which among the users in the set $\{u_1, u_2, \ldots, u_{j-1}\}$ are followed by u_j. Considering that $\mathbb{P}(\{v_l, v_j\}).\forall l < j, j \geq 2$ are real numbers and the prediction target is binary, we use the AUC (are under ROC curve) the measure the prediction performance of a particular probability scoring function. For each value of r in Eq. (1) we compute the mean AUC over all predictions. We perform a linear search for the optimal r between 10^{-8} to 3. Finally, 6.2×10^{-4} maximizes the mean of AUC and it is chosen as r value in the experiments in Sec. 5 and 6.

E Additional 2D densities plots

We show in Fig. 10 the additional 2D density plots mentioned in the main text Sec. 6.2.