Applied Network Science

RESEARCH Open Access

On the challenges of predicting microscopic dynamics of online conversations

John Bollenbacher*, Diogo Pacheco, Pik-Mai Hui, Yong-Yeol Ahn, Alessandro Flammini and Filippo Menczer

*Correspondence:
jmbollen@iu.edu
Center for Complex Networks and Systems Research, Indiana University, Bloomington, USA

Abstract

To what extent can we predict the structure of online conversation trees? We present a generative model to predict the size and evolution of threaded conversations on social media by combining machine learning algorithms. The model is evaluated using datasets that span two topical domains (cryptocurrency and cyber-security) and two platforms (Reddit and Twitter). We show that it is able to predict both macroscopic features of the final trees and near-future microscopic events with moderate accuracy. However, predicting the macroscopic structure of conversations does not guarantee an accurate reconstruction of their microscopic evolution. Our model's limited performance in long-range predictions highlights the challenges faced by generative models due to the accumulation of errors.

Keywords: Information cascades, Conversation trees, Prediction, Social media, Machine learning

Introduction

Can we predict whether a social media post will attract a large number of reposts or responses? Is it possible to predict the structure of the information cascade triggered by a specific post? Many models have been proposed to predict the virality of individual pieces of information (Weng et al. 2013; Cheng et al. 2014; Zhao et al. 2015; Li et al. 2017; Subbian et al. 2017; Kefato et al. 2018). Other approaches have aimed at *generating* ensembles of online conversations, providing population-level insights about how and why information spreads (Kumar et al. 2010; Wang et al. 2012; Gómez et al. 2013; Nishi et al. 2016; Lumbreras et al. 2017; Aragón et al. 2017b). This manuscript presents work that lies at the intersection of these two approaches. We propose a platform-agnostic generative model to grow *conversation (cascade) trees* that focuses on predicting each microscopic event (e.g., a comment or a retweet). The model is based on a supervised machine learning approach that leverages information about the tree generated up to a given point.

We conceptualize the conversation structures in various platforms as trees (Fig. 1). For instance, on Reddit, a post may attract comments, and each of these comments may induce further threaded conversations. On Twitter, similar conversation trees can be constructed from a more diverse set of interactions, such as retweets and replies.

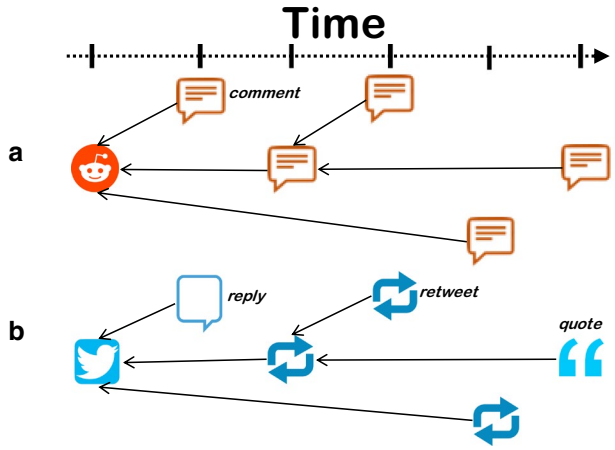

Fig. 1 Examples of conversation trees on **a** Reddit—a post followed by comments, and on **b** Twitter—a tweet followed by retweets, replies, and/or quotes. Link directions represent attention toward previous posts

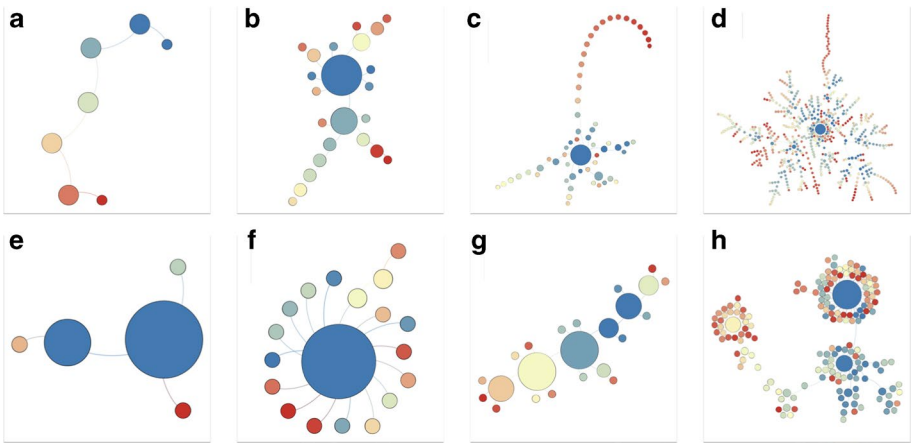

Fig. 2 A few example conversation trees from **a–d** Reddit and **e–h** Twitter. Each node represents a comment, retweet, or reply. Node size represents popularity (number of children) and node color represents temporal order in the tree sequence, from the root (blue) to the last leaf (red)

We propose a generative model for predicting the size and structure of conversation trees on social media, which first predicts the final size of the tree and then iteratively adds nodes to the tree until it reaches the predicted size. To evaluate our model, at each point in a conversation history, we measure the likelihood of correctly predicting which post will be replied/shared/commented upon next. We also assess the accuracy of our method by measuring the relative deviation in terms of depth, breadth, structural virality, and size of our simulated trees from those in the ground-truth data.

We test our model on Reddit and Twitter conversation datasets, made available as part of a challenge run by the Defense Advanced Research Projects Agency (DARPA). The conversation trees in the dataset have diverse sizes and structural features, as illustrated in Fig. 2. While our method outperforms a simple baseline and is competitive with a state-of-the-art baseline, its accuracy dramatically decreases for longer sequences of

microscopic predictions due to accumulating errors. These results highlight the challenges of detailed reconstruction of online diffusion networks (Goel et al. 2012).

In the next section we review relevant studies about both the prediction of important features of information cascades and the simulation of their evolution. The prediction task is defined and discussed in "Problem and models" section, along with our model and the baselines adopted for evaluation. "Feature description" and "Datasets" sections provide details about the features used by our model and the data used for training and testing. The results are presented in "Results" section and discussed in "Discussion" section.

Related work

Cascades have been characterized on many platforms (Gómez et al. 2008; boyd et al. 2010; Rossi and Magnani 2012; Dow et al. 2013; Weng et al. 2013, 2014; Weninger 2014; Choi et al. 2015; Hui et al. 2018). These studies revealed common properties among popular/viral conversations, suggesting the possibility of predicting viral content. Despite such efforts, the prediction of virality remains elusive, probably due to the intrinsic randomness of popularity (Salganik et al. 2006; Goel et al. 2012).

Some machine learning methods aim to predict macroscopic characteristics of individual cascades (mainly the size) from features observable at an early stage. Backstrom et al. (2013) identified features that describe novelty, arrival patterns, textual expression, and social influence to predict the size of media sharing cascades. Cheng et al. (2014) formalized size prediction as a binary classification problem, achieving an accuracy of 80% at predicting whether the number of shares of a photo on Facebook would double. Li et al. (2017) proposed DeepCAS, an end-to-end deep learning approach that leveraged raw data to directly predict logarithmic increments of the size of Twitter cascades and paper citations within finite time windows (1, 3, and 5 days). Using data from Twitter and Facebook, Subbian et al. (2017) proposed SansNet, an approach based on survival analysis (Klein and Moeschberger 2006) that predicts whether cascades will become viral. Their definition of virality is based on percentiles — the largest 0.5% trees are considered viral. Beck et al. (2019) applied various machine learning models to monitor cryptocurrency news and predict the number of article mentions given their Twitter activity in the previous day. Other methods leverage node characteristics (Pei et al. 2014), network structure (Weng et al. 2013), temporal patterns (Pinto et al. 2013), or a combination of them (Guo et al. 2015).

Generative approaches aim at identifying underlying mechanisms that can reproduce stylized traits of cascades. Aragón et al. (2017a) published an extensive survey of approaches of this kind. Here we mention just a few to highlight their main ingredients, beyond what one may expect from a pure *branching process à la* Galton–Watson (1875). Kumar et al. (2010) combined the concept of *novelty* with preferential attachment. Wang et al. (2012) explored reaction times and lifespans of cascades by continuous-time modeling of dynamics. Gómez et al. (2013) added a *root bias* to popularity and novelty, recognizing the special role that the original post plays in the spreading process. Aragón et al. (2017b) built upon this model by adding *reciprocity* among users. They focused on how the structural features of conversations are affected by their presentation in a platform's interface. Nishi et al. (2016) observed that the structural diversity observed

among cascades was not described adequately by previous generative models. Such a diversity was manifest in the co-existence of deep conversation-like threads, shallow star-like trees, and irregular patterns. To reproduce the conversation-like patterns, they introduced the concept of *segments*, long chains of posts with little or no branching involved. Lumbreras (2016); Lumbreras et al. (2017) were able to capture different *user roles* in the cascade formation, and to use this information to predict user engagement.

Some deep learning approaches model a cascade by predicting the next action and its timestamp. The next action could be one of a given set of events, or one of a given set of users. CAS2VEC (Kefato et al. 2018) produces cascades as sequences of event timestamps. The methods proposed by Wang et al. (2017) and Islam et al. (2018) predict when users in a known social graph would participate in a cascade. Du et al. (2016) applied recurrent neural networks and temporal point processes to predict sequences of events such as stock-market orders and taxi trips. In these approaches, time is the most important component being predicted; the tree structure is not reconstructed.

Point-process models can simulate cascades by fitting parameters to historical data (Shen et al. 2014; Gao et al. 2016; Mishra et al. 2016; Cao et al. 2017; Rizoiu et al. 2017). These models can be used to predict the entire tree structure, and have been successfully applied to the study of Twitter cascades (Kobayashi and Lambiotte 2016) as well as conversation threads on Reddit (Medvedev et al. 2019; Krohn and Weninger 2019). SEISMIC (Zhao et al. 2015) predicted the final size of a retweet cascade with a 15% relative error on average after observing the first hour of the cascade. Most point-process models cannot work from just the initial post, and so they are not directly comparable to ours. An exception is the Cascade Tree Prediction Model (CTPM) proposed by Krohn and Weninger (2019), which uses a *Hawkes process* to predict the full tree structure from only the initial post by cleverly inferring the necessary model parameters from similar posts in the training data. To our knowledge, CTPM is the only model directly comparable to the one presented here, and therefore we use it as a baseline in our evaluation.

Problem and models

We represent a social media conversation as a tree of messages, with the root node being the initial post, and all subsequent comments, replies, or shares being additional nodes in the tree. Given some initial condition of a conversation, one goal is to predict its final *macroscopic* state, i.e., the size and structure of its tree. In addition, we aim to predict the longitudinal evolution of a cascade tree as a sequence of *microscopic* states.

We propose a Tree Growth Model (TGM) that starts from a partial tree containing the first k nodes, as well as information about the root post such as its textual content. TGM breaks the tree prediction problem into two sub-tasks: (1) predicting the final size of the tree based on the initial information given ("Size prediction" section); and (2) predicting what node in the tree the next node will attach to ("Node placement" section). TGM grows the conversation by attaching nodes (messages) until the tree reaches the predicted size. For each sub-task, we develop a model and propose a baseline for evaluation.

Size prediction

We use regression to predict the final size of the tree based on features of the initial partial tree, which could be as minimal as a single first post (see "Feature description"

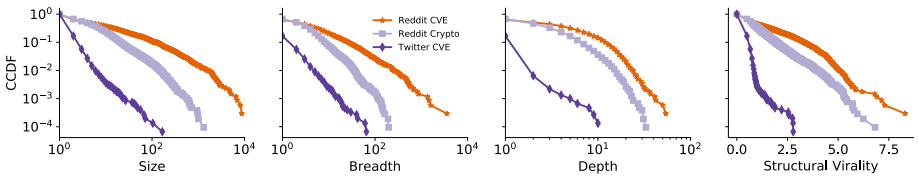

Fig. 3 Complementary Cumulative Distributions (CCDF) of cascade size, breadth, depth, and structural virality within each dataset

Table 1 Mean Relative Error in the size prediction problem by different regression models across our three datasets (cf. "Datasets" section), for $k = 2$ and $k = 10$

Classifier model	Reddit				Twitter	
	Crypto		CVE		CVE	
	$k = 2$	$k = 10$	$k = 2$	$k = 10$	$k = 2$	$k = 10$
Random Forest	0.77	0.46	1.4	0.84	0.22	0.41
Gradient Boosting Trees	0.79	0.43	1.6	0.81	0.25	0.45
KNN Regression	0.84	0.48	1.8	1.0	0.24	0.39
Linear Regression	1.9	0.88	1.9	0.88	0.36	2.4
Elastic Net	0.88	0.46	1.6	0.89	0.16	0.42

section). The distribution of tree size is heavy-tailed, as shown in Fig. 3. Since it is difficult for a regression method to predict a quantity with such a broad distribution, we instead predict the size-quantile of the tree and then map that quantile back to a size using the empirical size distribution. This approach, similar to quantile regression (Koenker and Bassett 1978), allows our method to deal with imbalanced data without resampling large trees or excluding small ones.

We evaluated several regression models for the size prediction task, using cross validation within the training set. We selected Random Forest regression, which provided the best performance (see Table 1).

We trained a different regression for each initial tree size k (for $1 \leq k \leq 10$), and trained each model based on the trees in the training data whose final size is larger than k. This performed better than simply using k as an additional feature in the regression.

We also considered an alternative model in which the size is not decided at the outset, but a decision whether to stop or continue is made at each step. However, without introducing strong assumptions and biases, this approach leads to unrealistic, exponentially narrow tree size distributions.

Node placement

Once we predict the final size, we iteratively attach additional nodes to the current conversation tree. To predict where the next node will be attached, we train an estimator to assign a likelihood score to each node in the current tree. We then draw a random node with probability proportional to its score. The selected node becomes the parent of the newly attached node. The process is illustrated in Fig. 4.

The node classifier takes in a set of features from a given node and the tree, and estimates the node's likelihood of becoming the parent of the next node in the tree.

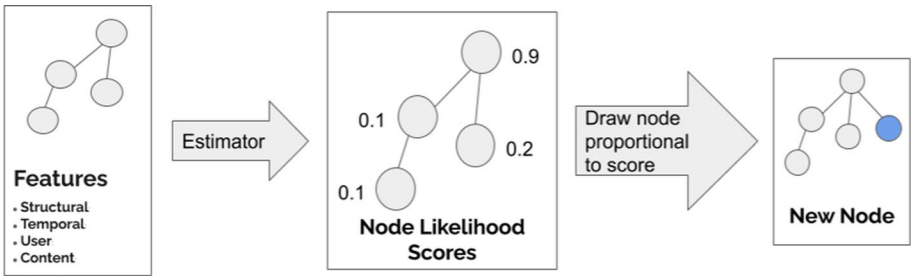

Fig. 4 Diagram illustrating how nodes are added to the tree

Table 2 AUC performance by various classifiers on the node placement problem

Classifier model	Reddit		Twitter
	Crypto	CVE	CVE
Gradient Boosting Trees	0.91	0.85	0.97
Random Forest	0.88	0.75	0.95
Multilayer Perceptron	0.80	0.53	0.78
KNN Classifier	0.54	0.61	0.92

We considered various machine learning models for this task. As shown in Table 2, the Gradient Boosting Trees classifier achieved the best accuracy, as measured by the Area Under the ROC Curve (AUC) across our three datasets (cf. "Datasets" section). AUC captures the trade-off between maximizing the true positive rate and minimizing the false positive rate.

Based on this analysis, we selected the Gradient Boosting Trees model for the node placement task. The classifier outputs a score between 0 and 1, such that if the score is closer to 1, the classifier estimates that the node is more likely to receive the next reply in the cascade. For details on the features we used and how the training set for this classifier was constructed from the empirical tree data, see "Feature description" section and Appendix "Extracting node placement classifier features".

Treating the raw classifier score as a probability (as is often done) does not yield optimal results. Instead, we apply a Bayesian correction to the raw classifier output to obtain each node's likelihood estimate. Specifically, given a node with classifier score S, let A be the case where the node receives the next reply, and $\neg A$ the case where the node does not receive it. Then we can compute the conditional probability that the node will receive the next reply given its score using Bayes theorem, $P(A \mid S = s) = P(S = s \mid A)\, P(A) \,/\, P(S = s)$.

We compute the values on the right hand side of the equation from our training data and the classifier scores for the nodes in the training data. We set $P(A) = N_A/N$, where N_A is the number of nodes in the training dataset that received the next reply and N is the total number of nodes in the training dataset. We compute $P(S = s) = P(S = s \mid A) + P(S = s \mid \neg A)$. Finally, both $P(S = s \mid A)$ and $P(S = s \mid \neg A)$ are given by Gaussian Kernel Density Estimates produced from the the set of scores for nodes in our training data with condition A and $\neg A$, respectively.

Baseline models

Let us introduce two baseline models. One is a simple model that follows our approach with size prediction and next-node placement sub-tasks. The other baseline is a state-of-the-art method for predicting the full tree structure.

Random baseline A *random size baseline* model draws a random size for each prediction from the empirical distribution of sizes with replacement. We then use a simple *random choice baseline* for node placement, in which all nodes are equally likely to receive the next reply. Let j be the time-ordered index of existing nodes, and let n be the number of existing nodes in the tree. The probability of node x_j being the parent of the next node is given by $P(x_j) = \frac{1}{n}$.

CTPM baseline The *Cascade Tree Prediction Model* uses Hawkes processes to reconstruct the structure of the cascade tree. While the details are found elsewhere (Krohn and Weninger 2019), here we explain the intuition behind the approach. Two distributions are fit, one for the root node and one for the other nodes, which determine branching factors at each node. CTPM infers the parameters of these distributions for each test post from similar posts in the training set. For each node, the number of children is drawn from its distribution, and the tree is constructed in a breadth-first way. The size is implicitly determined when the branching process stops after leaf nodes draw zero values for their numbers of children.

Feature description

Our models use four classes of features: structural, user, content, and temporal features. Each of these classes is described next. Table 3 outlines the features in each class. Appendix "Extracting node placement classifier features" discusses how the features are extracted, while Appendix "Feature importance" analyzes the importance of different features.

Structural features

We use two types of structural features for node placement and size regression. First, there are features about the current tree, including the initial size of the tree, its depth, and statistics about the placement of individual nodes (e.g., how many attached to the root node). Second, for the node placement task, there are features about the individual node, including its depth, how many children it has, how many siblings it has, and more.

Most of these features have standard definitions and are easily interpreted, but some are not. In particular, we introduce a *normalized delay steps since parent* measure, or simply the *parent delay*. It is an edge attribute capturing the relative difference in steps (actions) between a new node and its parent. Let the index of node i be ordered by the time of creation of the node. Then the parent delay of node i, d_i, is given by

$$d_i = \frac{(i - \pi(i)) - 1}{i - 1} \tag{1}$$

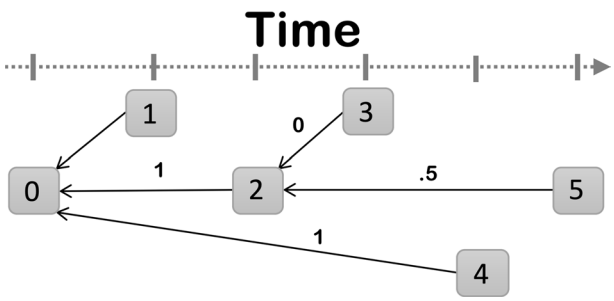

Fig. 5 Representation of the conversation trees in Fig. 1 with *parent delays* (see Eq. 1) as edge weights. Each node links to its parent. Note the parent delay is undefined for the edge between nodes 0 and 1

where $\pi(i)$ is i's parent ($i > \pi(i)$) and $d_i \in [0, 1]$. The root index is 0, so the parent delay of the second node attached to the tree is undefined. That is, d_i is defined only for the second edge (third node) and on. This measure aims to capture the *root bias* and the *novelty* features in prior work (Gómez et al. 2013; Lumbreras 2016; Aragón et al. 2017a). When a node is connected to the root, this edge is the most delayed possible $d_i = 1$. Conversely, attaching a node to the most recent node previously added to the tree gives the shortest delay $d_i = 0$. Figure 5 shows a conversation tree and the parent delay for each edge. We derive several features from this measure, including the mean and median delay, and the percentage of edges connecting to the root and to the most recent node.

User features

User features are derived only from the author of the conversation's initial post, or root. Since our approach is generative, incorporating information about other users would require the model to also associate users to each new node during the tree growth. We focus on the tree structure replication and leave user attribution for future work.

We extract structural properties of the conversations initiated by a user, such as max breadth, depth, and size (see Table 3 for the full list). The user features are the median and the mean of these properties.

Content features

As with user features, the content features are also limited to the root of the conversation, e.g., a Reddit post title or a Twitter text. Using any intermediate content would require the generative model to also produce content along the simulation, a task beyond the scope of this work.

We use *fastText*, a document embedding method developed by Facebook Research (Joulin et al. 2017), to represent the content of a conversation's root post as a 25-component feature vector used by the size regression model and the node placement classifier.

Temporal features

We use various temporal features, including the time of day and day of week of the root post, and the mean time between replies in a conversation (see Table 3 for the full list). Temporal features prove very valuable for the size prediction task. However, we do not use temporal features for the node placement task, because this would introduce too

Table 3 Description of features, grouped by class, showing if they were used in the size prediction regression model, the node placement classifier, or both

Feature	Task	Description
Global		
Current size	Node	Number of nodes and its log
Depth	Both	Conversation depth and its log
Root edges	Size	Percentage of edges connected to the root
Recent edges	Size	Percentage of edges connected to the most recent node (the last one) at each insertion
Parent delay	Size	Mean and median parent delay of nodes (cf. "Structural features" section)
Node		
Node index	Node	Time-ordered index of given node
Node depth	Node	Depth of given node
Node parent delay	Node	Parent delay of given node (cf. "Structural features" section)
Subtree size	Node	Size of subtree of given node
#Children	Node	Number of children of given node
#Grandchildren	Node	Number of grandchildren of given node
#Siblings	Node	Number of siblings of given node
#Cousins	Node	Number of cousins of given node
#Uncles	Node	Number of uncles of given node
User (mean and median over conversations initiated by the user)		
Max breadth	Node	Maximum breadth
Depth	Node	Conversations depth
1st Breadth	Node	Number of comments at the first level (depth=1)
Lifetime	Node	Conversation lifetime in seconds
Size	Node	Number of nodes in a conversation tree
Root edges	Node	Percentage of edges connecting to the root
Recent edges	Node	Percentage of edges connecting to the most recent node (the last one) at each insertion
#Posts	Node	Total number of posts by user
Content		
Root text	Both	Title of the Reddit post or Twitter text embedded in a 25-component vector
Temporal		
Response time	Size	Mean and median time between replies
1st Response Time	Size	Time between root post and first reply
Root time of day	Both	Time of day of root post
Root day of week	Both	Day of week of root post

much complexity, requiring our model to generate a time stamp for each new node as it is added to the tree.

Datasets

We use three social media cascade datasets that were made available as part of the DARPA SocialSim (DARPA 2018) December 2018 Challenge. Two originate from Reddit and the third from Twitter. All are anonymized. The Twitter dataset is a collection of public tweet cascades that contain Common Vulnerabilities and Exposure (CVE) codes and related keywords. Since retweets are always connected to the root tweet in the raw data, we use the follower data and timestamps to reconstruct the conversations

Table 4 Statistics about the three cascade datasets

Statistic	Reddit		Twitter
	Crypto	CVE	CVE
Cascades in training set	10,000	2911	10,000
Cascades in test set	500	500	5000
Median size in training (Test) set	3 (3)	3 (9)	1 (1)
Mean size in training (Test) set	12 (5)	73 (97)	1 (1)

(Goel et al. 2016; Pacheco 2019). One Reddit dataset was collected using the same CVE keywords as the Twitter dataset, while the other Reddit dataset consisted of cascades containing keywords related to several cryptocurrencies, including names, aliases, and cashtags of the coins.[1] Table 4 presents a statistical summary of the datasets.

For each dataset, we held out a number of the most recent cascades as the test set and used the rest for training. For the two Reddit datasets, we used 500 cascades for the test set. For Twitter, we used 5000 cascades because most cascades were trivial (containing only one tweet), so we needed a greater number and diversity of cascades to test our method's performance. For the Reddit Crypto and Twitter CVE datasets, we used 10,000 cascades for the training set. We decided to limit our training set to 10,000 cascades because extracting the features from these cascades during the training process was computationally expensive. For Reddit CVE, we were limited to 2911 cascades in the training set because the dataset only contains a total of 3411 cascades.

We also considered the use of some publicly available datasets that are often used to predict cascades of votes on the Digg news platform (Hogg and Lerman 2012), quote/phrase memes on the Web (Leskovec et al. 2009), and retweets on Twitter (Hodas and Lerman 2014). However, none of these datasets have a minimal set of features used in our model. In particular, they do not contain usable cascade tree structures or lack the information needed to reconstruct the tree structure of each cascade. Digg cascades are all trivial stars because all users in a cascade are voting for the same post, not creating a conversation tree. In addition, the Twitter and Digg datasets lack text content.

Evaluation

We evaluated our models for each sub-task, size prediction and next-node placement, as well as for the combined task of predicting the size and structure of whole conversation trees. In this section we describe our evaluation methods.

Size prediction

Given a partial conversation tree with k initial nodes and information about the initial post (post content, time of day, etc.), our model predicts the final size of the tree. We adopt the *Relative Error* as our error measure, calculated as $\frac{|s-\hat{s}|}{|s|}$, where s is the true size and $\hat{s}$ is the estimated size of the tree. We report the *Mean Relative Error* (MRE) over the set of trees in our test data.

[1] A *cashtag* is a unique identifier of a cryptocurrency, e.g., `$BTC` for Bitcoin.

Node placement

We evaluated the model's ability to predict where to place the next node and compared it to the random choice baseline model for node placement. Specifically, let $\tau(n)$ be the n-th stage in the construction of tree τ (n is the current size of tree τ). Each node i in $\tau(n)$ is assigned a probability $p(i_n)$ that it will be the parent of the next node. If i_n^* is the correct next node from the empirical data, we use $p(i_n^*)$ as our accuracy metric for this task because that is the probability of making the correct choice at step n in the evolution of the tree. This probability is averaged across all the steps across all the trees within our evaluation dataset.

For the random choice baseline, the probability of placing the next node correctly in a tree with n nodes is given by $p(i_n^*) = \frac{1}{n}$. To compare the models more easily, we plot the mean probabilities normalized by the baseline model, and call this metric *Relative Performance* in the next-node placement task.

Whole-tree simulation

To test the ability of our Tree Growth Model to grow entire conversation trees, we start from a partial tree with k initial nodes and information about the initial post (content, time of day, etc.), and ask TGM to predict the size and structure of the final state of the conversation tree. First, the model predicts the final size of the tree using our size prediction model. Second, we add one node at a time to the tree using our node placement model, until we reach the predicted size of the tree.

We evaluate the performance of TGM using macroscopic measures of tree structure, namely depth, breadth, and structural virality. The latter, also known as the Wiener index, is defined as the average distance between all pairs of nodes in the tree (Goel et al. 2016). We measure how these quantities depend on k and the final tree size.

Cumulative node placement

To assess the effects of cumulative errors in consecutive node placement decisions, we also compute the probability of placing all of the next t nodes correctly, starting from an initial tree of size k. For the random choice baseline, the probability of placing t consecutive nodes correctly is $\prod_{n=k}^{k+t-1} \frac{1}{n}$. Similarly for our TGM model, the probability of placing t consecutive nodes correctly is given as $\prod_{n=k}^{k+t-1} p(i_n^*)$, i.e., the product of the probabilities of getting each node placement correct for nodes $k+1$ to $k+t$. For CTPM, this calculation is infeasible because the probabilities $p(i_n^*)$ would have to be estimated by many partial simulations starting from each stage τ_n of every tree.

We explore how these probabilities depend on the parameters k and t. Because the probabilities end up being very small for large values of t, we plot the log-probability of each model, and use log-ratios to compare the models.

Results

Size prediction

For the size prediction task, our regression model achieved substantially better Mean Relative Error than the random baseline model on all three datasets (see Fig. 6).

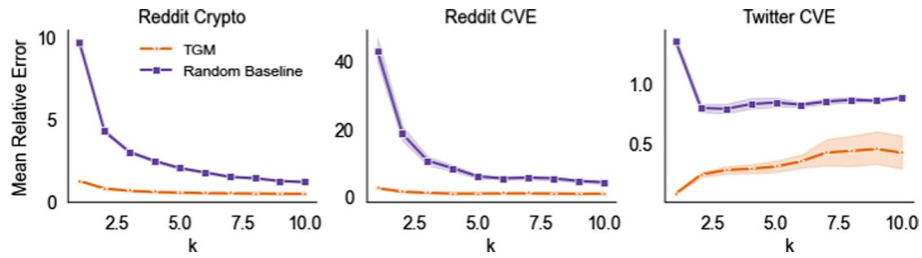

Fig. 6 Mean Relative Error of size prediction versus number of nodes k in the given initial tree. Shaded areas in this and the following plots indicate 95% confidence intervals

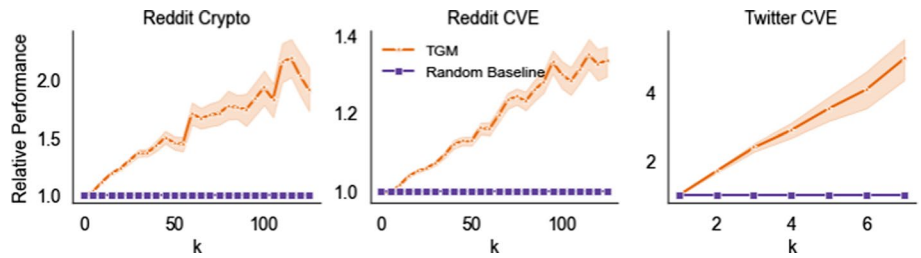

Fig. 7 Probability of placing the next node correctly, normalized by the random baseline model. Reddit Crypto and Reddit CVE results are binned over k (bin width $\Delta k = 5$) to improve readability and reduce noise

In both Reddit datasets, the error monotonically decreases as k increases; this is expected because as k increases, the regressor has more information about each cascade in the dataset. However, for the Twitter dataset, the error increases with k for both models. We attribute this to training data sparsity. As k increases, the number of potential targets for new nodes increases while the number of examples decreases: there are far fewer Twitter cascades of size 10 than of size 2. Consequently, the regression model has too few examples to learn from, and the random baseline model fails to adapt to the change in initial tree size.

Node placement

We computed the probability of choosing the correct next node, given the true tree up to each point. TGM achieves substantially improved performance compared to the random choice baseline (see Fig. 7).

Whole-tree simulation

To asses the ability of the Tree Growth Model to predict the size and macroscopic structure of whole conversation trees, we grow entire trees and measure their depth, breadth, and structural virality. For each dataset, we plot the Mean Relative Error in the predictions of these structural properties against the initial tree size k and the final size of the ground truth trees. Here k is a proxy for the amount of initial information given to the model about the trees it is supposed to generate.

We observe in Fig. 8 that in general, TGM outperforms the random baseline model for all tested values of k on Reddit datasets, but struggles with larger k values for the Twitter dataset. This may be due to data sparsity; only about 1% of the trees in the training

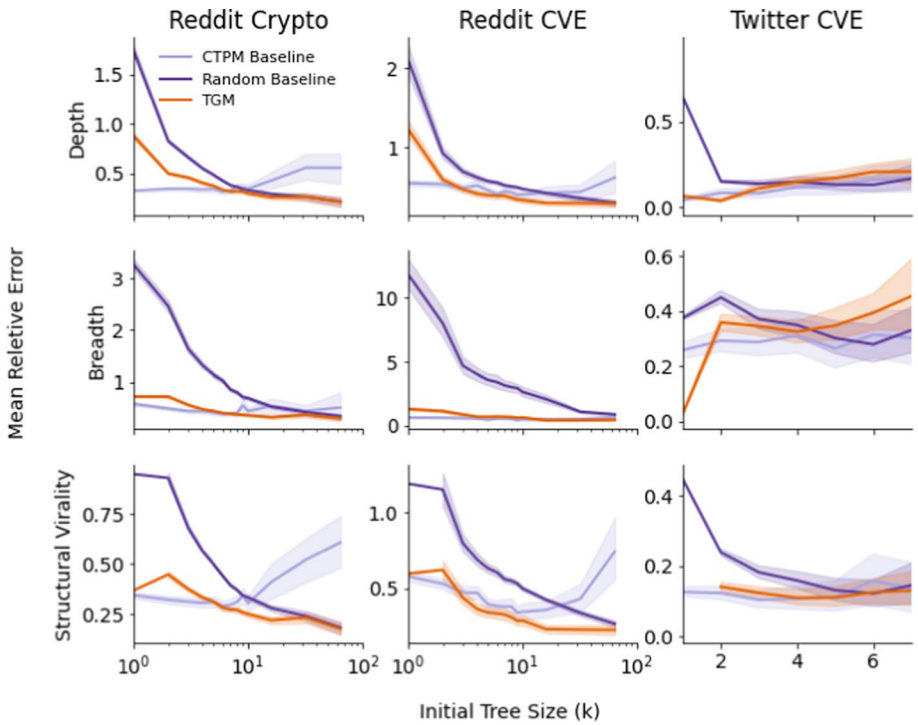

Fig. 8 Mean Relative Error of whole-tree metrics (depth, breath, and structural virality) versus initial size k. For the Twitter CVE dataset, TGM predicts all trees to have size one when $k = 1$, so that the structural virality is undefined

set have size larger than 5. Similarly, for $k = 64$ (the largest value tested here), the Reddit training sets are also sparse, hence the errors converge to near the random baseline model.

The state-of-the-art CTPM baseline typically outperforms TGM for small k values on Reddit datasets due to its clever parameter inference method. We also find that CTPM performance typically degrades with increasing k, while TGM typically improves with k. On the Twitter dataset, there is less of a difference between the performance of TGM and CTPM.

Figure 9 plots errors against final tree size. Our model generally outperforms the random baseline. Again, it performs best for smaller trees, for which we have more training data, and converges to errors near those of the random baseline model for larger trees. The CTPM baseline typically outperforms TGM for small trees on Reddit datasets, whereas TGM outperforms CTPM for medium and large trees. Performance is comparable on the Twitter dataset.

Cumulative node placement

Having found that our model performs well in reproducing the macroscopic structure of conversation trees, let us examine its accuracy in predicting their microscopic evolution. Since it is infeasible to compare TGM with CTPM on this task (cf. "Cumulative node placement" section), we only report on results comparing TGM against the random baseline model.

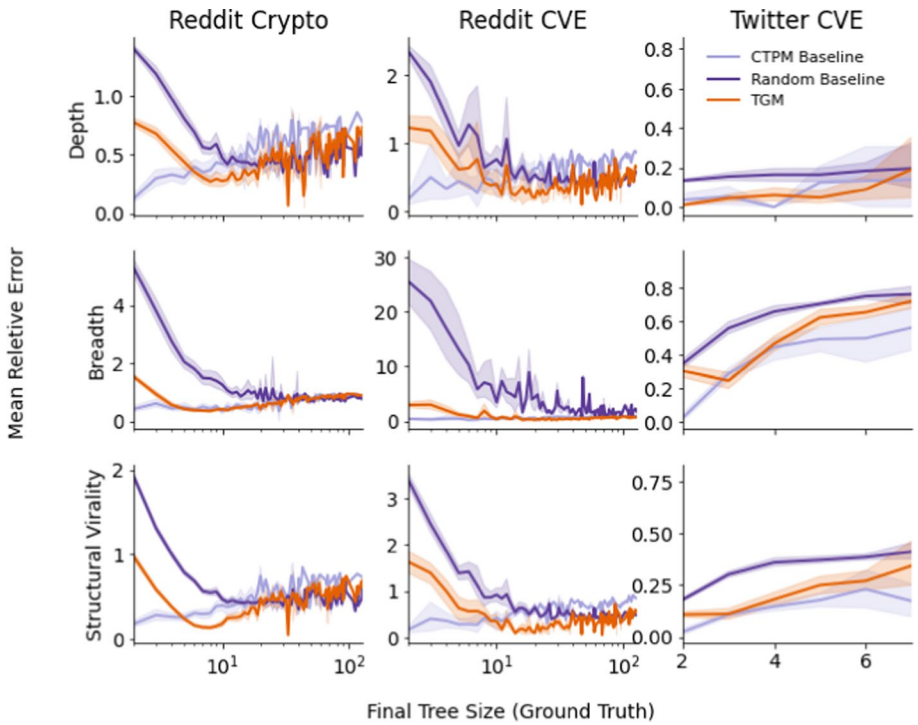

Fig. 9 Mean Relative Error of whole-tree metrics (depth, breath, and structural virality) versus final tree size of ground truth

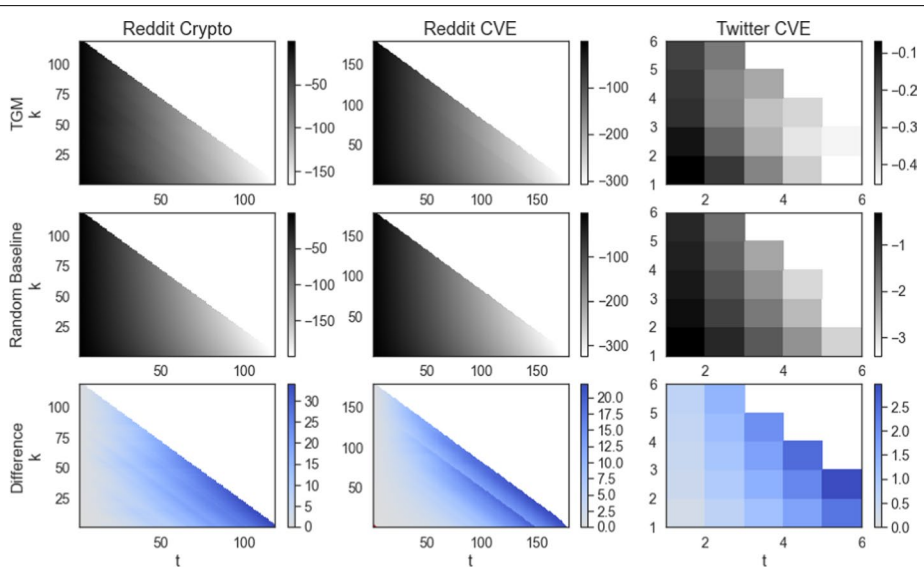

Fig. 10 Log-probability of placing a consecutive sequence of t nodes correctly, starting from an initial partial tree with k nodes, for each model and dataset. We also plot the difference in log-probabilities between TGM and the random baseline model, where blue regions show our model outperforming the baseline and red regions highlight the opposite case. Note that $t + k$ is equal to the tree size, so that tree size is conserved on the diagonals of these plots

Figure 10 shows that our model outperforms the baseline in consecutive node placement on all datasets and all values of t and k; the difference is very small for very small t (short prediction sequences). However, the small values of the log-probabilities indicate that the model is very rarely capable of correctly reproducing the microscopic evolution of the conversation trees, especially for long node placement sequences (large t). Performance is better for Twitter cascades, which are easier to predict because they tend to be very shallow. We believe that this poor performance is due to the accumulation of errors: even if we have a decent chance of selecting the next node correctly, the odds of doing so many times in a row drop geometrically.

Discussion

Our model outperforms the random baseline and is competitive with the state-of-the-art CTPM baseline in predicting macroscopic features such as depth, breadth, and structural virality of conversation trees. Nevertheless, the present evaluation reveals that our microscopic predictions deviate significantly from the empirical conversations. This negative result highlights the challenges of the microscopic prediction task for cascade trees: we are using a rich feature set and a strong model, and yet still failing.

In particular, our results highlight the effects of error accumulation: the probabilities of predicting long sequences of events are extremely small. The accumulation of errors could play an important role in limiting the microscopic predictability of many online diffusion networks (Goel et al. 2012). This problem should be investigated further in other predictive models of social systems.

How can our model do so poorly at microscopic predictions far into the future, and yet do reasonably well in predicting the structure of the final trees? This apparent contradiction reveals a lack of sensitivity of the traditional structural measurements. When a tree has many nodes, there are many distinct microscopic configurations leading to almost identical macroscopic structural properties such as size, depth, breadth, and structural virality. In other words, predicting these structural features is a lot easier than reproducing the evolving conversations.

Although the data provided by DARPA was gathered using public application programming interfaces, filtering public subreddits and public tweets, we cannot share it with the research community due to a non-disclosure agreement designed to protect the privacy of social media users. The lack of public datasets makes direct comparisons between methods difficult, hinders the reproduction of results, and slows down scientific progress in this area (Pasquetto et al. 2020). Unfortunately, it was not possible to evaluate TGM on publicly available datasets used in the evaluation of virality prediction models (Hogg and Lerman 2012; Leskovec et al. 2009; Hodas and Lerman 2014). We hope that in the future it will be possible to evaluate the proposed method on more standard datasets.

As in most studies of this kind, the present results are somewhat limited by the scope of our data, i.e., cryptocurrency and cyber-security. Yet, given the differences between these two domains, we expect that our model would perform similarly in other domains. Further experiments are necessary to confirm this conjecture. Regarding platforms, we detected imbalances between Reddit and Twitter in our results. Although our model is designed to be platform-agnostic, further analyses on other social media platforms (e.g.,

Facebook and Instagram) are needed to explore the universal applicability of the proposed model. In general, models like the ones presented in this paper are limited by the availability of training data, especially a sufficient number of large trees.

Finally, our framework involves extensive computational efforts to calculate the features, node-by-node and step-by-step. It does not scale well to simulate large cascades beyond tens of thousands of nodes. Future work should be devoted to optimizing the method to handle larger-scale cascades.

Conclusions

We proposed a generative model to grow individual conversation trees starting from minimal information—as little as just the initial post. Our approach combines two machine learning sub-models: a size predictor and a next-node placement model. We used three datasets from two social media platforms, Reddit and Twitter, spanning the topical domains of cryptocurrency and cyber-security.

A contribution of the proposed machine learning framework is the engineering of features useful for the size prediction and node placement tasks. We introduced the *parent delay* edge property to capture the characteristics of deep conversation threads versus shallow star-like broadcasts. This property encapsulate structural and (relative) temporal features of cascades, without requiring the model to estimate inter-arrival times. In addition, as an edge-level property, it is more informative about the growth dynamics than traditional tree-level properties such as depth and breadth.

Extensive validation results were presented to measure the accuracy of the two machine learning sub-models as well as the full aggregated generative model. To the best of our knowledge, this work presents the first evaluation of models aiming to reproduce the microscopic, step-by-step evolution of individual conversations trees.

Our results suggest that while it's possible to predict with some accuracy the global structural features of conversation cascades as well as microscopic dynamics in the short-term, predicting the long-term evolution of online conversations at the level of individual actions remains out of the reach of generative models.

Abbreviations

DARPA: Defense Advanced Research Projects Agency; CCDF: Complementary Cumulative Distributions Function; CVE: Common Vulnerabilities and Exposure; MRE: Mean Relative Error; CTPM: Cascade Tree Prediction Model; TGM: Tree Growth Model, our model.

Acknowledgements

We thank Rachel Krohn and Tim Weninger for providing access to their code for the CTPM baseline, and configuring it for our datasets. We are also grateful to Emilio Ferrara, Kristina Lerman, Jim Blythe, Goran Muric, and Alexey Tregubov for useful discussions.

Authors' contributions

All authors contributed to the conception and design of the study. JB, DP, and PMH developed the software implementing the machine learning models, processed and analyzed data, and generated figures. All authors interpreted results, contributed to the writing, and approved the final manuscript.

Funding

This work was supported by DARPA grant W911NF-17-C-0094.

Availability of data and materials

The data that support the findings of this study are available from DARPA but restrictions apply to the availability of these data, which were used under license for the current study, and so are not publicly available. Data are however available from the authors upon reasonable request and with permission of DARPA.

Competing interests

The authors declare that they have no competing interests.

Appendix

Extracting node placement classifier features

The node placement classifier's job is to take a node and a tree in a given state of its growth, and determine if that node will receive the next reply in the tree's growth process. So to train the classifier, we need to give it examples of nodes in the context of a tree at a particular stage in the tree's growth. To extract the training data for this classifier from the set of trees in our training set, we loop over each tree, each stage of the tree's growth, and every node within the tree at that stage of its growth (see Algorithm 1).

Algorithm 1 Node Placement Classifier Training Data Feature Extraction

```
output ← list()
for all tree in trainingTrees do
    if size(tree) > maxSize then
        tree ← trim(tree, maxSize)
    for t ←size(tree) to 1 do
        tree ← trim(tree, t)
        for all node in tree do
            features ← extractFeatures(tree, node)
            output.append(features)
return output
```

For a tree of size N, this results in $\sum_{n=1}^{N} n = N(N+1)/2$ data points. This $O(N^2)$ scaling naturally results in a very large training set for this classifier if we give it large trees. To manage this complexity, we set a maximum size of 150 nodes and trim all trees larger than that size down to the maximum size. Less than 1% of trees are trimmed, yet this choice greatly mitigates the computational complexity.

Feature importance

We computed feature importances using Mean Decrease Impurity (Louppe et al. 2013) for both the node classifier and the size regression on the Reddit datasets (Table 5) and Twitter dataset (Table 6).

In the size prediction task, we found that by far the most important features were the response time features (e.g., mean time between replies). These were strongly predictive of tree size for both the Reddit and Twitter datasets. On Twitter, the text content features were also significant, as was the time of day of the post; on Reddit, these features weren't very important. Furthermore, we found that the feature importance changes significantly with k. Notably, content features become less important with larger k, because there is more temporal information for the classifier to use and those signals are stronger.

In general, we found the the user features were unimportant in both tasks, especially for Reddit. Early tests indicated that, contrary to our expectation, including user features in the size prediction task led to worse results; they were essentially just a source of noise. Therefore in our final model we did not include them. However we suspect that in larger datasets, where the signal from user features is more reliable, these features could be important for size prediction. We did include user features in

Table 5 Feature importance in the node placement classifier and the size prediction regression model at $k = 2$ and $k = 5$, both for the Reddit Crypto dataset. Reddit CVE feature importance is similar

Feature	Node	Size (k=2)	Size (k=5)
Global			
Current size	0.71	–	–
Depth	–	0	0
Root edges	–	0	0
Recent edges	–	0	0
Parent delay (mean, median)	–	0, 0	0.002, 0
Node			
Node index	0.008	–	–
Node depth	0.015	–	–
Node parent delay	0.14	-	–
Subtree size	0.001	–	–
#Children	0.10	–	–
#Grandchildren	0.003	–	–
#Siblings	0.010	–	–
#Cousins	$< 10^{-3}$	–	–
#Uncles	$< 10^{-3}$	–	–
User (mean and median over conversations initiated by the user)			
Max breadth	$< 10^{-3}, < 10^{-3}$	–	–
Depth	$< 10^{-3}, 0$	–	–
1st Breadth	$< 10^{-3}, < 10^{-3}$	–	–
Lifetime	0, 0	–	–
Size	$< 10^{-3}, < 10^{-3}$	–	–
Root edges	$0.001, < 10^{-3}$	–	–
Recent edges	$< 10^{-3}, < 10^{-3}$	–	–
#Posts	0	–	–
Content (25 features, range given)			
Root text	$< 10^{-3}$	$< 10^{-3}$	$< 10^{-3}$
Temporal			
Response time (mean, median)	–	0.28, 0.21	0.84, 0.04
1st Response time	–	0.50	0.11
Root time of day	0	0	0
Root day of week	0	0	0

the node placement task because they did not seem to harm performance; but they did not significantly help either.

In the node placement task, the important features differed significantly between Reddit and Twitter. On Reddit, the strongest features were the current size of the whole tree, the number of children a node has, the parent delay of the node, and the node index. Of these, the number of children is significant because it tells the classifier how active the conversation stemming from this post/reply is, and the classifier has learned that more active conversations are more likely to receive replies. The other three features allow the classifier to pick the root node or the most recent node of the tree, which are common choices in conversation trees. On Twitter, the most important features were the node index and the node depth. This is likely because on

Table 6 Feature importance in the node placement classifier and the size prediction regression model at $k = 2$ and $k = 5$, both for the Twitter CVE dataset

Feature	Node	Size (k=2)	Size (k=5)
Global			
Current size	0.029	–	–
Depth	–	0.0	0.005
Root edges	–	0.0	0.11
Recent edges	–	0.0	0.0
Parent delay (mean, median)	–	0.0, 0.0	0.083, 0.082
Node			
Node index	0.54	–	–
Node depth	0.28	–	–
Node parent delay	0.013	–	–
Subtree size	0.021	–	–
#Children	0.043	–	–
#Grandchildren	0.008	–	–
#Siblings	0.012	–	–
#Cousins	0.002	–	–
#Uncles	0.002	–	–
User (mean and median over conversations initiated by the user)			
Max breadth	0, 0	–	–
Depth	0.003, 0.002	–	–
1st Breadth	$< 10^{-3}$, 0	–	–
Lifetime	0, 0	–	–
Size	$< 10^{-3}$, $< 10^{-3}$	–	–
Root edges	$< 10^{-3}$, $< 10^{-3}$	–	–
Recent edges	0.011, 0.011	–	–
#Posts	0	–	–
Content (25 features, range given))			
Root text	0–0.002	0–0.21	0–0.12
Temporal			
Response time (mean, median)	–	0.02, 0.006	0.028, 0.009
1st Response time	–	0.43	0.028
Root time of day	$< 10^{-3}$	0.012	0.009
Root day of week	0	0.0	0.0

Twitter most conversation trees are stars or near-stars, and these two features allow the classifier to choose the root node and its children. Subtree size, number of children, and parent delay were also significant, but less important than node index and depth.

Received: 22 September 2020 Accepted: 28 January 2021
Published online: 15 February 2021

References

Aragón P, Gómez V, García D, Kaltenbrunner A (2017a) Generative models of online discussion threads: state of the art and research challenges. J Internet Serv Appl 8(1):15. https://doi.org/10.1186/s13174-017-0066-z

Aragón P, Gómez V, Kaltenbrunner A (2017b) To thread or not to thread: the impact of conversation threading on online discussion. In: Proceedings of eleventh international AAAI conference on Web and social media, pp 12–21. https://www.aaai.org/ocs/index.php/ICWSM/ICWSM17/paper/viewPaper/15609

Backstrom L, Kleinberg J, Lee L, Danescu-Niculescu-Mizil C (2013) Characterizing and curating conversation threads. In: Proceedings of 6th ACM international conference on Web search and data mining (WSDM), pp 13–22. https://doi.org/10.1145/2433396.2433401. http://dl.acm.org/citation.cfm?doid=2433396.2433401

Beck J, Huang R, Lindner D, Guo T, Ce Z, Helbing D, Antulov-Fantulin N (2019) Sensing social media signals for cryptocurrency news. In: Companion proceedings of the 2019 World Wide Web conference, pp 1051–1054

Boyd D, Golder S, Lotan G (2010) Tweet, tweet, retweet: conversational aspects of retweeting on Twitter. In: Proceedings of 43rd Hawaii international conference on system sciences, pp 1–10. https://doi.org/10.1109/HICSS.2010.412. http://ieeexplore.ieee.org/document/5428313/

Cao Q, Shen H, Cen K, Ouyang W, Cheng X (2017) DeepHawkes: bridging the gap between prediction and understanding of information cascades. In: Proceedings of ACM international conference on information and knowledge management (CIKM)

Cheng J, Adamic LA, Dow PA, Kleinberg J, Leskovec J (2014) Can cascades be predicted? In: Proceedings 23rd international conference on World Wide Web, pp 925–936. https://doi.org/10.3390/ijms17101719. arXiv:1403.4608. https://doi.org/10.1145/2566486.2567997

Choi D, Han J, Chung T, Ahn Y-Y, Chun B-G, Kwon TT (2015) Characterizing conversation patterns in reddit. In: Proceedings of ACM conference on online social networks (COSN), pp 233–243. https://doi.org/10.1145/2817946.2817959

DARPA (2018) Computational simulation of online social behavior (SocialSim). https://www.darpa.mil/program/computational-simulation-of-online-social-behavior. Accessed 16 Jan 2021

Dow PA, Adamic L, Friggeri A (2013) The anatomy of large Facebook cascades. In: Proceedings of international AAAI conference on Web and social media. https://www.aaai.org/ocs/index.php/ICWSM/ICWSM13/paper/view/6123

Du N, Dai H, Trivedi R, Upadhyay U, Gomez-Rodriguez M, Song L (2016) Recurrent marked temporal point processes: Embedding event history to vector. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 1555–1564

Gao J, Shen H, Liu S, Cheng X (2016) Modeling and predicting retweeting dynamics via a mixture process. In: Proceedings 25th international conference companion on World Wide Web (WWW)

Goel S, Watts DJ, Goldstein DG (2012) The structure of online diffusion networks. In: Proceedings of 13th ACM conference on electronic commerce (EC), pp 623–638. https://doi.org/10.1145/2229012.2229058. http://dl.acm.org/citation.cfm?doid=2229012.2229058

Goel S, Anderson A, Hofman J, Watts DJ (2016) The structural virality of online diffusion. Manag Sci 62(1):180–196. https://doi.org/10.1287/mnsc.2015.2158

Gómez V, Kaltenbrunner A, López V (2008) Statistical analysis of the social network and discussion threads in slashdot. In: Proceedings of 17th international conference on World Wide Web (WWW), p 645. https://doi.org/10.1145/1367497.1367585. http://portal.acm.org/citation.cfm?doid=1367497.1367585

Gómez V, Kappen HJ, Litvak N, Kaltenbrunner A (2013) A likelihood-based framework for the analysis of discussion threads. World Wide Web 16(5–6):645–675. https://doi.org/10.1007/s11280-012-0162-8

Guo R, Shaabani E, Bhatnagar A, Shakarian P (2015) Toward order-of-magnitude cascade prediction. In: Proceedings of IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM)

Hodas NO, Lerman K (2014) The simple rules of social contagion. Sci Rep 4(1):4343

Hogg T, Lerman K (2012) Social dynamics of digg. EPJ Data Sci 1(1):5

Hui P-M, Weng L, Sahami Shirazi A, Ahn Y-Y, Menczer F (2018) Scalable detection of viral memes from diffusion patterns. In: Lehmann S, Ahn Y-Y (eds) Complex spreading phenomena in social systems: influence and contagion in real-world social networks. Computational social sciences, pp 197–211. Springer, Cham. https://doi.org/10.1007/978-3-319-77332-2_11

Islam MR, Muthiah S, Adhikari B, Prakash BA, Ramakrishnan N (2018) Deepdiffuse: predicting the 'who' and 'when' in cascades. In: Proceedings of IEEE international conference on data mining (ICDM), pp 1055–1060

Joulin A, Grave E, Bojanowski P, Mikolov T (2017) Bag of tricks for efficient text classification. In: Proceedings of the 15th conference of the European chapter of the association for computational linguistics, vol 2, pp 427–431

Kefato ZT, Sheikh N, Bahri L, Soliman A, Montresor A, Girdzijauskas S (2018) CAS2VEC: network-agnostic cascade prediction in online social networks. In: Proceedings of 5th international conference on social networks analysis, management and security (SNAMS), pp 72–79. https://doi.org/10.1109/SNAMS.2018.8554730. https://ieeexplore.ieee.org/document/8554730/

Klein J, Moeschberger M (2006) Survival analysis: techniques for censored and truncated data. Springer, Berlin

Kobayashi R, Lambiotte R (2016) Tideh: time-dependent Hawkes process for predicting retweet dynamics. In: Proceedings of tenth international AAAI conference on Web and social media (ICWSM)

Koenker R, Bassett G (1978) Regression quantiles. Econometrica 46(1):33–50

Krohn R, Weninger T (2019) Modelling online comment threads from their start. In: Proceedings of 2019 IEEE international conference on Big Data (Big Data), pp 820–829

Kumar R, Mahdian M, McGlohon M (2010) Dynamics of conversations. In: Proceedings of 16th ACM SIGKDD international conference on knowledge discovery and data mining (KDD), p 553. https://doi.org/10.1145/1835804.1835875

Leskovec J, Backstrom L, Kleinberg J (2009) Meme-tracking and the dynamics of the news cycle. In: Proceedings of 15th ACM SIGKDD international conference on knowledge discovery and data mining, pp 497–506

Li C, Ma J, Guo X, Mei Q (2017) DeepCas: an end-to-end predictor of information cascades. In: Proceedings of the 26th international conference on World Wide Web (WWW), pp 577–586. https://doi.org/10.1145/3038912.3052643

Louppe G, Wehenkel L, Sutera A, Geurts P (2013) Understanding variable importances in forests of randomized trees. In: NIPS'13: proceedings of the 26th international conference on neural information processing systems

Lumbreras A (2016) Automatic role detection in online forums. PhD thesis, Université de Lyon. https://tel.archives-ouvertes.fr/tel-01439342/

Lumbreras A, Jouve B, Velcin J, Guégan M (2017) Role detection in online forums based on growth models for trees. Soc Netw Anal Min 7(1):49. https://doi.org/10.1007/s13278-017-0472-z
Medvedev AN, Delvenne J-C, Lambiotte R (2019) Modelling structure and predicting dynamics of discussion threads in online boards. J Complex Netw 7(1):67–82
Mishra S, Rizoiu M-A, Xie L (2016) Feature driven and point process approaches for popularity prediction. In: Proceedings ACM international conference on information and knowledge management (CIKM)
Nishi R, Takaguchi T, Oka K, Maehara T, Toyoda M, ichi Kawarabayashi K, Masuda N (2016) Reply trees in Twitter: data analysis and branching process models. Soc Netw Anal Min 6(1):26. https://doi.org/10.1007/s13278-016-0334-0
Pacheco D (2019) twitter_cascades. https://github.com/diogofpacheco/twitter_cascades. Accessed 16 Jan 2021
Pasquetto IV, Swire-Thompson B et al (2020) Tackling misinformation: what researchers could do with social media data. HKS Misinf Rev. https://doi.org/10.37016/mr-2020-49
Pei S, Muchnik L, Andrade J, Zheng Z, Makse H (2014) Searching for superspreaders of information in real-world social media. Sci Rep 4:5547
Pinto H, Almeida J, Gonçalves M (2013) Using early view patterns to predict the popularity of youtube videos. In: Proceedings of ACM international conference on web search and data mining (WSDM)
Rizoiu M-A, Lee Y, Mishra S, Xie L (2017) A tutorial on Hawkes processes for events in social media. arXiv:1708.06401, arXiv
Rossi L, Magnani M (2012) Conversation practices and network structure in Twitter. In: Proceedings of international AAAI conference on Web and social media (ICWSM). https://www.aaai.org/ocs/index.php/ICWSM/ICWSM12/paper/view/4634
Salganik MJ, Dodds PS, Watts DJ (2006) Experimental study of inequality and unpredictability in an artificial cultural market. Science 311(5762):854–856. https://doi.org/10.1126/science.1121066
Shen H-W, Wang D, Song C, Barabási A-L (2014) Modeling and predicting popularity dynamics via reinforced poisson processes. In: Proceedings of 28th AAAI conference on artificial intelligence
Subbian K, Prakash BA, Adamic L (2017) Detecting large reshare cascades in social networks. In: Proceedings of 26th international conference on World Wide Web, pp 597–605. https://doi.org/10.1145/3038912.3052718
Wang C, Ye M, Huberman BA (2012) From user comments to on-line conversations. In: Proceedings of 18th ACM SIGKDD international conference on knowledge discovery and data mining (KDD), pp 244–252. https://doi.org/10.2139/ssrn.2012183. http://dl.acm.org/citation.cfm?doid=2339530.2339573
Wang J, Zheng VW, Liu Z, Chang KC-C (2017) Topological recurrent neural network for diffusion prediction. In: IEEE international conference on data mining (ICDM), pp 475–484
Watson HW, Galton F (1875) On the probability of the extinction of families. J Anthropol Inst G B Irel 4:138. https://doi.org/10.2307/2841222
Weng L, Menczer F, Ahn Y-Y (2013) Virality prediction and community structure in social networks. Sci Rep 3:2522. https://doi.org/10.1038/srep02522
Weng L, Menczer F, Ahn Y-Y (2014) Predicting successful memes using network and community structure. In: Proceedings of eighth international AAAI conference on weblogs and social media (ICWSM). http://www.aaai.org/ocs/index.php/ICWSM/ICWSM14/paper/view/8081
Weninger T (2014) An exploration of submissions and discussions in social news: mining collective intelligence of Reddit. Soc Netw Anal Min 4(1):1–19. https://doi.org/10.1007/s13278-014-0173-9
Zhao Q, Erdogdu MA, He HY, Rajaraman A, Leskovec J (2015) SEISMIC: a self-exciting point process model for predicting tweet popularity. In: Proceedings of 21th ACM SIGKDD international conference on knowledge discovery and data mining, pp1513–1522. https://doi.org/10.1145/2783258.2783401. arXiv:1506.02594

Publisher's Note