CrossCas: A Novel Cross-Platform Approach for Predicting Cascades in Online Social Networks with Hidden Markov Model

Jinwei Liu*, Xiaonan Zhang†, Richard A. Alo*, Xiuzhen Huang‡, Long Cheng§, Feng Deng¶
*Department of Computer and Information Sciences, Florida A&M University, Tallahassee, FL 32307, USA
†Department of Computer Science, Florida State University, Tallahassee, FL 32306, USA
‡Department of Computer Science, Arkansas State University, Jonesboro, AR 72401, USA
§School of Control and Computer Engineering, North China Electric Power University, Beijing, China
¶School of Automation, Beijing Information Science & Technology University, Beijing, China
*{jinwei.liu, richard.alo}@famu.edu, †xzhang@cs.fsu.edu, ‡xhuang@astate.edu,
§lcheng@ncepu.edu.cn, ¶dengfeng@bistu.edu.cn

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10001560

***Abstract*—Information sharing through online social networks (OSNs) facilitates quick discovery and consumption of information online. Many OSNs such as Facebook, Twitter provide resharing or reposting features, which allows users to share others' content with their own friends or followers. As content is shared from person to person, cascades of information-sharing can occur. There are many existing works focusing on analyzing and characterizing the cascades in OSNs. However, previous works focus on the analysis and characterization of cascades without providing a solution to accurately predict cascades. Although some methods for cascade prediction have been proposed recently, their methods work in social networks such as Facebook (or Twitter), and do not work well simultaneously in multiple OSNs such as Software Social Network (SSN) GitHub, Twitter and Reddit because GitHub, Twitter and Reddit have different social activity patterns. In this paper, we first perform a thorough analysis of cascades in multiple OSNs: GitHub, Twitter and Reddit, and identify the cascades of information-sharing. We then propose CrossCas, a novel cross-platform approach for predicting cascades in multiple OSNs with Hidden Markov Model (HMM). The experimental results show that our proposed method achieves high performance.**

***Index Terms*—cascades, information sharing, prediction, OSNs, cross-platform**

I. Introduction

With the popularity and wide accessibility of social networks, information sharing through social networks has become an essential part of modern life, and it enables people to quickly discover and consume information online. Not only users can generate their own content and share it with others, but also they can discover/consume the information generated by others (e.g., their social contacts) and reshare it to others (e.g., their own contacts). In some instances, an information can be reshared multiple times, and the resulting multiplicative mechanism can cause cascades over a huge amount of users, possibly even reaching regions of the social graph distant from the original post [1]. Most of the cascades do not spread far and beyond but are restricted in a small group of users [2], but few of them become very big and are referred to as viral cascades [3]. Cascade prediction aims to predict the process of information diffusion in the future based on observed cascades. Cascade prediction helps us uncover the basic mechanisms that govern collective human behavior in networks, and it is critical to decision-making on social networks such as viral marketing, online advertising, recommender systems and support for Internet of Things. However, it is not trivial to make predictions due to the myriad factors that influence a user's decision to reshare content.

Cascades tend to be bursty, and with a spike of activity occurring within a certain period of time of the content's introduction into the network [4], [5], which attracts many interests for many applications in various domains such as product sales prediction [6], [7], stock market prediction [8], [9] and disaster relief [10]. With the bursty nature of the cascades and the challenge of information overload in social media, the following interesting problems arise: (1) Do cascades occur in multiple OSNs with different activity patterns? (2) When do the cascades occur in those OSNs?

Can cascades be predicted? Many believe that cascades are inherently unpredictable, while a recent work [2] has developed a framework for addressing the cascade prediction problem and answered this question. Indeed, cascades of microblogs/Tweets [11]–[19], photos [2], videos [20] and academic papers [19], [21] have been proved to be predictable to some extent. Knowing "whether cascades occur in multiple OSNs" and "when cascades would occur in those OSNs" would be valuable, which can help us understand the longevity of content beyond its initial popularity, and points toward a more holistic view of how content spreads in a network.

Several works have studied the properties of cascades, such as size [22], growth [2], shape [2], and burst time [8]. Traditional methods cannot well handle the prediction of cascades simultaneously in multiple OSNs with different activity patterns such as GitHub, Twitter and Reddit. This is because the groups in different social media platforms have different activity patterns, and the groups in Twitter have generally shorter lifespans compared to those in GitHub and Reddit, which impacts the performance in predicting bursts of activity.

In this paper, we are going to answer two questions: 1) Do cascades occur in multiple OSNs with different activity patterns? 2) When would the cascades occur in those OSNs? We predict in which time window the cascades would occur. To handle the above problems, we propose a novel cross-platform approach for predicting cascades in multiple OSNs using HMM. We summarize the main contributions below.

- We provide a thorough analysis of cascades in three

OSNs: GitHub, Twitter and Reddit. The analysis results confirm our conjecture.
- We develop CrossCas, a novel cross-platform approach for predicting cascades in multiple OSNs using Hidden Markov Model.
- In addition to predicting whether cascades occur in multiple OSNs with different activity patterns, we also answer the question when the cascades would occur in those OSNs.
- Experiments have been carried out to show the advantages of CrossCas in predicting cascades in multiple OSNs with different activity patterns.

II. Related Work

With the rapid development of OSNs, cascade attracts more attention in computer science. The following reviews two major categories of the previous research on cascade.

Cascade Analysis and Characterization. Rodrigues *et al.* [23] analyzed the characteristics of the information cascades in Twitter. Yang *et al.* [24] proposed a time series clustering method to find the information diffusion patterns in Twitter. Caetano *et al.* [25] characterized attention cascades in WhatsApp groups from three distinct perspectives: structural, temporal and interaction patterns. Huang *et al.* [26] presented a comprehensive study on systematically characterizing socware cascades on Facebook. However, these works focus on discovering the rules and patterns of information cascades in social networks without providing a solution to accurately predict cascades.

Cascade Prediction. The methods for predicting size of information cascades can be categorized into two major approaches: feature-based methods and model-based methods. The feature-based methods compute a huge amount of potentially relevant features and use them in classification setting. Cheng *et al.* [2] developed a framework to predict the final size of information cascades based on content, behavioral and structural features. Cheng *et al.* [5] performed a large-scale analysis of cascades on Facebook over significantly longer time scales, and predicted recurrence of cascades and the relative size of cascades (i.e., resulting burst). Inspired by the recent success of deep learning in various complicated tasks, some studies [22], [27], [28] adopted deep learning methods to leverage various features for cascade prediction. However, the feature-based methods have some limitations such as laborious feature engineering, extensive training, scalability issues in terms of computing these features at scale and in an online manner [8]. Some model-based methods have also been proposed to predict cascades. Cui *et al.* [15] considered all nodes as features and presented a logistic model to measure the relative importance of nodes that have propagated before them. Zhao *et al.* [13] proposed SEISMIC to predict the final size of an information cascade spreading through a network. SEISMIC models the information cascades as self-exciting point processes on Galton-Watson trees. Yu *et al.* [29] proposed a novel NEtworked WEibull Regression model for modeling microbehavioral dynamics that significantly improved the interpretability and generalizability of traditional survival models. Xu *et al.* [30] proposed a deep learning architecture for cascade growth prediction, called CasGCN, which employs the graph convolutional network to extract structural features from a graphical input, followed by the application of the attention mechanism on both the extracted features and the temporal information before conducting cascade size prediction. Kong *et al.* [31] proposed a dual mixture self-exciting process, which leverages a Borel mixture model and a kernel mixture model, to jointly model the unfolding of a heterogeneous set of cascades. However, it is difficult for model-based methods to maintain the status of cascades across all nodes in a network when the number of nodes is in billions.

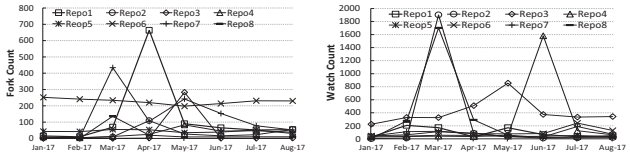
Our work differs from this literature as we consider multiple OSNs with different activity patterns and propose CrossCas, a novel cross-platform approach for predicting cascades in multiple OSNs with different activity patterns. CrossCas not only answers the question "Do cascades occur in multiple OSNs?" but also answers the question "When the cascades would occur in those OSNs?". CrossCas works well in multiple OSNs such as GitHub, Twitter and Reddit, which is more general. Finally, our model achieves high performance on the evaluation metrics (Accuracy, Precision, Recall and F1) and has low cost compared to the feature-based methods that require laborious feature engineering or extensive training.

III. Background and Objective

A. Social Platforms

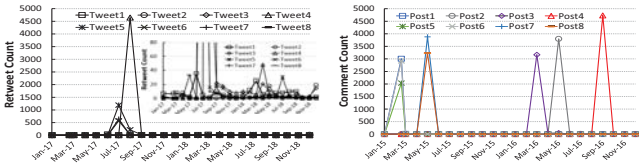
GitHub is an online collaborative software development platform that allows users to share and edit software repositories. The GitHub social network consists of two types of nodes: users and software repositories, and multiple types of links: events between a user and a repository. These link types include "Watch", "Fork", "Pull", "Push", "Issue", "Create", "Delete", "PullRequest", "IssueComment", "PullRequestReviewComment" and "CommitComment". A "Watch" event occurs when a user clicks the "Watch" button on GitHub to watch a repository. A "Fork" event occurs when a user makes a copy (or a branch) of the repository, etc. In this paper, we focus on the ForkEvent and WatchEvent in GitHub. The Reddit social network consists of two types of nodes: users and subreddit, and two types of links: events between a user and a subreddit. The two types are "Post" and "Comment". A "Post" event occurs when a user makes a submission (e.g., sharing a link, etc.). The Twitter social network consists of one node type users and four types of links (events): "Quoted Tweet", "Retweet", "Tweet" and "Reply". "Quoted Tweet" enables a user to say something along with his/her Retweet, while showing people the original tweet.

GitHub allows users to interact directly with each other by contributing to repos, and to interact indirectly by following other users or by watching specific repos. The ForkEvent and WatchEvent reflect information sharing. In this paper, we model the activity of Fork and Watch in GitHub, the activity of Retweet in Twitter and the activity of Comment in Reddit, and predict the cascades of ForkEvent and WatchEvent, the



(a) Cascades of Fork count. (b) Cascades of Watch count.

Fig. 1: Cascades of Fork and Watch in GitHub.



(a) Cascades of Retweet in Twitter. (b) Cascades of Comment in Reddit.

Fig. 2: Cascades of Retweet in Twitter and Comment in Reddit.

cascades of Retweet and the cascades of Comment in GitHub, Twitter and Reddit, respectively.

IV. Data Description and Analysis

In this section, we first describe our datasets, then we provide analysis of cascades in three OSNs: GitHub, Twitter and Reddit.

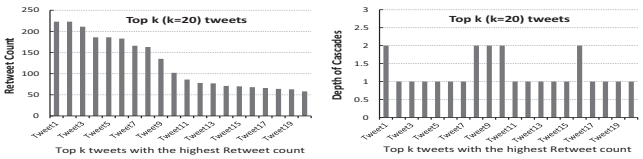
A. Dataset

Our datasets consist of the data in three OSNs: GitHub, Reddit and Twitter. The GitHub dataset consists of 8-month data (January to August, 2017). The Twitter data consists of two datasets. Twitter dataset1 consists of 24-month data (January, 2016 to December, 2017), and Twitter dataset2 consists of 24-month data (January, 2017 to December, 2018). The Reddit dataset consists of 24-month data (January, 2015 to December, 2016). The raw data consists of hourly activities on GitHub, Twitter and Reddit.

Figure 1(a) and Figure 1(b) show the cascades of Fork count and Watch count in GitHub data. In Figure 1(a), we see that the Fork count of some repos (e.g., Repo1, Repo7 and Repo3) increases dramatically in some months. In Figure 1(b), we see that the Watch count of some repos (e.g., Repo2, Repo8, Repo4 and Repo3) increases dramatically in some months. The results in Figure 1 confirm our conjecture that the cascade of content sharing (activity of content sharing) exists in software OSNs such as GitHub.

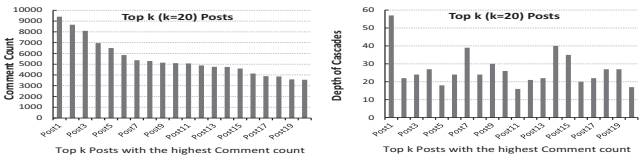
Figure 2(a) and Figure 2(b) show the cascades of Retweet (count) in Twitter (dataset2) and the cascades of Comment (count) in Reddit data, respectively. We see Retweet count of the tweets and Comment count of posts increase dramatically in some months. Similarly, the results in Figures 2(a) and 2(b) confirm our conjecture that the cascade of content sharing exists in OSNs such as Twitter and Reddit. By examining Figures 1 and 2, we find that the groups in different social media platforms have different activity patterns, and the groups in Twitter have generally shorter lifespans compared to those in GitHub and Reddit.

Figure 3(a) shows Retweet count of top k ($k = 20$) Tweets in Twitter (dataset1). We see that every Tweet has over 50 Retweet count, and the highest Retweet count is over 200.



(a) Retweet count of top k (k=20) tweets. (b) Depth of cascades of top k (k=20) tweets.

Fig. 3: Retweet count and depth of cascades of top k (k=20) tweets with the highest Retweet count in Twitter.



(a) Comment count of top k (k=20) Posts. (b) Depth of cascades of top k (k=20) Posts.

Fig. 4: Comment count and depth of cascades of top k (k=20) Posts with the highest Comment count in Reddit.

Figure 3(b) shows the depth of cascades of Retweet count of those top k ($k = 20$) tweets in Twitter data.

Figure 4(a) shows Comment count of top k ($k = 20$) Posts in Reddit. We see that every Post has over 4,000 Comment count, and the highest Comment count is over 9,000. Figure 4(b) shows the depth of cascades of Comment count of those top k ($k = 20$) Posts in Reddit data. Comparing Figure 3 and Figure 4, we find that Comment in Reddit has larger count of content sharing activity and larger depth of cascades.

V. Predicting Cascades in OSNs using HMM

In this paper, we use the Hidden Markov Model to predict the activity occurrence of cascades. Below we illustrate the process of predicting the activity (e.g., Fork, Retweet, etc.) occurrence of cascades.

Given the historical data, the information of occurrence of cascades can be obtained. Denote max_c, m_c and min_c as the maximum count, average count and minimum count of content resharing (Suppose the count of content resharing is measured based on a time unit, e.g., 1 month), respectively. CrossCas splits the interval $[min_c, max_c]$ into 3 subintervals: $[min_c, min_c + \frac{1}{5}(m_c - min_c)]$, $(min_c + \frac{1}{5}(m_c - min_c), m_c + \frac{1}{5}(max_c - m_c))$, $[m_c + \frac{1}{5}(max_c - m_c), max_c]$, and defines these three parts as valley, center and peak, respectively, to categorize the observation symbols of the HMM model. The corresponding hidden states determined by the observation symbols are under-growth (UG), normal-growth (NG), over-growth (OG), respectively (see Figure 5).

Denote $S = \{S_1, ..., S_H\}$ $(H = 3)$ as the set of states, q_t as the state at t, and $Q = q_1 q_2 ... q_T$ as a state sequence. Let $V = \{1, ..., M\}$ $(M = 3)$ be the set of possible observation symbols per state, and $O = \{O_1, ..., O_T\}$ $(O_i \in V,\ \forall i = 1, ..., T)$ be the observation sequence, where M is the number of observation symbols ($1, 2, 3$ represent "peak", "center" and "valley" regions, respectively) and T is the length of observation sequence. To determine the observation symbols, CrossCas considers the time interval between two consecutive observation time slots j and $j + 1$ $(j = 1, ..., T - 1)$ as a

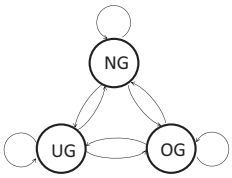


Fig. 5: Hidden Markov Model with three states: over-growth (OG), normal-growth (NG), and under-growth (UG).

window, and divides the window into $L-1$ subwindows. Let Δ_j be the difference between the maximum count of content resharing and the minimum count of content resharing in the window. If Δ_j falls in $[min_c, min_c + \frac{1}{5}(m_c - min_c)]$, then CrossCas considers the observation symbol at $j+1$ is valley; if Δ_j falls in $(min_c + \frac{1}{5}(m_c - min_c), m_c + \frac{1}{5}(max_c - m_c))$, then it considers the observation symbol at $j+1$ is center; otherwise, the observation symbol at $j+1$ is peak. Hence, the state transition probability matrix can be obtained as follows:

$$A = \{a_{ij}\}\ (a_{ij} = P\{q_{t+1} = S_j | q_t = S_i\},\ 1 \le i, j \le H), \quad (1)$$

where the state transition coefficients satisfy: $a_{ij} \ge 0$ and $\sum_{j=1}^{H} a_{ij} = 1$. The observation probability matrix $B = \{b_j(k)\}$ can be obtained as follows:

$$B = \{b_j(k)\}\ (b_j(k) = P\{O_t = k | q_t = S_j\}, 1 \le j \le H, 1 \le k \le M), \quad (2)$$

where $b_j(k)$ is the probability that the observation symbol is k given the sate at t is S_j. Equ. (2) records the observation symbol probability distribution in state S_j. Hence, we get the initial state distribution

$$\pi = \{\pi_i\}\ (\pi_i = P\{q_1 = S_i\},\ 1 \le i \le H). \quad (3)$$

Given the model $\lambda = (A, B, \pi)$ and an observation sequence O, our goal is to find the most likely state sequence. Specifically, we aim to maximize the expected number of correct states for the HMM. We define $\gamma_t(i)$ as the probability of being in state S_i at time t, given the observation sequence O and the model λ:

$$\gamma_t(i) = P\{q_t = S_i | O, \lambda\}. \quad (4)$$

By using the forward-backward variables, we can simplify Equ. (4) as follows:

$$\gamma_t(i) = \alpha_t(i)\beta_t(i)/P(O|\lambda), \quad (5)$$

where $\alpha_t(i)$ is the forward variable defined as

$$\alpha_t(i) = P(O_1 O_2 \cdots O_t, q_t = S_i | \lambda), \quad (6)$$

where $\beta_t(i)$ is the backward variable defined as

$$\beta_t(i) = P(O_{t+1} O_{t+2} \cdots O_T | q_t = S_i, \lambda). \quad (7)$$

Based on [32], $\alpha_t(i)$ and $\beta_t(i)$ can be solved inductively.

By using $\gamma_t(i)$, we can solve for the individually most likely state q_t at time t, as

$$q_t = argmax_{1 \le i \le H}[\gamma_t(i)],\ 1 \le t \le T. \quad (8)$$

Equ. (8) chooses the most likely state for each t to maximize the expected number of correct states. In implementation, we use Viterbi algorithm to find the single best state sequence (path), denoted by Q^*=$q_1^* ... q_L^*$, i.e., maximizing $P(Q, O|\lambda)$ which is equivalent to maximizing $P(Q|O, \lambda)$ [32], and we use the method in [33] to re-estimate the parameters A, B, π.

Based on the work [34], we can estimate the probability distribution of the next cascade observation as follows:

$$E_{P_{T+1}(k)} = \sum_{j=1}^{H} P(q_{T+1} = S_j | q_T = q_L^*) \cdot b_j(k)\ (k \in \{1, ..., M\}). \quad (9)$$

We consider the observation symbol which has the highest value of $E_{P_{T+1}(k)}$ as the observation symbol of the next time $T+1$, that is, $k|_{E_{P_{T+1}(k)} = max_{u=1}^{M}(E_{P_{T+1}(u)})}$.

VI. Performance Evaluation

In this section, we first describe the metrics used for evaluation, then present how to setup the experiments, and finally present our findings and analyses.

A. Performance Metrics

To evaluate the performance of CrossCas, we primarily focus on the evaluation metrics Accuracy, Precision, Recall and F1.

- **Precision:** the fraction of the number of time windows (with predicted cascades) in which the cascades indeed occur, computed as $\frac{TP}{TP+FP}$, where TP represents True Positive, and FP represents False Positive.
- **Recall:** the fraction of all cascades that were correctly identified by the system, computed as $\frac{TP}{TP+FN}$, where FN represents False Negative.
- **F1:** the geometric mean of Precision (P) and Recall (R) measures, computed as $\frac{2PR}{P+R}$.
- **Accuracy:** the overall fraction of instances classified correctly into the proper class, computed as $\frac{TP+TN}{TP+TN+FP+FN}$, where TN represents True Negative.

B. Experiment Setup

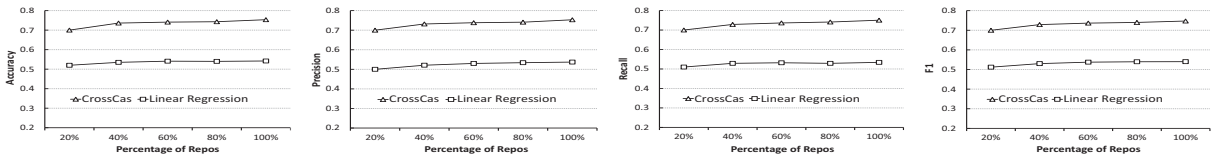
1) Data Collection: Our dataset consists of Github, Reddit, and Twitter activities. The GitHub dataset considered for the development of CrossCas for predicting cascades, consists of 6-month training data (January to June, 2017) and two-month testing data (July to August, 2017). The Twitter data consists of two datasets. The Twitter dataset1 consists of 18-month training data (January, 2016 to June, 2017) and 6-month testing data (July to December, 2017). The Twitter dataset2 consists of 18-month training data (January, 2017 to June, 2018) and 6-month testing data (July to December, 2018). The raw data consists of hourly activities on GitHub, Reddit and Twitter. In the data preprocessing phase, necessary information in the form of user-id, repository-id in GitHub, Subreddit-id in Reddit, Hashtag in Twitter and event-type with timestamps were extracted from these activities.

2) Method for Comparison: We compared our algorithm with Linear Regression.

Linear Regression. Linear regression is a method of data evaluation and modeling that establishes linear relationships between variables that are dependent and independent. This method would thus model relationships between dependent variables and independent variables from the analysis and learning to the current training results. Linear regression is commonly used in mathematical research methods, where it is possible to measure the predicted effects and model them against multiple input variables.

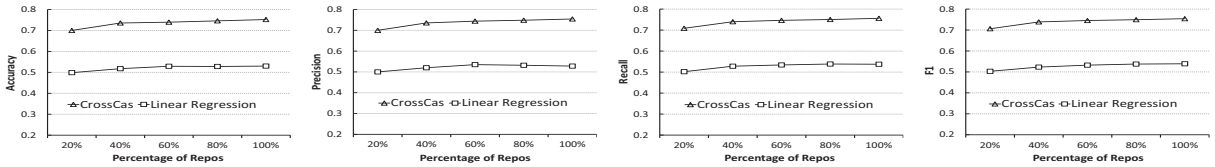
C. Findings and Analyses

Figure 6(a) shows the Accuracy of cascade prediction of ForkEvent in GitHub across different methods. In Figure 6(a), we see that the Accuracy of CrossCas is higher than that of



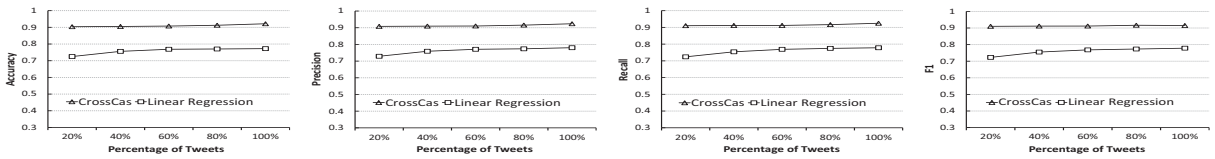
(a) Accuracy (b) Precision (c) Recall (d) F1s

Fig. 6: Performance of various evaluation metrics for cascade prediction of ForkEvent across different methods.



(a) Accuracy (b) Precision (c) Recall (d) F1s

Fig. 7: Performance of various evaluation metrics for cascade prediction of WatchEvent across different methods.



(a) Accuracy (b) Precision (c) Recall (d) F1s

Fig. 8: Performance of various evaluation metrics for cascade prediction of Retweet across different methods based on Twitter dataset1.

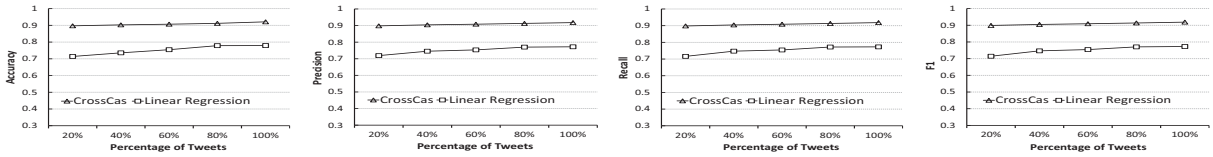
Linear Regression, which indicates that CrossCas has better performance on Accuracy than Linear Regression. The Accuracy slightly increases as the percentage of repos increases, and the overall prediction Accuracy is around 0.735. Figure 6(b) shows the Precision of cascade prediction of ForkEvent in GitHub across different methods. In Figure 6(b), we see that the Precision of CrossCas is higher than that of Linear Regression, which indicates that CrossCas has better performance on Precision than Linear Regression. The value of Precision slightly increases as the percentage of repos increases, and the overall Precision is around 0.733. Figure 6(c) shows the Recall of cascade prediction of ForkEvent in GitHub across different methods. In Figure 6(c), we see that the Recall of CrossCas is higher than that of Linear Regression, which indicates that CrossCas has better performance on Recall than Linear Regression. The value of Recall slightly increases as the percentage of repos increases, and the overall Recall is around 0.731. Figure 6(d) shows the F1 score of cascade prediction of ForkEvent in GitHub across different methods. In Figure 6(d), we see that the F1 score of CrossCas is higher than that of Linear Regression, which indicates that CrossCas has better performance on F1 than Linear Regression. The F1 score slightly increases as the percentage of repos increases, and the overall F1 score is around 0.730. The possible reasons behind this include: Linear Regression assumes a linear relationship between the input and output variables, and it fails to fit complex datasets properly; Linear Regression is sensitive to outliers; the use of feature vectors in HMM model makes the transition probability sensitive to any word in the input sequence.

Figure 7 shows the Accuracy, Precision, Recall and F1 score of WatchEvent in GitHub across different methods. The overall Accuracy, Precision, Recall and F1 score are around 0.735, 0.736, 0.74 and 0.739, respectively. The results in Figure 7 are similar to that in Figure 6 due to the same reasons.

Figure 8(a) shows the prediction Accuracy of Retweet cascades in Twitter across different methods based on Twitter dataset1. In Figure 8(a), we see that the prediction Accuracy of CrossCas is higher than that of Linear Regression, which indicates that CrossCas has better performance on Accuracy than Linear Regression. The overall prediction Accuracy is around 0.910. Figure 8(b) shows the Precision of cascade prediction of Retweet in Twitter across different methods based on Twitter dataset1. In Figure 8(b), we see that the Precision of CrossCas is higher than that of Linear Regression, which indicates that CrossCas has better performance on Precision than Linear Regression. The overall Precision is around 0.913. Figure 8(c) shows the Recall of cascade prediction of Retweet in Twitter across different methods based on Twitter dataset1. In Figure 8(c), we see that the Recall of CrossCas is higher than that of Linear Regression, which indicates that CrossCas has better performance on Recall than Linear Regression. The overall Recall is around 0.914. Figure 8(d) shows the F1 score of cascade prediction of Retweet in Twitter across different methods based on Twitter dataset1. In Figure 8(d), we see that the F1 score of CrossCas is higher than that of Linear Regression, which indicates that CrossCas has better performance on F1 than Linear Regression. The overall F1 score is around 0.913.

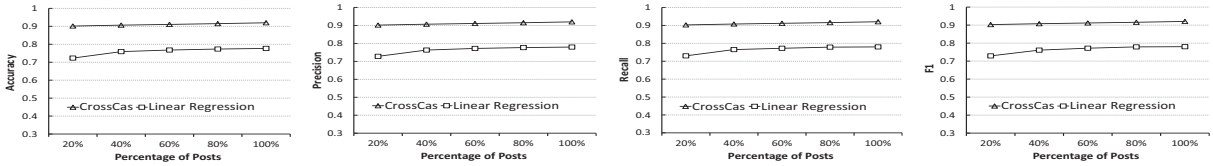
Figure 9 shows the Accuracy, Precision, Recall and F1 score of Retweet cascade prediction across different methods based on Twitter dataset2. The overall Accuracy, Precision, Recall and F1 score are around 0.908, 0.908, 0.908 and 0.909, respectively. The results in Figure 9 are similar to that in Figure 8.

Figure 10 shows the Accuracy, Precision, Recall and F1 score of Comment cascade prediction in Reddit across different methods. The overall Accuracy, Precision, Recall and F1 score are around 0.911, 0.911, 0.912 and 0.912, respectively. The results in Figure 10 are similar to that in



(a) Accuracy (b) Precision (c) Recall (d) F1s

Fig. 9: Performance of various evaluation metrics for cascade prediction of Retweet across different methods based on Twitter dataset2.



(a) Accuracy (b) Precision (c) Recall (d) F1s

Fig. 10: Performance of various evaluation metrics for cascade prediction of Comment across different methods.

Figures 8, 9, 6 and 7.

VII. Conclusion

This paper presents a novel cross-platform approach for predicting cascades in multiple OSNs with different activity patterns. In this paper, we first perform a thorough large-scale analysis of cascades in three OSNs: GitHub, Twitter and Reddit, and identify the cascades of information-sharing. We then propose CrossCas for predicting cascades in multiple OSNs using Hidden Markov Model. The experimental results show that CrossCas achieves high performance on Accuracy, Precision, Recall and F1 compared to the existing approach. In our future work, we will compare CrossCas with the state-of-the-art (e.g., deep learning algorithm) to fully verify the performance of CrossCas. We will consider handling the gap between the training period and testing period (which typically exists in practical scenarios). We will also consider the intensity of cascades in OSNs, and the effects of rockstar influence and network structure on cascades in OSNs.

Acknowledgment

This research was supported by Faculty Research Awards Program at Florida A&M University.

References

[1] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proc. of ACM WWW*, 2010.
[2] J. Cheng, L. Adamic, P. A. Dow, J. M. Kleinberg, and J. Leskovec. Can cascades be predicted? In *Proc. of WWW*, 2014.
[3] K. Subbian, B. Prakash, and L. Adamic. Detecting large reshare cascades in social networks. In *Proc. of ACM WWW*, Perth, 2017.
[4] S. Myers and J. Leskovec. The bursty dynamics of the twitter information network. In *Proc. of ACM WWW*, 2014.
[5] J. Cheng, L. Adamic, J. Kleinberg, and J. Leskovec. Do cascades recur? In *Proc. of ACM WWW*, pages 671–681, 2016.
[6] D. Gruhl, R. Guha, R. Kumar, J. Novak, and A. Tomkins. The predictive power of online chatter. In *Proc. of ACM SIGKDD*, Chicago, 2005.
[7] Y. Chen, H. Amiri, Z. Li, and T. Chua. Emerging topic detection for organizations from microblogs. In *Proc. of SIGIR*, pages 43–52, 2013.
[8] S. Wang, Z. Yan, X. Hu, P. S. Yu, and Z. Li. Burst time prediction in cascades. In *Proc. of AAAI*, Austin, 2015.
[9] D. Sornette. Predictability of catastrophic events: Material rupture, earthquakes, turbulence, financial crashes, and human birth. In *Proc. of Nat. Acad. Sci.*, volume 99, pages 2522–2529, 2002.
[10] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: Real-time event detection by social sensors. In *Proc. of WWW*, 2010.
[11] L. Yu, P. Cui, F. Wang, C. Song, and S. Yang. Uncovering and predicting information cascading process with behavioral dynamics. In *Proc. of ICDM*, 2015.
[12] L. Weng, F. Menczer, and Y. Ahn. Predicting successful memes using network and community structure. In *Proc. of ICWSM*, 2014.
[13] Q. Zhao, M. A. Erdogdu, H. Y. He, A. Rajaraman, and J. Leskovec. Seismic: A self-exciting point process model for predicting tweet popularity. In *Proc. of KDD*, 2015.
[14] M. Jenders, G. Kasneci, and F. Naumann. Analyzing and predicting viral tweets. In *Proc. of ACM WWW*, 2013.
[15] P. Cui, S. Jin, L. Yu, F. Wang, W. Zhu, and S. Yang. Cascading outbreak prediction in networks: a data-driven approach. In *SIGKDD*, 2013.
[16] A. Guille and H. Hacid. A predictive model for the temporal dynamics of information diffusion in online social networks. In *WWW*, 2012.
[17] B. Wang, C. Wang, J. Bu, C. Chen, W. V. Zhang, D. Cai, and X. He. Whom to mention: expand the diffusion of tweets by @ recommendation on micro-blogging systems. In *Proc. of WWW*, pages 1331–1340, 2013.
[18] X. Tang, D. Liao, W. Huang, J. Xu, L. Zhu, and M. Shen. Fully exploiting cascade graphs for real-time forwarding prediction. In *Proc. of AAAI*, 2021.
[19] X. Xu, F. Zhou, K. Zhang, S. Liu, and G. Trajcevski. Casflow: Exploring hierarchical structures and propagation uncertainty for cascade prediction. *IEEE TKDE*, 2021. doi:10.1109/TKDE.2021.3126475.
[20] C. Bauckhage, F. Hadiji, and K. Kersting. How viral are viral videos. In *Proc. of AAAI Conference on Web and Social Media (ICWSM)*, 2015.
[21] H. Shen, D. Wang, C. Song, and A. Barabási. Modeling and predicting popularity dynamics via reinforced poisson processes. In *AAAI*, 2014.
[22] C. Li, J. Ma, X. Guo, and Q. Me. Deepcas: an end-to-end predictor of information cascades. In *Proc. of WWW*, Perth, 2017.
[23] T. Rodrigues, F. Benevenuto, M. Cha, K. Gummadi, and V. Almeida. On word-of-mouth based discovery of the web. In *Proc. of IMC*, 2011.
[24] J. Yang and J. Leskovec. Patterns of temporal variation in online media. In *Proc. of WSDM*, 2011.
[25] J. Caetano, G. Magno, M. Gonçalves, J. Almeida, H. Marques-Neto, and V. Almeida. Characterizing attention cascades in whatsapp groups. In *Proc. of WebSci*, 2019.
[26] T. Huang, M. Rahman, H. Madhyastha, M. Faloutsos, and B. Ribeiroy. An analysis of socware cascades in online social networks. In *Proc. of WWW*, 2013.
[27] W. Zhang, W. Wang, J. Wang, and H. Zha. User-guided hierarchical attention network for multi-modal social image popularity prediction. In *Proc. of ACM WWW*, 2018.
[28] F. Ducci, M. Kraus, and S. Feuerriegel. Cascade-lstm: A tree-structured neural classifier for detecting misinformation cascades. In *KDD*, 2020.
[29] L. Yu, P. Cui, F. Wang, C. Song, and S. Yang. Uncovering and predicting the dynamic process of information cascades with survival model. *Knowl Inf syst*, 50(2):633–659, 2017.
[30] Z. Xu, M. Qian, X. Huang, and J. Meng. CasGCN: Predicting future cascade growth based on information diffusion graph. *arXiv preprint arXiv:2009.05152*, 2020.
[31] Q. Kong, M. Rizoiu, and L. Xie. Describing and predicting online items with reshare cascades via dual mixture self-exciting processes. In *Proc. of CIKM*, 2020.
[32] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proc. of IEEE*, 77(2):257–286, 1989.
[33] M. Stamp. A revealing introduction to hidden markov models. January 18, 2004, http://www.cs.sjsu.edu/faculty/stamp/RUA/HMM.pdf.
[34] W. Gao and G. Cao. Fine-grained mobility characterization: Steady and transient state behaviors. In *Proc.of MOBIHOC*, 2010.