IEEE Workshop on Parallel and Distributed Processing for Computational Social Systems (ParSocial 2017)
Proc. IEEE International Parallel and Distributed Processing Symposium Workshops
June 2 2017, Orlando, Florida, pp. 1447-1456 DOI 10.1109/IPDPSW.2017.82

Predicting Viral News Events in Online Media

Xiaoyan Lu
Computer Science Department
Rensselaer Polytechnic Institute
Troy, NY, USA
Email: lux5@rpi.edu

Boleslaw Szymanski
Computer Science Department
Rensselaer Polytechnic Institute
Troy, NY, USA
Email: szymab@rpi.edu

***Abstract*—The information diffusion and dissemination define critical dynamics observed in large complex networks. The underlying information propagation topology, however, is often hidden or incomplete because of the lack of explicit citations of the sources. We proposed a scalable parallel algorithm to derive the node embeddings to better understand the information dissemination patterns and predict emergent cascades of viral events in online media. Unlike previous works which concentrate on modeling the links of information propagation, our algorithm infers the topic-specific output influence and the input selectivity of nodes. The parallel algorithm iteratively merges local node embeddings in particular communities to obtain the global optimal results so that the processing of cascades can be significantly accelerated. Based on the obtained latent representation of nodes, the emergent cascades of viral news events in online media can be successfully predicted with an 80% accuracy at its early stage. Experimental results show that our parallel inference algorithm achieves a 50-fold speedup and requires a low communication overhead, while the accuracy of the cascade size prediction is preserved.**

***Keywords*-cascade; information propagation; network; viral news events; prediction; online media**

I. Introduction

With the rise of online social networking sites such as Facebook and Twitter, information propagation in these platforms attracts the interests of many researchers. Revealing the patterns of contents sharing and dissemination in such networks is a critical task because it not only provides insight into human social behaviors but also finds its application in predicting emergent social, economical and political phenomenons.

The mutual excitations between nodes are widely observed in a variety of networks, ranging from the diffusion of purchasing behavior among friends to the spread of contagious disease. In online media, the spreading of news events can also be treated as the cascade among online news sites, which exhibits an emergent pattern - some breaking news events get massively reported around the globe within a few hours. Understanding the information propagation structure among the news sites is critical to the prediction of such viral news events. However, the underlying information propagation topology is often hidden or incomplete because of the lack of explicit references to the sources making certain relationships missing in the observed records. Recently there have been many works [1]–[5] on propagation network reconstruction, yet the focuses are on inferring the edges, which requires analysis of a huge number of suspected links among all the pairs of nodes. This is because, given the observed cascades in which n nodes are involved, $O(n^2)$ potential edges need to be taken into consideration to locate the real propagation links.

In this paper, we model the nodes directly. Every node in the propagation network has two latent vectors, one representing its influence to other nodes on different topics and the other representing its selectivity upon the inputs from other nodes on different subjects [6]. Such latent representations of the influence and selectivity are analogous to the node's vector representation in the non-negative matrix factorization approach [7]–[9], where the inner product of two vectors indicates the strength of connection between the corresponding nodes. In this way, the total number of variables is linear to the number of nodes, and, more importantly, it permits a fast stochastic gradient descent algorithm searching for the optimal node embeddings.

As shown in [10], strong community structures in a network enhance the local, intra-community spreading. This phenomenon is supported by the observations in online media: most news events are reported by the news sites in the same region, using the same language. We exploit such community structures to parallelize the derivation of node embeddings because most cascades occurs in local communities. According to their frequency of co-occurence in cascades, the nodes in a network are divided into multiple communities in which the propagation occurs frequently. A process is assigned to every single community and searches for the local optimal influence and selectivity vectors for the nodes in it. These local vector representations are merged together in a hierarchical manner to obtain the optimal global results. Since the communities do not have any intersection, the Write-Write conflicts can be completely avoided during the gradient descent process, so these processes can execute in parallel with small communication overhead.

In the context of epidemic models, the propagation of information can be considered as infections spreading from a node to its neighbor. This is the simplest SI epidemics because the adoption of information happens only once, i.e.

a node can not recover from the infection, which indicates that the spread delay depends only on the earliest infection spread speed. This feature has been highlighted in the stochastic propagation model proposed by Kempe et al. [11], which was shown to be related to survival analysis [2]. In this paper, we follow the continuous time infection framework proposed in [2] which models the information propagation processes are modeled as stochastic infection process. Unlike [2], we represent the latent parameters of one link by the influence and selectivity of its two endpoints, which reduces the number of latent parameters and permits an efficient parallel algorithm to infer them. These node embeddings are then used as input to predict the size of cascades.

Experimental results show that our algorithm can obtain accurate influence and selectivity vectors, which serve as the indicators to predict the virality of cascades. The parallelization of the inference algorithm achieves a decent performance and low communication overhead. The block-coordinate stochastic gradient descent algorithm converges very fast in practice to maximize the log-likelihood. We also demonstrate the applications of our approach in identification of the significant influencers and the prediction of emergent cascades in the GDELT dataset, which includes thousands of news sites and news events in tens of languages. The derived node influence and selectivity representations can serve as a tool to predict the viral news events reported in global scale.

In Section II, we describe the GDELT dataset and interesting features of the spread of news events contained in the dataset. The related work on stochastic propagation model and survival analysis are presented in Section III-A, followed by the explanation of our extension and the fast inference algorithm in Section. Section IV describes the inference algorithm in detail and Section V presents our approach to prediction inference. The experimental results are presented in Section VI and we conclude our work in Section VII.

II. News Events Data

The Global Database of Events, Language, and Tone (GDELT)[1] project extracts news events from tens of thousands of news sites around the world. Advances in natural language processing allows the sentiment analysis, real-time translation of 65 languages and the identification in events data of named entities including organization, location, count and theme. The dataset is currently available on Google Cloud Platform. Users can use Google BigQuery to download specific tables or process the data remotely by SQL commands. The GDELT database contains records the mentions of news events by news sites since February 19, 2015 to present. Here we present some unique features explored in these records.

[1]http://www.gdeltproject.org/

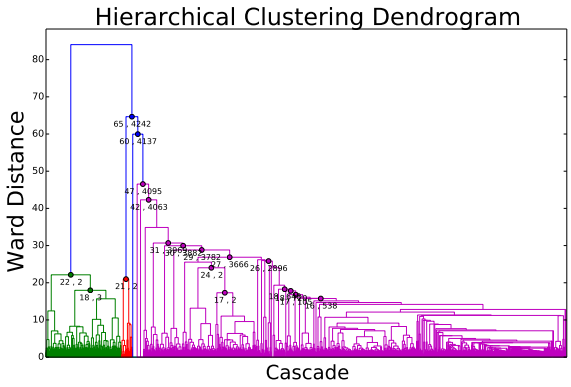

Figure 1. Dendrogram of hierarchical clustering of the 5,000 random cascades of news events.

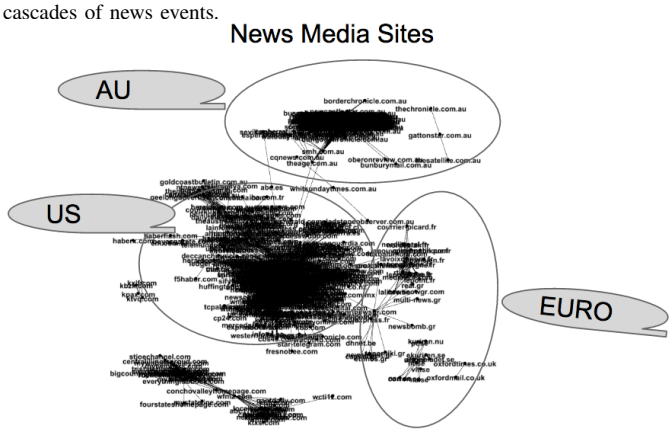

Figure 2. Backbone network of the news sites associated with the 5,000 sampled cascades of news events.

- **Emergence of news events**. We compute the duration of events, computed as the period since its earliest record until the latest. Most news events are reported by the news site within the first 50 hours. This reflects the short life cycle of the production and consumption of the news events. One explanation for this phenomenon is that a news site would prefer not to report an event which is considered out-of-date. In contract to the breaking news, the discussion of long-term topics such as greenhouse effect lasts a very long time, but these events constitute a very small portion of the dataset.
- **Most cascades are local**. For a period covering one year, we randomly choose 5,000 news events in the GDELT dataset. Any two news sites reporting at least 50 events together are linked in the graph shown in Figure 2. Clearly, there are four clusters visible in Figure 1, one standing for the news sites in the U.S., one for the news sites in Australia and the other for the news sites in European countries, while the remaining one is a mixture of sites in different regions. Initially, the pair-wise distance between any two cascades of

events is measured by the Jaccard-index

$$\text{Jaccard} = \frac{|N(i) \cap N(j)|}{|N(i) \cup N(j)|} \tag{1}$$

where $N(j)$ is the set of news sites reporting the news event j. Then the hierarchical clustering algorithm [12], which merges iteratively the closest cascades according to the Ward distance measure among all pairs of cascades, is applied to obtain a dendrogram as shown in Figure 1. The Ward distance and the number of cascaded contained in a cluster are also plotted in text at some associated inner nodes of the clustering tree. There are three obvious clusters shown in the dendrogram. The largest cluster in blue corresponds to the news sites in the U.S., while most news sites in Australia are located in the middle cluster in green. The news sites in U.K. and European countries are placed in the left-most cluster in red. Although the plot shows only a raw distribution of the news site in different clusters, the dual visualization indicates that the hierarchical structures of cascades correlate with the community structure of co-reporting networks of the news sites. That is, as an illustrative example here, most cascades of news events occur frequently within certain regions only.

- **Matthew effect in online media**. The term Matthew effect (or accumulated advantage) originated in sociology and refers to a phenomenon in which "the rich get richer and the poor get poorer." In network science, it refers to preferential attachment of earlier nodes in a network [13] where nodes that few nodes that acquire more connections than others will increase its connectivity at a higher rate while others have few connections. As seen in Fig. 3, the distribution of the number events reported per site follows the Power Law, which has been widely observed in a variety of networks. As shown in Figure 3, only a few popular news sites have reported millions of events, while most of the news site reported 5,000-10,000 events. Here the news site which reported less than 5,000 events in one year are ignored because they are less influential than the remaining ones, which causes the straight cut-off in the Figure 3.

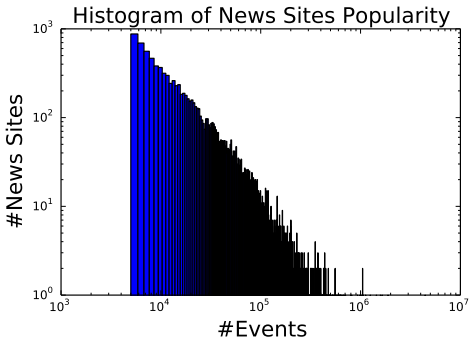

Figure 3. The number of events reported by the news sites. The news sites reporting at least 5,000 events are considered.

III. Approach

A. Preliminary

In our work, we adapt the stochastic propagation model proposed by Kempe et al. [11] to model the information dissemination in graphs. One unique feature of the model is that a node can not adapt the same message twice, i.e. any node can be infected at most once. Hence, every infected node has a unique infection time for every message. In the stochastic propagation model, a message diffuses from a node to its neighbor with a random delay. The distributions of such infection delays are independent from each other for different links, governed by the parameters of the corresponding links separately.

According to the definition of the stochastic propagation model, the propagation of messages can be naturally considered as the stochastic processes which are characterized by the link parameters defining infection delays. Formally, a realization of such stochastic process is defined as a cascade, which refers to a sequence of infection events.

Definition 1 A cascade is a sequence of distinct infections (v_i, t_{v_i}) for $i = 1, 2, \ldots, s$, where an infection is a tuple (v_i, t_{v_i}) indicating the node v_i gets infected at time t_{v_i}.

Any node infected earlier than v is considered a potential source of node v's infection. The stochastic propagation model permits only one single source for each infection – if v is infected by u at time t_v, then the other predecessors of v must not infect v at a time earlier than t_v. Given the observed cascade, the probability [2] that node u infects node v is

$$\mathcal{P}[u \to v] = h_{uv}(t_v - t_u) \prod_{l \prec v} S_{lv}(t_v - t_l) \tag{2}$$

where $l \prec v$ if and only if l is infected earlier than v in the cascade, i.e. $t_l < t_v$, $S_{lv}(\Delta t)$ denotes the probability that node v was not infected by node l within the period Δt, and $h_{uv}(\Delta t)$ denotes the probability that node u infects node v with a delay of Δt conditioned on the fact that no previous infection occurs before Δt. Hence, the likelihood of generating a particular cascade is

$$L = \prod_v \sum_{u \prec v} \mathcal{P}[u \to v] \tag{3}$$

$$= \prod_v \sum_{u \prec v} \Big[h_{uv}(t_v - t_u) \prod_l S_{lv}(t_v - t_l) \Big] \tag{4}$$

$$= \prod_v \prod_l S_{lv}(t_v - t_l) \sum_{u \prec v} h_{uv}(t_v - t_u) \tag{5}$$

It is worth noting that in survival analysis [14] $h()$ and $S()$ are the hazard function and survival function respectively. And the survival function, according to its definition, can be derived from the harzard function. A common choice of the hazard function is the exponentially decaying, which simplifies the derivation of the corresponding likelihood.

B. Embedding representation of nodes

Provided with the set of edges and the associated harzard function of each edge, the likelihood generating a particular cascade is defined by (3). Once the harzard and survival functions are parameterized, the model can then be recovered by maximizing its likelihood given the observed cascades. However, the major concern here is that, given a network of n nodes, $O(n^2)$ candidate links have to be taken into consideration. Rather than model the propagation links, our framework models the nodes directly. It finds suitable embedding vector representations of nodes in the network so that such vectors reflects the influence and selectivity of nodes. The primary advantage is that the number of latent variables becomes linear in the number of nodes. More importantly, it permits the derivation of a linear-time inferring algorithm as presented in Section IV.

For every single node u, $A_{u,k}$ measures its influence and $B_{u,k}$ measures its selectivity on the k-th topic. In the news events data, $A_{u,k}$ indicates the probability that other news sites reports the same event after the news site u's coverage of this event, and $B_{u,k}$ indicates how likely the input is accepted by news site u. A high value of $A_{u,k}$ and a high value of $B_{v,k}$ implies that the information disseminated from node u is likely to be accepted by node v. The values of $A_{u,k}$ and $B_{u,k}$ are not necessarily correlated.

For the k-th topic, the propagation from a node u to a node v depends on the influence of u and selectivity of v. If the infection delay on each topic is assumed to follow exponential distribution with a rate parameter $A_{u,k}B_{v,k}$, then minimum infection delay across the K topics also follows the exponential distribution with a rate parameter $\sum_k A_{u,k}B_{v,k}$. Hence, the hazard function and survival function from u to v have the function forms,

$$h_{uv}(\Delta t) = \sum_{k=1}^{K} A_{u,k}B_{v,k} = A_u B_v^T \tag{6}$$

$$S_{uv}(\Delta t) = \prod_{k=1}^{K} e^{-A_{u,k}B_{v,k}\Delta t} = e^{-A_u B_v^T \Delta t} \tag{7}$$

These function forms are the special cases of the Alan's additive regression model [15]. The corresponding log-likelihood has the form,

$$\mathcal{L}_c(A,B) = \sum_{v \in c} \Big[\sum_{l \prec_c v} (t_l - t_v) A_l B_v^T + \ln \sum_{u \prec_c v} A_u B_v^T \Big] \tag{8}$$

where $u \prec_c v$ if nodes u and v are in cascade c, i.e. $u \in c$, $v \in c$, and $t_u < t_v$. Using the MLE estimator, the optimization problem becomes,

$$\max \quad \sum_{c \in C} \mathcal{L}_c(A,B) \tag{9}$$

$$\text{subject to} \quad A_{u,k} \geq 0 \qquad \forall u,k \tag{10}$$

$$B_{v,k} \geq 0 \qquad \forall v,k \tag{11}$$

The constraints are necessary because the hazard function $h_{u,v}(t)$ is non-negative for $t \geq 0$.

IV. Inference algorithm

We proposed a fast inference algorithm to obtain the influence and selectivity vectors of the nodes using projected gradient descent [16].

A. Projected gradient descent

For a particular cascade c and some node v involved in c, the first order derivative of $\mathcal{L}_c(A,B)$ over B_v is

$$\nabla_{B_v} \mathcal{L}_c(A,B) = \sum_{l \prec_c v} (t_l - t_v) A_l + \frac{\sum_{u \prec_c v} A_u}{\sum_{u \prec_c v} A_u B_v^T} \tag{12}$$

The project gradient descent [16] method updates the vector B_v along the direction of the derivative of the log-likelihood. When the components of B_v becomes negative, it forces the value to be zero. The numerator of second term $\sum_{u \prec_c v} A_u$ represents the total influence of all the potential source nodes causing v's infection. Adding the second term to B_v would always increase the inner product between these two vectors. Yet, adding the term $(t_l - t_v)A_l$ would decrease the inner product between A_l and B_v because $t_l < t_v$. In this regard, eventually, the nodes with long infection delay $(t_v - t_l)$ to node v would have a relatively small impact on it because $A_l B_v^T$ would be small.

Equation (12) can be rewritten as

$$\nabla_{B_v} \mathcal{L}_c(A,B) = G(v) - t_v H(v) + \frac{H(v)}{H(v) B_v^T} \tag{13}$$

$$H(v) = \sum_{l \prec_c v} A_l \tag{14}$$

$$G(v) = \sum_{l \prec_c v} (t_l A_l) \tag{15}$$

It requires one sweep over the infected nodes to compute $H(v)$ and $G(v)$ for any v in the cascade c. Then, another sweep is sufficient to compute the values of $\nabla_{B_v} \mathcal{L}_c(A,B)$ for $\forall v \in c$. The time complexity here is linear in the number of infections in the cascade.

Similarly, it costs linear time to compute the derivative of $\mathcal{L}_c(A,B)$ over A_u for any node u involved in cascade c, according to the equation:

$$\nabla_{A_u} L(A,B) = P(u) t_u - Q(u) + \sum_{v : u \prec_c v} \frac{B_v}{H(v) B_v^T} \tag{16}$$

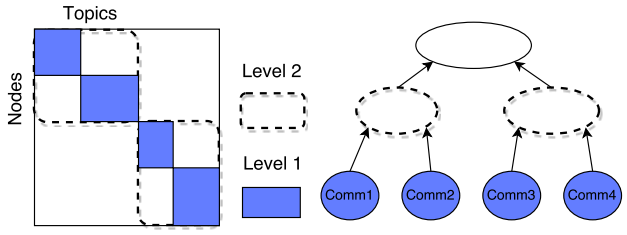

Figure 4. Illustration of the community-based parallel stochastic gradient descent algorithm. On the right is the hierarchical clustering tree where every level illustrates a valid partition of the network. The parent community is derived by joining its two child communities. The algorithm searches for the local optimal embedding of nodes for all communities at the same level in parallel and use such embedding as the initial values for the upper level. During parallel computation at a level, every processor needs the access to a separate block of the influence or selectivity matrix on the left, which avoid Write-Write conflicts. The algorithm iteratively traverses from the bottom level to the top level until its termination at the root of the tree.

where $P(u) = \sum_{v:u \prec_c v} B_v$ and $Q(u) = \sum_{v:u \prec_c v} B_v t_v$.

B. Community-based parallelization

As shown in [10], strong communities facilitate global diffusion by enhancing local, intra-community spreading. This phenomenon is also validated by the observations on news event datasets: most events are massively reported by the news sites from the same region. As communities structure are widely observed in networks, we can exploit such communities structures to parallelize the derivation of node embeddings. The first step is to divide the network topology into multiple dense sub-modules in which the propagation occurs frequently. In this way, the inference algorithm can search for local optimal node embeddings in each individual sub-module in parallel and then merge these local embeddings iteratively to obtain the global optimal results.

A frequent co-occurence graph is constructed by counting the number of appearances of a pair of nodes in the observed cascades. For two nodes u and v, the weight of the directed edge (u, v) is defined as

$$w(u, v) = \frac{2c(u, v)}{c(u) + c(v)}$$

where $c(u)$ is the number of cascades including node u and $c(u, v)$ is the number of appearances of both nodes u and v when node u becomes infected before node v does in some cascade. Such weight is in the range $[0, 1]$.

As nodes in the same community are likely to appear in the same cascade frequently, we run the community detection algorithm, SLPA [17], in this frequent co-occurence graph to detect the dense sub-modules. According to the resulting partition, a process is created for every single community and take the infections of the nodes inside it as the input to update their influence and selectivity vectors. The pseudo code of the parallel algorithm is shown in Algorithm 1. At the beginning, each cascade is divided into multiple sub-cascades according to the node memberships (the lines 1-7). In this way, the inference algorithm can work separately on each individual community, using the cascades among nodes in each community as input (the lines 10-19). The inference algorithm for a specific local community terminates when the corresponding log-likelihood no longer increases or the max number of iterations is exceeded.

Algorithm 1 Parallel Inference Algorithm (One Level)

```
1:  for each node v do
2:      r[v] ← the community of node v
3:  end for
4:  m ← the total number of communities
5:  for each cascade c do
6:      create m empty sub-cascade c^1, c^2, ..., c^m
7:      for each infection (v, t) in c do
8:          r ← r[v]
9:          add (v, t) to sub-cascade c^r
10:     end for
11: end for
12: for all community r do in parallel
13:     repeat
14:         dA ← 0
15:         dB ← 0
16:         for each cascade c^r do
17:             for each node u in c^r do
18:                 dA[u] = dA[u] + α * ∇_{A_u} L_{c^r}(A, B)
19:                 dB[u] = dB[u] + α * ∇_{B_u} L_{c^r}(A, B)
20:             end for
21:         end for
22:         for each node u in community r do
23:             Update A[u] using dA[u]
24:             Update B[u] using dB[u]
25:         end for
26:     until Early stopping or the max number of iterations
        is exceeded
27: end for
```

In Algorithm 1, the Write-Write conflicts can be avoided because each process writes to the distinct non-intersecting rows in matrices A and B which are associated with two set of nodes in the different communities. Hence, the communication overhead is reduced to a minimum.

The spreading across different communities can actually play an important role in the global spreading. Hence, we propose a hierarchical approach which merges small communities into large ones. The idea is to distribute the heavy workload to the inference process in small communities which can be parallelized well. As illustrated in Figure 4, a hierarchical clustering tree is constructed to guide the parallel algorithm. The communities obtained by SLPA algorithm serve as the leaves of the clustering tree. For a particular level with k communities, k processes work on

the separate communities in parallel. A barrier is created for these k processes. When they have finished, the algorithm goes to the upper level where two communities from the previous level are merged into one community. This process results in $k/2$ communities and consequently $k/2$ processes are required. The derived influence and selectivity vectors in the previous level then becomes the initial values for the upper level. Hence, the processes can start from these initial values to continue searching for the optimal vectors in the upper level.

For example, Figure 4 shows four non-overlapping communities at the level 1. Four processes are allocated to these communities separately, each searching for the local optimal embedding of the nodes in one community. Then, every two communities at level 1 are merged into one community at level 2, leading to a total of 2 communities. So, two processes are required at level 2. Finally, at the root of the tree, one unique process sequentially updates the influence and selectivity vectors of all the nodes in the network. From the aspect of matrix operations, this process can be interpreted as follows: initially, four processes update the four different non-overlapping parts of the matrix in parallel. The rows of each part correspond to the nodes in an individual community. Then, in the next phrase, every process updates two such parts and two processes are needed. Finally, one unique process updates the entire matrix.

The pseudo code of the hierarchical approach is presented in Algorithm 2. The hierarchical approach terminates when the number of remaining communities is smaller than a threshold.

Algorithm 2 Parallel Inference Algorithm (Hierarchical)

```
L[1] ← communities detected by SLPA algorithm.
i ← 1
repeat
    Call Algorithm 1 with partition L[i]
    L[i + 1] ← join every two communities in L[i]
until the number of communities in L[i] ≤ q
```

The distribution of the amounts of work among different processors depends on the modularity of the propagation graph. If a graph consists of many dense sub-graphs, the algorithm can handle these sub-graphs in parallel, but if we can not find such community structures, the efficiency will be reduced. For example, in a core-periphery graph [18] with an unique dense core structure, the community detection algorithm may output a large community, representing the core, along with many small ones. The processors handling small communities might wait for the processor handling the large community to finish. In Algorithm 2, a binary tree is used for load balancing. The binary tree is balanced as the numbers of tree nodes contained in any two branches are approximately equal. In order to increase the efficiency, it is recommended to balance the tree by the number of graph nodes contained in two different branches rather than the number of tree nodes. We leave this improvement as the future work and remain the simple hierarchical design here. In [19], the authors proposed a lock-free design for parallelizing stochastic gradient descent so that a nearly optimal rate of convergence can be achieved with theoretical guarantees. We plan to provide similar theoretical results for our hierarchical design in the future.

V. Predicting Viral Phenomenon

The approaches to predicting viral information propagation can be generally divided into two categories: One includes the feature-based regression or classification models [20], [21] which predict the size and duration of a cascade by its features at the initial stage. The other includes stochastic process approaches which simulates the progress of information dissemination as point process [22]. The first category of approaches extracts important features of the cascades, such as the number of early adopters, uninfected neighbors of early adopters and the communities where such early adopters are located. The second category, on the other hand, treats the growth of infections as self-excited counting process, thus the network topology is not needed for the prediction as in the previous case. Since network topology is not available in many applications, the first approach may suffer from the lack of accurate network topology information. For example, in the news event dataset, the records of events are provided but the associations between the news sites in different regions are hidden.

Inferring the hidden propagation paths is critical for the success of feature-based predictions. In [1]–[3], [23], authors proposed algorithms to infer the structure of the information diffusion network in social networks and online media. In our work, instead of using the edges of the networks, we use the influence and selectivity vectors of the early adopters directly as the features for predicting the final cascade size. The framework is illustrated in Figure 5. The first k cascades are used to infer the influence and selectivity vectors of the nodes. When a new cascade occurs, the influence and selectivity of the early adopters are then used as the input of feature-based prediction model to predict the results. Specifically, the following features are extracted:

- The divergence of the influence vectors of the early adopters

$$\text{diverA} = \max_{i,j:t_i<t_j} \|A_i - A_j\| \tag{17}$$

  which is the maximum euclidean distance between a pair of influence vectors. Since the nodes who are influential in a certain topic may not necessarily be influential in another topic, a high divergence of the influence vectors indicates that the cascade is likely to spread in different topics.

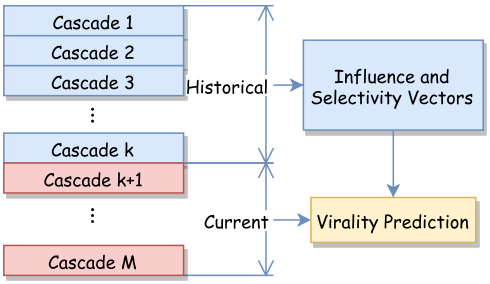

Figure 5. The prediction framework via the influecne and selectivity of the early adopters.

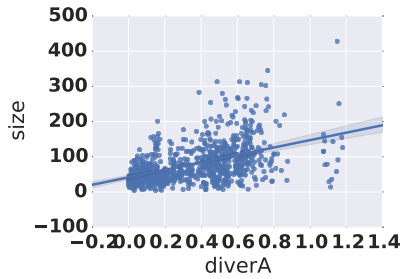

Figure 6. Extracted feature *diverA* of the early adopters and the final cascade sizes in the SBM graphs.

- The euclidean norm of the summation of the influence vector of early adopters

$$\text{normA} = \|\sum_{i \in c} A_i\| \tag{18}$$

- The maximum component of the summation of the influence vectors of early adopters

$$\text{maxA} = \max_k (\sum_{i \in c} A_i)_k \tag{19}$$

where x_k represents the k-th component of vector x.

Using these features, a SVM [24] model with linear kernel is used to predict the final cascades size. We use a simple classifier because it can demonstrate that these features are representative for predicting the final size of cascade.

VI. Experiments

We evaluate the performance of the parallel inference algorithm and measure the accuracy of cascade prediction using the derived influence and selectivity vectors. Different number of processes are used to evaluate the performance of the parallelization.

A. Synthetic Networks

Synthetic networks are generated using Stochastic Blockmodels (SBM) [25]. Given the membership of all the n nodes, the edge with two endpoints in different communities is created with the probability β and the edge inside a community is created with probability α. In most cases, α is much larger than β, so the graph contains the dense intra-community structures and sparse inter-communities structures. We create the SBM graphs containing 2,000 nodes with $\alpha = 0.2$ and $\beta = 0.001$. The average degree of nodes is approximately 10. Each community in SBM graph contains approximately 40 nodes.

On each network instance, the spreading process is simulated according to the stochastic propagation model proposed by Kempe et al. [11]. Since any cascade would eventually flood the entire network, we set an observation window during for the cascades. After the observation window, the current spreading process will be terminated instantly and a random node is chosen as the initiator to start the simulation of the next cascade. A total of 3,000 cascades are collected for each graph instance. The first 2,000 cascades are used to infer the influence and selectivity vectors of nodes in the network and the last 1,000 cascades are used to test the accuracy of the virality prediction algorithm. For the last 1,000 cascades, the infections occurring in the first 2/7 time of the observation window are provided for the prediction task, while the remaining parts of infections are assumed to be unknown.

The virality prediction problem is formulated as a binary classification problem. According to the number of infections involved, the cascades are divided into two different classes: the cascades with final size smaller than a threshold and the others with final size larger than that threshold. As mentioned in Section V, the features of the early adopters are extracted and fed to a linear SVM classifier. The performance of the prediction is evaluated by the F1-measure [26] using a 10-fold cross validation. Generally speaking, a high threshold makes the prediction problem challenging because the samples in two classes are unbalanced.

The extracted features of early adopters including *diverA*, *normA* and *maxA* are shown in Figures 6, 7 and 8. The extracted features serve as critical indicators for the viral cascades. The size of the cascade grows almost linearly as these features increase. The divergence of the influence vectors of the early adopters is an obvious signal to distinguish the potential viral cascades. In Figure 6, almost all the cascades whose sizes are larger than 2,00 have the feature diverA larger than 3.1. The extracted features including normA and maxA also serve as critical signals to distinguish the potential viral cascades.

The accuracy of the classification is illustrated in Figures 9. In the histogram, each bin contains the cascades of the sizes in a particular range. We use different number of nodes as the threshold for the binary classification and plot the F1-measure in the red curve. The accuracy of predicting the top 20% cascades is around 80%.

B. GDelt News Events

From the GDelt dataset, we choose a total of 6,000 most popular news sites around the world including yahoo.com,

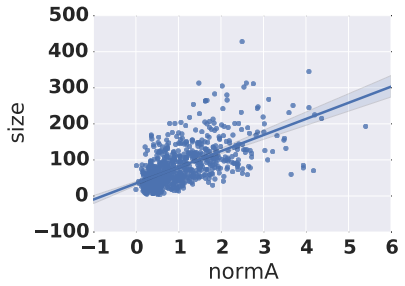

Figure 7. Extracted feature *normA* of the early adopters and the final cascade sizes in the SBM graphs.

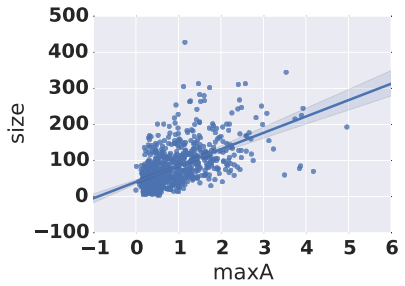

Figure 8. Extracted feature *maxA* of the early adopters and the final cascade sizes in the SBM graphs.

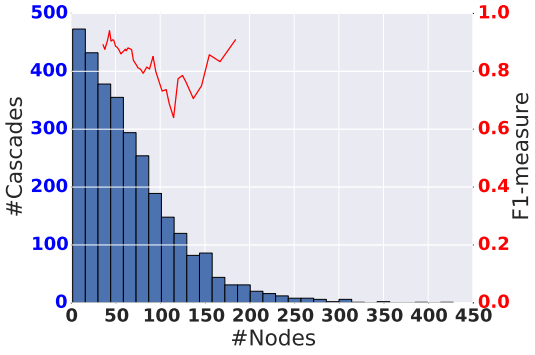

Figure 9. Accuracy of the popular cascade prediction in SBM graphs.

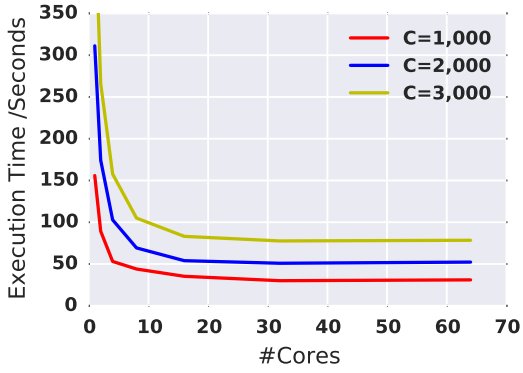

Figure 10. The time costs of processing cascades on SBM graph with 2,000 nodes. C denotes the number of cascades.

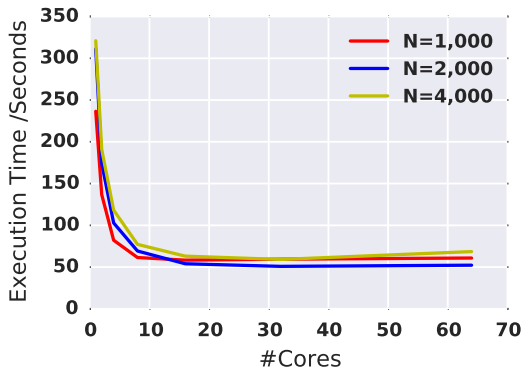

Figure 11. The time costs of processing 2,000 cascades on the SBM graphs of different sizes. N denotes the number of nodes.

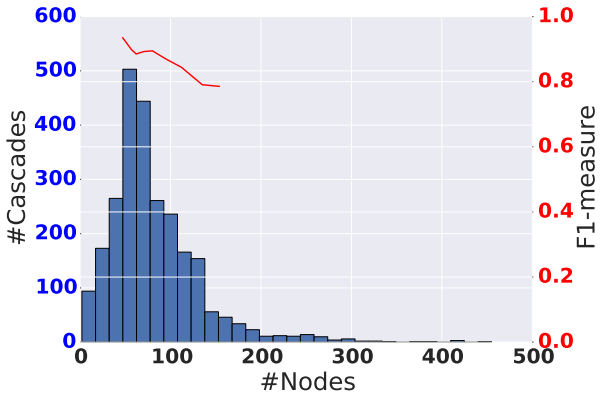

Figure 12. Accuracy of the popular news event prediction in GDELT dataset.

bbc.com and nytimes.com etc.. The popularity of a news site is measured by the number of news events it reported from year 2012 to 2014. A total of 2,600 news events are randomly sampled from the top one million most reported news events in the world. For each news event, the news sites reporting the event in the first 5 hours are used to the predict the total number of reports in 3 days.

As shown in Figure 12, using the extracted features of the news sites including *diverA*, *normA* and *maxA*, the prediction accuracy is approximately 80%, which generally matches the performance of predictions made on SBM graphs.

C. Speedup

The time costs of processing different numbers of cascades on SBM graph are shown in Figure 10. In our experiment, we evaluate the performance of the parallelization using 1, 2, 4, 8, 16, 32, 64 cores. One process can execute on a particular core at the same time. For a fair comparison, the inference algorithm and community detection algorithm SLPA use the same parameters in all the cases.

As seen in Figure 10, the time cost of processing cascades have been reduced significantly when the number of cores

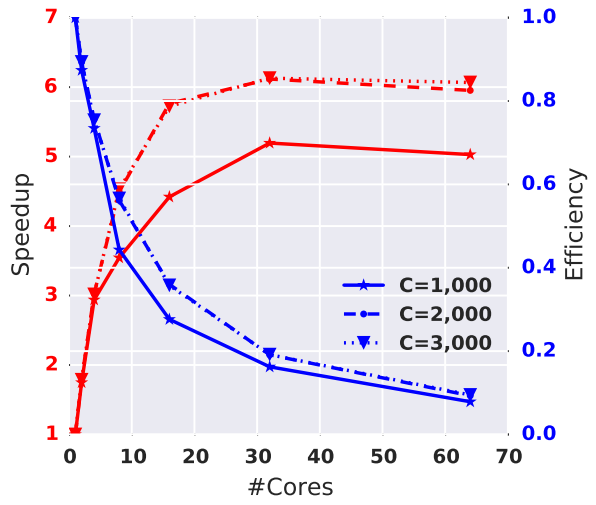

Figure 13. Speedup and efficiency of the parallel inference algorithm. C denotes the number of cascades.

increases. On the same SBM graph with 2,000 nodes, we evaluate the time needed to process 1,000, 2,000 and 3,000 cascades. The time cost is generally linear in the number of cascades. It demonstrates the good scalability of our approach. As the number of cores increases, the communication overhead also increases, thus, the speedup is not very high from 32 cores to 64 cores. However, comparing to the sequential algorithm, the achieved speedup is still impressive. We also measure the time costs of processing 2,000 cascades on the SBM graphs of different sizes. In Figure 11, the time costs needed to process the graphs of different sizes are similar. The difference in time costs of processing different graphs are approximately 10-20 seconds. As the inference algorithm takes the cascades as input, the time cost does not increase significantly even if more nodes are involved.

We evaluate the speedup and efficiency of the parallel inference algorithm processing cascades of different sizes. The speedup measures the improvement in speed of execution using multiple processors, which is given as

$$s_n = \frac{t_1}{t_n} \tag{20}$$

where t_n denotes the execution time using n processors. The efficiency is a metric of the utilization of the resources which is defined as

$$e_n = \frac{s_n}{n} \tag{21}$$

The SBM graph includes 2,000 nodes in our experiments and the sizes of cascades considered are 1,000, 2,000 and 3,000. Figure 13 shows that the inference algorithm scales well to 8-16 processors in general. The algorithm achieves the best speedup with 32 cores.

VII. Conclusion

We explore the news event dataset, GDELT, to understand the structure of the online media and the emergent propagation patterns of the news events among online news sites around the world. A novel framework is proposed to infer the influence and selectivity of the online media sites on a variety of topics. An efficient parallel inference algorithm is proposed. Hence, the framework can be applied to the large dataset GDELT which includes thousands of news sites and millions of news events. In addition, we also use machining learning models to predict the number of news sites reporting one specific event. By extracting many features of a news event at its early stage, the model achieves a decent accuracy on predicting the final virality of the news event. This approach also reveals the importance of the extracted features on the eventual popularity of the news events, which would shed light into the rapid growth of the reports on some critical news events in online media.

Acknowledgment

This work was supported in part by the Army Research Laboratory (ARL) under Cooperative Agreement Number W911NF-09-2-0053, (NS CTA), and by the Army Research Office (ARO), grant W911NF-16-1-0524. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies either expressed or implied of the Army Research Laboratory, the Army Research Office or the U.S. Government.

References

[1] Z. Shen, W.-X. Wang, Y. Fan, Z. Di, and Y.-C. Lai, "Reconstructing propagation networks with natural diversity and identifying hidden sources," *arXiv preprint arXiv:1407.4451*, 2014.

[2] M. Gomez Rodriguez, J. Leskovec, and B. Schölkopf, "Structure and dynamics of information pathways in online media," in *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*. ACM, 2013, pp. 23–32.

[3] A. Braunstein and A. Ingrosso, "Network reconstruction from infection cascades," *arXiv preprint arXiv:1609.00432*, 2016.

[4] X. Han, Z. Shen, W.-X. Wang, Y.-C. Lai, and C. Grebogi, "Reconstructing direct and indirect interactions in networked public goods game," *Scientific Reports*, vol. 6, 2016.

[5] H. Liao and A. Zeng, "Reconstructing propagation networks with temporal similarity," *Scientific Reports*, vol. 5, 2015.

[6] P. Bogdanov, M. Busch, J. Moehlis, A. K. Singh, and B. K. Szymanski, "Modeling individual topic-specific behavior and influence backbone networks in social media," *Social Network Analysis and Mining*, vol. 4, no. 1, pp. 1–16, 2014.

[7] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems*, 2001, pp. 556–562.

[8] I. Psorakis, S. Roberts, M. Ebden, and B. Sheldon, "Overlapping community detection using bayesian non-negative matrix factorization," *Physical Review E*, vol. 83, no. 6, p. 066114, 2011.

[9] J. Yang and J. Leskovec, "Overlapping community detection at scale: a nonnegative matrix factorization approach," in *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*. ACM, 2013, pp. 587–596.

[10] A. Nematzadeh, E. Ferrara, A. Flammini, and Y.-Y. Ahn, "Optimal network modularity for information diffusion," *Physical Review Letters*, vol. 113, no. 8, p. 088701, 2014.

[11] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2003, pp. 137–146.

[12] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

[13] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[14] R. G. Miller Jr, *Survival analysis*. John Wiley & Sons, 2011, vol. 66.

[15] O. Aalen, O. Borgan, and H. Gjessing, *Survival and event history analysis: a process point of view*. Springer Science & Business Media, 2008.

[16] C.-J. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural Computation*, vol. 19, no. 10, pp. 2756–2779, 2007.

[17] J. Xie, B. K. Szymanski, and X. Liu, "Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process," in *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*. IEEE, 2011, pp. 344–349.

[18] P. Csermely, A. London, L.-Y. Wu, and B. Uzzi, "Structure and dynamics of core/periphery networks," *Journal of Complex Networks*, vol. 1, no. 2, pp. 93–123, 2013.

[19] B. Recht, C. Re, S. Wright, and F. Niu, "Hogwild: A lock-free approach to parallelizing stochastic gradient descent," in *Advances in Neural Information Processing Systems*, 2011, pp. 693–701.

[20] L. Weng, F. Menczer, and Y.-Y. Ahn, "Predicting successful memes using network and community structure," *arXiv preprint arXiv:1403.6199*, 2014.

[21] J. Cheng, L. Adamic, P. A. Dow, J. M. Kleinberg, and J. Leskovec, "Can cascades be predicted?" in *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 2014, pp. 925–936.

[22] 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 *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 1513–1522.

[23] M. G. Rodriguez, D. Balduzzi, and B. Schölkopf, "Uncovering the temporal dynamics of diffusion networks," *arXiv preprint arXiv:1105.0697*, 2011.

[24] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, 1999.

[25] P. W. Holland, K. B. Laskey, and S. Leinhardt, "Stochastic blockmodels: First steps," *Social Networks*, vol. 5, no. 2, pp. 109–137, 1983.

[26] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.