Proceedings of the Ninth International AAAI Conference on Web and Social Media

CENI: A Hybrid Framework for Efficiently Inferring Information Networks

Qingbo Hu*, Sihong Xie*, Shuyang Lin*, Senzhang Wang†, Philip S. Yu*
*University of Illinois at Chicago, †Beihang University
{qhu5, sxie6, slin38, psyu}@uic.edu, szwang@cse.buaa.edu.cn

Abstract

Information links among users or websites drive the development of countless innovative applications. However, in reality, it is easier for us to observe the timestamps when different nodes in the network react on a message, while the connections empowering the information diffusion remain hidden. This motivates recent extensive studies on the ***network inference problem***: how to uncover the edges from the records of messages disseminated through them. Many existing solutions are computationally expensive, which motivates us to develop an efficient two-step framework, ***Clustering Embedded Network Inference*** (CENI). CENI integrates clustering strategies to improve the efficiency of network inference. By clustering nodes on the timelines of messages, we propose two naive implementations of CENI: ***Infection-centric CENI*** and ***Cascade-centric CENI***. We further point out the ***critical dimension*** problem of CENI: in order to preserve the node structure hidden in the cascades, we need to first project the nodes into an Euclidean space of certain dimension before clustering. By addressing this problem, we propose the third implementation of the CENI framework: ***Projection-based CENI***. Experiments on two real datasets show that the three CENI models only need around $20\% \sim 50\%$ of the running time of some state-of-the-art methods, while the inferred edges have similar or slightly better F-measure.

Introduction

With the assistance of online social media and networks, a piece of information has the potential to be widely spread within a short period of time. In real-life, industrial companies utilize such fact to develop innovative applications, such as promoting new products through online social networks. One of the most fundamental problems behind such applications is to infer the possibility to transmit information between two users by analyzing their online activities.

Network Inference Problem

In general, we name a message that diffuses through networks as a ***cascade*** and those users who react to the message as ***infected*** users. Ideally, cascades are able to reveal the information links of network users, if we can fully observe the information flow. In other words, if every infected user explicitly mentions from whom s/he receives the cascade, we are aware of the connection between him/her and his/her information source. However, this is far from the truth. In reality, the information flow is usually partially observable or completely unobservable. For example, the user connections in an online social network may not be entirely visible because of the privacy settings. In a more challenging scenario, a blog user may never mention the information source when s/he posts a blog. Fortunately, the time when each user receives a cascade is usually easy to be observed. As a result, how to uncover the connections among users from their infection time to different cascades, a.k.a. ***network inference problem***, has lately received extensive studies.

Recently, survival analysis based solutions to the network inference problem have burgeoned (Gomez-Rodriguez, Balduzzi, and Schölkopf 2011; Wang et al. 2014). The idea behind them is that regarding to a specific cascade, an infected user has a higher probability to be infected by another who has a closer infection time with respect to his/hers. Moreover, they assume such probability follows a distribution governed by a parameter associated with each pair of users, which can be estimated through Maximum Likelihood Estimation (MLE). The value of the parameter indicates the strength of the directed connection between two users. Since information might be passed between each pair of users in both directions, we need to estimate N^2 parameters, where N is the number of users. Such large number of parameters causes efficiency problems for survival analysis models. For instance, if two users stay far away in the infection timelines of most cascades, the models might end up with assigning a weak link between them. Inferring such links brings us extra computational burdens and is usually unnecessary, since they are normally viewed as noise and being discarded later.

In order to improve the efficiency of survival analysis based approaches, we may utilize the nodes' cluster structures. If we are aware of the clusters in the network, we may omit the weak inter-cluster connections when inferring the network from cascades. This reduces the size of our problem: clusters produce the division of cascades, since we only need to consider the potential influencers within a cluster for each node. Although it might lose some true inter-cluster edges, we may be able to control such lost to an acceptable level with a carefully designed clustering strategy. Besides the benefit of improving the efficiency, the precision of inferred edges may increase as well, since the clus-

ters might cut off incorrectly inferred edges of traditional approaches. However, in the setting of network inference problem, even the edges are not observed, not to mention the clusters in the network. In fact, this causes the clustering methods based on user connections not applicable. Moreover, the extracted clusters in this article aim to provide the best efficiency-effectiveness trade-off for the network inference, rather than finding communities with real-life meaning in the dataset. To extract the clustering structure from the cascades, we adopt a simple intuition: if two users' infection time is relatively close in observed cascades, the two users might have a strong connection and should be considered in the same cluster. This also indicates we might want the network inference method to spend time on estimating such strong intra-cluster connection. This idea leads to the proposed two-step hybrid framework: ***Clustering Embedded Network Inference*** (CENI). CENI first attempts to obtain the cluster structures from the cascades and then incorporate them with a network inference model to infer edges.

Critical Dimension

Obtaining clusters directly from the timelines of infection is straightforward. For example, we can assign a window for each infected node to identify its possible cluster members. Therefore, we first propose two implementations of the CENI framework, namely ***Infection-centric CENI*** (I-CENI) and ***Cascade-centric CENI*** (C-CENI), which attempt to naively find clusters based on the timelines. Practically, we find that although I-CENI and C-CENI are very efficient, they are not stable for all datasets: in some cases, compared to some state-of-the-art algorithms, I-CENI and C-CENI may lose much accuracy of inferred edges. We propose a possible explanation to this phenomena: the cluster structure hidden in the diffusion time gaps are too complicated to be captured by the one-dimensional approaches. Therefore, we may want to look for better clustering strategies on higher dimensional space. How to efficiently find a particular dimension of node representation so that the obtained clusters lead to high-quality inferred edges? We refer this specific dimension to the ***critical dimension*** of CENI.

To solve this problem, we propose another CENI model: ***Projection-based CENI*** (P-CENI). P-CENI first adopts a heuristic method by using a hinge loss estimator to identify the critical dimension. Afterwards, P-CENI obtain the embedded node representations in a space of the critical dimension and then apply a clustering algorithm to find the clusters for network inference. Through substantial experiments on two real datasets, we show that P-CENI runs in a similar time as I-CENI and C-CENI, while still preserves or even improves the effectiveness of state-of-the-art network inference methods, in terms of the F-measure of inferred edges.

Clustering Embedded Network Inference

Preliminaries

First, we introduce some important concepts and notations related to this article. We denote a network by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. A cascade, say c, can be represented by a vector of length N: $\boldsymbol{t}^c = (t_1^c, t_2^c, ...t_N^c)$, where t_i^c is the i^{th} user's infection time to cascade c, and N is the number of nodes in the network, a.k.a. $|\mathcal{V}| = N$. Additionally, we have $t_i^c \in [0, T] \cup \{\infty\}$, where T is a fixed length of the observation time window for all cascades, and ∞ denotes the user is not infected within the time window. Each cascade has a separate clock, and it is set to 0 at the time point when the first node is infected. The network inference problem can be stated as how to uncover hidden $\mathcal{E}$ from known $\mathcal{V}$ and $\boldsymbol{t}^c, c = \{1, 2, ..., M\}$, where M is the number of cascades. Next, we also assume there is a node subset associated with each node, which contains the members in the same cluster. We denote the subsets by $\mathcal{C} = \{C_1, C_2, ..., C_N\}$, where C_i contains the cluster members of the i^{th} node. Moreover, in some implementations of CENI, $\mathcal{C}$ might change with respect to cascades. In such cases, we replace $\mathcal{C}$ with $\mathcal{C}^c = \{C_1^c, C_2^c, ..., C_N^c\}$ for the ease and rigorousness of statement, where C_i^c denotes the cluster members of the i^{th} node in cascade c. As we stated previously, CENI is a two-step framework: the first phase derives $\mathcal{C}$ from observed cascades based on network nodes, and the second phase integrates the obtained $\mathcal{C}$ to a traditional network inference model to infer $\mathcal{E}$, which also requires the observed cascades as input. In this article, we emphasize the discussion of the implementation of the first phase, which essentially leads to three distinct CENI models. Meanwhile, we fix the approach in the inference phase as the same one for different CENI models to illustrate a better clustering strategy. More particularly, we slightly modify the NetRate (Gomez-Rodriguez, Balduzzi, and Schölkopf 2011) to fit in the second phase. A version that contains more details of this paper is also available online[1].

Inference Phase

If we have the cluster members for each node, we can integrate them into the *transmission likelihood* of NetRate model. The transmission likelihood, denoted by $f(t_i^c | t_j^c; \alpha_{j,i})$, refers to the conditional probability that an infected node i of cascade c, is infected by another node j, given the fact j is also a infected node of c. $f(t_i^c | t_j^c; \alpha_{j,i})$ is a function of the infection time of node i and j to cascade c, and $\alpha_{j,i}$ is a non-negative model parameter associated with edge $j \to i$. If the directed edge $j \to i$ exists, $\alpha_{j,i}$ has a positive value, otherwise $\alpha_{j,i} = 0$. Practically, we employ two types of transmission likelihood functions that are widely used in NetRate, i.e. an exponential distribution:

$$f(t_i^c | t_j^c; \alpha_{j,i}) = \begin{cases} \alpha_{j,i} \cdot e^{-\alpha_{j,i} \Delta_{j,i}^c} & t_j^c < t_i^c, j \in C_i \text{ or } C_i^c \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

and a Rayleigh distribution:

$$f(t_i^c | t_j^c; \alpha_{j,i}) = \begin{cases} \alpha_{j,i} \Delta_{j,i}^c \cdot e^{-\frac{1}{2}\alpha_{j,i} (\Delta_{j,i}^c)^2} & t_j^c < t_i^c, j \in C_i \text{ or } C_i^c \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where $\Delta_{j,i}^c = t_i^c - t_j^c$. The above two transmission likelihood functions assume that an infected node will not have further influence on any node who is already infected. Moreover, Eq. (1) and Eq. (2) have an additional assumption related

[1] http://arxiv.org/abs/1503.04927

to the cluster members: a node can only infect its cluster members, while the inter-cluster connections are neglected.

Survival function, denoted by $S(t_i|t_j;\alpha_{j,i})$, refers to the probability that a node j does **not** infect i by time t_i, given that the node j gets infected at time t_j. Thus, we have $S(t_i|t_j;\alpha_{j,i}) = 1 - \int_{t_j}^{t_i} f(t|t_j;\alpha_{j,i})dt$. **Hazard function**, denoted by $H(t_i|t_j;\alpha_{j,i})$, on the other hand, refers to the instantaneous infection rate, which is: $H(t_i|t_j;\alpha_{j,i}) = \frac{f(t_i|t_j;\alpha_{j,i})}{S(t_i|t_j;\alpha_{j,i})}$. With the notations of survival and hazard functions, the log-likelihood of a cascade c, denoted by $\ell_c(\boldsymbol{t}^c;\boldsymbol{A},\mathcal{C})$ can be readily computed (Gomez-Rodriguez, Balduzzi, and Schölkopf 2011), where $\boldsymbol{A}$ is the matrix notation of model parameters ($\boldsymbol{A_{i,j}} = \alpha_{i,j}$):

$$\begin{aligned}\ell_c(\boldsymbol{t}^c;\boldsymbol{A},\mathcal{C}) = &\sum_{\{t_i \le T\}} \sum_{\{m:t_m^c > T\}} \log S(T|t_i^c;\alpha_{i,m}) + \\ &\sum_{\{j:t_j^c < t_i^c, j \in C_i \text{ or } C_i^c\}} \log S(t_i^c|t_j^c;\alpha_{j,i}) + \\ &\log\left[\sum_{\{k:t_k^c < t_i^c, k \in C_i \text{ or } C_i^c\}} H(t_i^c|t_k^c;\alpha_{k,i})\right]\end{aligned} \tag{3}$$

Similar to the original NetRate, we view each cascade as independent from each other. Finally, CENI adopts gradient descent to solve the following convex optimization problem:

$$\begin{aligned}\underset{\boldsymbol{A}}{\text{minimize}} \quad & -\textstyle\sum_c \ell_c(\boldsymbol{t}^c;\boldsymbol{A},\mathcal{C}) \\ \text{subject to} \quad & \alpha_{i,j} \ge 0, i, j = 1, 2, ..., N\end{aligned} \tag{4}$$

Clustering Phase

Infection-centric CENI (I-CENI) Infection-centric CENI (I-CENI) imposes a time window to identify the cluster members for each node in each cascade. Assuming that all nodes are sorted in the ascending order of their infection time to cascade c, let $\tau(i,c)$ be the position of node i in the sorted list. Moreover, we define $\tau(i,c) = \infty$, if $t_i^c = \infty$. For a specific cascade c, node $i \in C_j^c$, if and only if $|\tau(i,c) - \tau(j,c)| <= \theta$, where θ is a predefined threshold. One should notice that the "clusters" in I-CENI are not conventional clusters as in many clustering problems: they are essentially the "neighborhoods" of nodes in the timelines of cascades. Moreover, such neighborhoods naturally change with respect to cascades, since each cascade is an independent timeline. For a clearer demonstration, we use the example of node u (marked by red color) in Fig. 1(a) to illustrate the clustering result of I-CENI, when $\theta = 2$.

Cascade-centric CENI (C-CENI) Cascade-centric CENI (C-CENI) adopts a K-means based strategy for clustering. We first view each t_i^c as a data point in a one-dimensional space and assume that we have a predefined cluster number, K, which can be determined by cross validations. Initially, we randomly select K centroids for each cluster. Next, at each iteration, we assign each data point to the cluster of which centroid is the closest, then we recalculate the new centroids by using the means of data points within the clusters. We continue to do this until it converges. Similar to I-CENI, a node's cluster members in C-CENI may also change with respect to cascades, since the infection time of this node may vary in different cascades. In Fig. 1(b), we illustrate the clustering result of C-CENI, when $K = 3$.

(a) I-CENI ($\theta = 2$) (b) C-CENI ($K = 3$)

Figure 1: Clustering results of I-CENI and C-CENI. In (a), red nodes are the centers of three time windows of size 2. The boundaries of windows are marked by square brackets, and the blue nodes are considered in the same cluster with u for each cascade. In (b), the boundaries of clusters are marked by dashed lines, and nodes in the same cluster share the same color. Different μs are the centroids of each cluster.

Projection-based CENI (P-CENI) Unlike I-CENI and C-CENI directly perform clustering on the timelines, P-CENI first projects each node into a space of certain dimension, say D. In other words, each node in the network will be endowed with unique D-dimensional coordinates. In fact, how to obtain the coordinates to explain the observed cascades is a relatively new problem and has several solutions very recently. The intuition of these solutions is that if the infection time of two nodes are relatively close in observed cascades, their positions in the projected space should be near enough to reflect such closeness. In the proposed P-CENI, we adopt the method in (Bourigault et al. 2014) to map nodes. If we denote the coordinates of node i by $\boldsymbol{z}_i$, and let s_c be the first infected user of cascade c, the projection procedure of P-CENI is to find $\{z_i|i = 1...N\}$ that minimizes the following hinge loss function, where $\max(x,y)$ is a function returning the larger one between x and y:

$$\sum_c \sum_{\{i|t_i^c < \infty\}} \sum_{\{j|t_j^c > t_i^c\}} \max\left(0, 1 - (\|\boldsymbol{z}_{s_c} - \boldsymbol{z}_i\|^2 - \|\boldsymbol{z}_{s_c} - \boldsymbol{z}_j\|^2)\right) \tag{5}$$

As introduced in (Bourigault et al. 2014), we adopt Monte Carlo approximation to get the solution that minimizes Eq. (5). In practice, we also find that Eq. (5) is a good heuristic estimator to find the critical dimension for P-CENI by iterating through different choices of D. After obtaining each $\boldsymbol{z}_i$, a K-means clustering algorithm is applied on the coordinates of nodes to obtain the cluster members for each node. Since each node i corresponds to a unique $\boldsymbol{z}_i$, its cluster members, C_i, does not change with respect to cascades in P-CENI , which is different from the cases in I-CENI and C-CENI.

Experiments

Description of Datasets

MemeTracker MemeTracker (Leskovec, Backstrom, and Kleinberg 2009) is a project to track the most frequently appeared quotes and phrases in numerous news and blog websites. In this dataset, each website is viewed as a node in the network. Websites can actively publish articles containing different memes, where each meme is deemed as a cascade. Moreover, articles may contain hyperlinks referring to the

information sources, which can be viewed as $\mathcal{E}$. Based on the raw data of Jan. 2009, we first extract 1,000 websites that post most memes, as well as all their published memes in Jan. 2009. T is set to 1 week, and the memes that do not have enough observation time are discarded. In total, we obtain 2,762 different cascades and 4,323 directed edges.

Sina Weibo Sina Weibo is a Twitter-like microblogging website that widely used in China. The second network is extracted from the raw data of a Sina Weibo dataset published by (Zhang et al. 2013). Based on the raw data in Aug. 2012, we extract 1,000 users who have most tweets and all their posts. The set of users are deemed as $\mathcal{V}$, and the users are connected through "follow" relationship. If a user follows another user, s/he can see the posted tweets of the followee and reacts to it by retweet or repost. Therefore, the "follow" relationship among users can empower the diffusion of tweets, which are viewed as cascades in this dataset. Among the 1,000 nodes, we obtain 1,671 "follow" edges to be used as $\mathcal{E}$. We set T to 2 weeks for each cascade, and any tweets do not have enough time to be observed are removed. In total, we get 2,907 different cascades.

Experimental Results

Based on the two extracted datasets, we compare the proposed three CENI models with two state-of-the-art network inference methods: NetRate (Gomez-Rodriguez, Balduzzi, and Schölkopf 2011) and MMRate (Wang et al. 2014). Since F-measure is a metric considering both precision and recall of the inferred edges, the reported overall performance of different algorithms are under the condition that every model is tuned to generate the optimal F-measure. The results on the MemeTracker dataset when the transmission likelihood function takes the exponential and Rayleigh forms are listed in Table 1. Accordingly, the results of the Sina Weibo dataset are listed in Table 2. In general, CENI models are much more efficient than MMRate and NetRate: they only need around 20% $\sim$ 50% of the running time of MMRate and NetRate on the two datasets. Among the three CENIs, C-CENI is usually the fastest one except in the MemeTracker dataset when transmission likelihood is a Rayleigh distribution. Although both I-CENI and C-CENI work generally well on the Sina Weibo dataset, they are relatively worse on the MemeTracker. P-CENI, on the other hand, is more stable and can produce edges of the highest F-measure in most cases. We set the number of iterations of the Monte Carlo simulation in the node projection step of P-CENI to 100,000, which produces robust results to be used to find the critical dimension and generate clusters. We iterates through 1 to 8 for D to find the critical dimension according to the value of Eq. (5). Comparing to the thousands of seconds spent on the network inference, finding the critical dimension only cost us less than 5 minutes for each dataset. Through more detailed analysis, we found out that although CENI models may have lower recalls than MMRate and NetRate, they normally have higher precisions. This leads to the F-measure of CENI models are similar to or even slightly better than the baselines. Considering both the computational time and the F-measure of inferred edges, we think the proposed CENI framework is more cost-effective than the compared state-of-the-art network inference methods.

	f	P-CENI	C-CENI	I-CENI	MMRate	NetRate
Time (sec.)	Exp.	8182	**7729**	8559	37328	16461
	Rayleigh	**9860**	10121	10446	46078	18378
F-measure	Exp.	**0.082**	0.059	0.073	0.070	0.071
	Rayleigh	**0.064**	0.032	0.039	0.012	0.049

Table 1: Results on the MemeTracker Dataset

	f	P-CENI	C-CENI	I-CENI	MMRate	NetRate
Time (sec.)	Exp.	5877	**5415**	7358	21677	14404
	Rayleigh	7724	**7408**	9247	33965	16219
F-measure	Exp.	0.145	0.142	0.137	**0.148**	0.142
	Rayleigh	**0.130**	0.126	0.108	0.128	0.117

Table 2: Results on the Sina Weibo Dataset

Conclusion

This article addresses the network inference problem, i.e. how to unveil the hidden network from the observed information cascades. We propose an innovative two-step framework, Clustering Embedded Network Inference (CENI), which aims to improve the efficiency of the network inference process, while still preserving the effectiveness. By adopting different clustering strategies, we develop three distinct CENI models: Infection-centric CENI (I-CENI), Cascade-centric CENI (C-CENI) and Projection-based CENI (P-CENI). Furthermore, we point out the critical dimension problem: in order to ensure the obtained clusters can eventually produce high-quality inferred edges, we may need to first estimate the dimensionality of the cascades and map the nodes into the space of that dimensionality. To solve this problem, we adopt a hinge loss estimator to find the critical dimension before actually performing clustering and the network inference. Through substantial experiments based on two datasets, the results demonstrate that the proposed three CENIs only need around 20% $\sim$ 50% of the running time of some state-of-the-art approaches, while the inferred edges have similar or even slightly better F-measure.

References

Bourigault, S.; Lagnier, C.; Lamprier, S.; Denoyer, L.; and Gallinari, P. 2014. Learning social network embeddings for predicting information diffusion. In *WSDM '14*, 393–402. ACM.

Gomez-Rodriguez, M.; Balduzzi, D.; and Schölkopf, B. 2011. Uncovering the temporal dynamics of diffusion networks. In *ICML '11*.

Leskovec, J.; Backstrom, L.; and Kleinberg, J. 2009. Meme-tracking and the dynamics of the news cycle. In *KDD '09*, 497–506. ACM.

Wang, S.; Hu, X.; Yu, P. S.; and Li, Z. 2014. Mmrate: Inferring multi-aspect diffusion networks with multi-pattern cascades. In *KDD '14*, 1246–1255. ACM.

Zhang, J.; Liu, B.; Tang, J.; Chen, T.; and Li, J. 2013. Social influence locality for modeling retweeting behaviors. In *IJCAI '13*, 2761–2767. AAAI Press.