Learnability of Influence in Networks

Harikrishna Narasimhan **David C. Parkes** **Yaron Singer**
Harvard University, Cambridge, MA 02138
hnarasimhan@seas.harvard.edu, {parkes, yaron}@seas.harvard.edu

Abstract

We establish PAC learnability of influence functions for three common influence models, namely, the Linear Threshold (LT), Independent Cascade (IC) and Voter models, and present concrete sample complexity results in each case. Our results for the LT model are based on interesting connections with neural networks; those for the IC model are based an interpretation of the influence function as an expectation over random draw of a subgraph and use covering number arguments; and those for the Voter model are based on a reduction to linear regression. We show these results for the case in which the cascades are only partially observed and we do not see the time steps in which a node has been influenced. We also provide efficient polynomial time learning algorithms for a setting with full observation, i.e. where the cascades also contain the time steps in which nodes are influenced.

1 Introduction

For several decades there has been much interest in understanding the manner in which ideas, language, and information cascades spread through society. With the advent of social networking technologies in recent years, digital traces of human interactions are becoming available, and the problem of predicting information cascades from these traces has gained enormous practical value. For example, this is critical in applications like viral marketing, where one needs to maximize awareness about a product by selecting a small set of influential users [1].

To this end, the spread of information in networks is modeled as an *influence function* which maps a set of seed nodes who initiate the cascade to (a distribution on) the set of individuals who will be influenced as a result [2]. These models are parametrized by variables that are unknown and need to be estimated from data. There has been much work on estimating the parameters of influence models (or the structure of the underlying social graph) from observed cascades of influence spread, and on using the estimated parameters to predict influence for a given seed set [3, 4, 5, 6, 7, 8]. These parameter estimation techniques make use of local influence information at each node, and there has been a recent line of work devoted to providing sample complexity guarantees for these local estimation techniques [9, 10, 11, 12, 13].

However, influence functions can be sensitive to errors in model parameters (as we shall see with an example in Section 2), and existing results do not tell us to what accuracy the individual parameters need to be estimated to obtain accurate influence predictions. Moreover, one cannot locally estimate the influence parameters when the cascades are only partially observed (e.g. when the time of influence of a node is not seen). If the primary goal in an application is to predict influence accurately, it is natural to ask for learnability guarantees on the influence function itself. A benchmark for studying such questions is the Probably Approximately Correct (PAC) learning framework [14]:

Are influence functions PAC learnable?

While many influence models have been popularized due to their approximation guarantees for influence maximization [2, 15, 16], learnability of influence is an equally fundamental property.

Part of this work was done when HN was a PhD student at the Indian Institute of Science, Bangalore.

In this paper, we establish PAC learnability for three well-studied influence models: the Linear Threshold, the Independent Cascade, and the Voter models. We primarily consider a setting where the cascades are *partially* observed, i.e. where only the nodes influenced and not the time steps at which they are influenced are observed. This is a setting where existing local estimation techniques cannot be applied to obtain parameter estimates. Additionally, for a *fully* observed setting where the time of influence is also observed, we show polynomial time learnability; our methods here are akin to using local estimation techniques, but come with guarantees on the global influence function.

Main results. Our learnability results are summarized below.

- **Linear threshold (LT) model:** Our result here is based on an interesting observation that LT influence functions can be seen as multi-layer neural network classifiers, and proceed by bounding their VC-dimension. The method analyzed here picks a function with zero training error. While this can be computationally hard to implement under partial observation, we provide a polynomial time algorithm for the full observation case using local computations.
- **Independent cascade (IC) model:** Our result uses an interpretation of the influence function as an expectation over random draw of a subgraph [2]; this allows us to show that the function is Lipschitz and invoke covering number arguments. The algorithm analyzed for partial observation is based on global maximum likelihood estimation. Under full observation (and additional assumptions), we show polynomial time learnability using a local estimation technique.
- **Voter model:** Our result follows from a reduction of the learning problem to a linear regression problem; the resulting learning algorithm can be implemented in polynomial time for both the full and partial observation settings.

Related work. A related problem to ours is that of inferring the structure of the underlying social graph from cascades [6]. There has been a series of results on polynomial sample complexity guarantees for this problem under variants of the IC model [9, 12, 10, 11]. Most of these results make specific assumptions on the cascades/graph structure, and assume a full observation setting. On the other hand, in our problem, the structure of the social graph is assumed to be known, and the goal is to provably learn the underlying influence function. Our results do not depend on assumptions on the network structure, and primarily apply to the more challenging partial observation setting.

The work that is most related to ours is that of Du et al. [13], who show polynomial sample complexity results for learning influence in the LT and IC models (under partial observation). However, their approach uses approximations to influence functions and consequently requires a strong technical condition to hold, which is not necessarily satisfied in general. Our results for the LT and IC models are some what orthogonal. While the authors in [13] trade-off assumptions on learnability and gain efficient algorithms that work well in practice, our goal is to show unconditional sample complexity for learning influence. We do this at the expense of the efficiency of the learning algorithms in the partial observation setting. Moreover, the technical approach we take is substantially different.

There has also been work on learnability of families of discrete functions such as submodular [17] and coverage functions [18] under both the PAC and the variant PMAC frameworks, with specific assumptions made on the input distribution. While the LT and IC influence functions can be seen as coverage functions (of exponentially large sizes) [2], existing PAC algorithms [18] that are applicable to these functions turn out to be improper learning algorithms, i.e. do not necessarily output an influence function of the desired form (or even a coverage function). In contrast, our focus is on *proper* learning of influence functions (see Appendix A for a discussion); this is important in applications where the learned function needs to be subsequently used for other tasks like influence maximization that crucially rely on the form of the function [2]. Moreover, since we look at specific function classes (rather than large families of discrete functions), our results apply to general seed distributions for most part. Other results relevant to our work include learnability of linear influence games [19], where the techniques used bear some similarity to our analysis for the LT model.

2 Preliminaries

Influence models. We represent a social network as a finite graph $G = (V, E)$, where the nodes $V = \{1, \ldots, n\}$ represent a set of n individuals and edges $E \subseteq V^2$ represent their social links. Let $|E| = r$. The graph is assumed to be directed unless otherwise specified. Each edge $(u, v) \in E$ is associated with a weight $w_{uv} \in \mathbb{R}_+$ that indicates the strength of influence of node v on node

u. We consider a setting where each node in the network holds an opinion in $\{0, 1\}$ and opinions disseminate in the network. This dissemination process begins with a small subset of nodes called the *seed* which have opinion 1 while the rest have opinion 0, and continues in discrete time steps. In every time step, a node may change its opinion from 0 to 1 based on the opinion of its neighbors, and according to some *local* model of influence; if this happens, we say that the node is *influenced*. We will use $N(u)$ to denote the set of neighbors of node u, and A_t to denote the set of nodes that are influenced at time step t. We consider three well-studied models:

- **Linear threshold (LT) model:** Each node u holds a *threshold* $r_u \in \mathbb{R}_+$, and is influenced at time t if the total incoming weight from its neighbors that were influenced at the previous time step $t-1$ exceeds the threshold: $\sum_{v \in N(u) \cap A_{t-1}} w_{uv} \geq k_u$. Once influenced, node u can then influence its neighbors for one time step, and never changes its opinion to 0.[1]
- **Independent cascade (IC) model:** Restricting edge weights w_{uv} to be in $[0, 1]$, a node u is influenced at time t independently by each neighbor v who was influenced at time $t-1$. The node can then influence its neighbors for one time step, and never changes its opinion to 0.
- **Voter model:** The graph is assumed to be undirected (with self-loops); at time step t, a node u adopts the opinion of its neighbor v with probability $w_{uv} / \sum_{v' \in N(u) \cup \{u\}} w_{uv'}$. Unlike the LT and IC models, here a node may change its opinion from 1 to 0 or 0 to 1 at every step.

We stress that a node is influenced ***at*** time t if it changes its opinion from 0 to 1 exactly at t. Also, in both the LT and IC models, *no node gets influenced more than once and hence an influence cascade can last for at most n time steps*. For simplicity, we shall consider in all our definitions only cascades of length n. While revisiting the Voter model in Section 5, we will look at more general cascades.

Definition 1 (Influence function). *Given an influence model, a (global) influence function $F : 2^V \to [0, 1]^n$ maps an initial set of nodes $X \subseteq V$ seeded with opinion 1 to a vector of probabilities $[F_1(X), \ldots, F_n(X)] \in [0, 1]^n$, where the u^{th} coordinate indicates the probability of node $u \in V$ being influenced during **any time step** of the corresponding influence cascades.*

Note that for the LT model, the influence process is deterministic, and the influence function simply outputs a binary vector in $\{0, 1\}^n$. Let $\mathcal{F}_G$ denote the class of all influence functions under an influence model over G, obtained for different choices of parameters (edge weights/thresholds) in the model. We will be interested in learning the influence function for a given parametrization of this influence model. We shall assume that the initial set of nodes that are seeded with opinion 1 at the start of the influence process, or the seed set, is chosen i.i.d. according to a distribution μ over all subsets of nodes. We are given a training sample consisting of draws of initial seed sets from μ, along with observations of nodes influenced in the corresponding influence process. Our goal is to then learn from $\mathcal{F}_G$ an influence function that best captures the observed influence process.

Measuring Loss. To measure quality of the learned influence function, we define a *loss function* $\ell : 2^V \times [0, 1]^n \to \mathbb{R}_+$ that for any subset of influenced nodes $Y \subseteq V$ and predicted influence probabilities $\mathbf{p} \in [0, 1]^n$ assigns a value $\ell(Y, \mathbf{p})$ measuring discrepancy between Y and $\mathbf{p}$. We define the error of a learned function $F \in \mathcal{F}_G$ for a given seed distribution μ and model parametrization as the expected loss incurred by F:

$$\text{err}^\ell[F] \;=\; \mathbf{E}_{X,Y}\big[\ell\big(Y, F(X)\big)\big],$$

where the above expectation is over a random draw of the seed set X from distribution μ and over the corresponding subsets of nodes Y influenced during the cascade.

We will be particularly interested in the difference between the error of an influence function $F_S \in \mathcal{F}_G$ learned from a training sample S and the minimum possible error achievable over all influence functions in $\mathcal{F}_G$: $\text{err}^\ell\big[F_S\big] - \inf_{F \in \mathcal{F}_G} \text{err}^\ell\big[F\big]$, and would like to learn influence functions for which this difference is guaranteed to be small (using only polynomially many training examples).

Full and partial observation. We primarily work in a setting in which we observe the nodes influenced in a cascade, but not the time step at which they were influenced. In other words, we assume availability of a *partial observed* training sample $S = \{(X^1, Y^1) \ldots, (X^m, Y^m)\}$, where X^i denotes the seed set of a cascade i and Y^i is the set of nodes influenced in that cascade. We will also consider a refined notion of *full observation* in which we are provided a training sample

[1] In settings where the node thresholds are unknown, it is common to assume that they are chosen randomly by each node [2]. In our setup, the thresholds are parameters that need to be learned from cascades.

$S = \{(X^1, Y^1_{1:n}) \ldots, (X^m, Y^m_{1:n})\}$, where $Y^i_{1:n} = \{Y^i_1, \ldots, Y^i_n\}$ and Y^i_t is the set of nodes in cascade i who were influenced precisely at time step t. Notice that here the complete set of nodes influenced in cascade i is given by $\bigcup_{t=1}^n Y^i_t$. This setting is particularly of interest when discussing learnability in polynomial time. The structure of the social graph is always assumed to be known.

PAC learnability of influence functions. Let $\mathcal{F}_G$ be the class of all influence functions under an influence model over a n-node social network $G = (V, E)$. We say $\mathcal{F}_G$ is *probably approximately correct (PAC) learnable w.r.t. loss* ℓ if there exists an algorithm s.t. the following holds for $\forall \epsilon, \delta \in (0, 1)$, for all parametrizations of the model, and for all (or a subset of) distributions μ over seed sets: when the algorithm is given a partially observed training sample $S = \{(X^1, Y^1), \ldots, (X^m, Y^m)\}$ with $m \geq \text{poly}(1/\epsilon, 1/\delta)$ examples, it outputs an influence function $F_S \in \mathcal{F}_G$ for which

$$\mathbf{P}_S\Big(\text{err}^\ell\big[F_S\big] - \inf_{F \in \mathcal{F}_G} \text{err}^\ell\big[F\big] \;\geq\; \epsilon\Big) \;\leq\; \delta,$$

where the above probability is over the randomness in S. Moreover, $\mathcal{F}_G$ is *efficiently PAC learnable* under this setting if the running time of the algorithm in the above definition is polynomial in m and in the size of G. We say $\mathcal{F}_G$ is *(efficiently) PAC learnable under full observation* if the above definition holds with a fully observed training sample $S = \{(X^1, Y^1_{1:n}), \ldots, (X^m, Y^m_{1:n})\}$.

Sensitivity of influence functions to parameter errors. A common approach to predicting influence under full observation is to estimate the model parameters using local influence information at each node. However, an influence function can be highly sensitive to errors in estimated parameters. E.g. consider an IC model on a chain of n nodes where all edge parameters are 1; if the parameters have all been underestimated with a constant error of ϵ, the estimated probability of the last node being influenced is $(1 - \epsilon)^n$, which is exponentially smaller than the true value 1 for large n. Our results for full observation provide concrete sample complexity guarantees for learning influence functions using local estimation, to any desired accuracy; in particular, for the above example, our results prescribe that ϵ be driven below $1/n$ for accurate predictions (see Section 4 on IC model). Of course, under partial observation, we do not see enough information to locally estimate the individual model parameters, and the influence function needs to be learned directly from cascades.

3 The Linear Threshold model

We start with learnability in the Linear Threshold (LT) model. Given that the influence process is deterministic and the influence function outputs binary values, we use the 0-1 loss for evaluation; for any subset of nodes $Y \subseteq V$ and predicted boolean vector $\mathbf{q} \in \{0, 1\}^n$, this is the fraction of nodes on which the prediction is wrong: $\ell_{0\text{-}1}(Y, \mathbf{q}) = \frac{1}{n}\sum_{u=1}^n \mathbf{1}(\chi_u(Y) \neq q_u)$, where $\chi_u(Y) = 1(u \in Y)$.

Theorem 1 (PAC learnability under LT model). *The class of influence functions under the LT model is PAC learnable w.r.t. $\ell_{0\text{-}1}$ and the corresponding sample complexity is $\widetilde{O}\big(\epsilon^{-1}(r + n)\big)$. Furthermore, in the full observation setting the influence functions can be learned in polynomial time.*

The proof is in Appendix B and we give an outline here. Let $F^{\mathbf{w}}$ denote a LT influence function with parameters $\mathbf{w} \in \mathbb{R}^{r+n}$ (edge weights and thresholds) and let us focus on the partial observation setting (only a node and not its time of influence is observed). Consider a simple algorithm that outputs an influence function with zero error on training sample $S = \{(X^1, Y^1), \ldots, (X^m, Y^m)\}$:

$$\frac{1}{m}\sum_{i=1}^m \ell_{0\text{-}1}\big(Y^i,\, F^{\mathbf{w}}(X^i)\big) \;=\; \frac{1}{mn}\sum_{i=1}^m\sum_{u=1}^n \mathbf{1}\big(\chi_u(Y^i) \neq F^{\mathbf{w}}_u(X^i)\big). \tag{1}$$

Such a function always exists as the training cascades are generated using the LT model. We will shortly look at computational issues in implementing this algorithm. We now explain our PAC learnability result for this algorithm. The main idea is in interpreting LT influence functions as neural networks with linear threshold activations. The proof follows by bounding the VC-dimension of the class of all functions $F^{\mathbf{w}}_u$ for node u, and using standard arguments in showing learnability under finite VC-dimension [20]. We sketch the neural network (NN) construction in two steps (local influence as a two-layer NN, and the global influence as a multilayer network; see Figure 1), where a crucial part is in ensuring that no node gets influenced more than once during the influence process:

1. **Local influence as a two-layer NN.** Recall that the (local) influence at a node u for previously influenced nodes Z is given by $\mathbf{1}\big(\sum_{v \in N(u) \cap Z} w_{uv} \;\geq\; k_u\big)$. This can be modeled as a linear (binary) classifier, or equivalently as a two-layer NN with linear threshold activations. Here the input layer

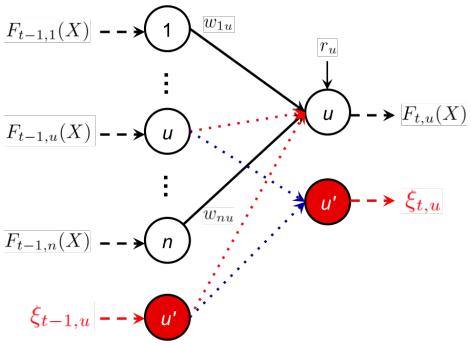

Figure 1: Modeling a single time step t of the influence process $F_{t,u} : 2^V \to \{0,1\}$ as a neural network ($t \geq 2$): the portion in black computes whether or not node u is influenced in the current time step t, while that in red/blue enforces the constraint that u does not get influenced more than once during the influence process. Here $\xi_{t,u}$ is 1 when a node has been influenced previously and 0 otherwise. The dotted red edges represent strong negative signals (has a large negative weight) and the dotted blue edges represent strong positive signals. The initial input to each node u in the input layer is $\mathbf{1}(u \in X)$, while that for the auxiliary nodes (in red) is 0.

contains a unit for each node in the network and takes a binary value indicating whether the node is present in Z; the output layer contains a binary unit indicating whether u is influenced after one time step; the connections between the two layers correspond to the edges between u and other nodes; and the threshold term on the output unit is the threshold parameter k_u. Thus the first step of the influence process can be modeled using a NN with two n-node layers (the input layer takes information about the seed set, and the binary output indicates which nodes got influenced).

2. **From local to global: the multilayer network.** The two-layer NN can be extended to multiple time steps by replicating the output layer once for each step. However, the resulting NN will allow a node to get influenced more than once during the influence process. To avoid this, we introduce an additional binary unit u' for each node u in a layer, which will record whether node u was influenced in previous time steps. In particular, whenever node u is influenced in a layer, a strong positive signal is sent to activate u' in the next layer, which in turn will send out strong negative signals to ensure u is never activated in subsequent layers[2]; we use additional connections to ensure that u' remains active there after. Note that a node u in layer $t+1$ is 1 whenever u is influenced at time step t; let $F^{\mathbf{w}}_{t,u} : 2^V \to \{0,1\}$ denote this function computed at u for a given seed set. The LT influence function $F^{\mathbf{w}}_u$ (which for seed set X is 1 whenever u is influenced in any one of the n time steps) is then given by $F^{\mathbf{w}}_u(X) = \sum_{t=1}^{n} F^{\mathbf{w}}_{t,u}(X)$. Clearly, $F^{\mathbf{w}}_u$ can be modeled as a NN with $n+1$ layers.

A naive application of classic VC-dimension results for NN [21] will give us that the VC-dimension of the class of functions F_u is $\widetilde{O}(n(r+n))$ (counting $r+n$ parameters for each layer). Since the same parameters are repeated across layers, this can be tightened to $\widetilde{O}(r+n)$. The remaining proof involves standard uniform convergence arguments [20] and a union bound over all nodes.

3.1 Efficient computation

Having shown PAC learnability, we turn to efficient implementation of the prescribed algorithm.

Partial observation. In the case where the training set does not specify the time at which each node was infected, finding an influence function with zero training error is computationally hard in general (as this is similar to learning a recurrent neural network). In practice, however, we can leverage the neural network construction, and solve the problem *approximately* by replacing linear threshold activation functions with sigmoidal activations and the 0-1 loss with a suitable continuous surrogate loss, and apply back-propagation based methods used for neural network learning.

Full observation. Here it turns out that the algorithm can be implemented in polynomial time using *local* computations. Given a fully observed sample $S = \{(X^1, Y^1_{1:n}), \ldots, (X^m, Y^m_{1:n})\}$, the loss of an influence function F for any $(X, Y_{1:n})$ is given by $\ell_{0\text{-}1}(\cup_{t=1}^{n} Y_t, F(X))$ and as before measures the fraction of mispredicted nodes. The prescribed algorithm then seeks to find parameters $\mathbf{w}$ for which the corresponding training error is 0. Given that the time of influence is observed, this problem can be decoupled into a set of linear programs (LPs) at each node; this is akin to locally estimating the parameters at each node. In particular, let $\mathbf{w}_u$ denote the parameters local to node u (incoming weights and threshold), and let $f_u(Z; \mathbf{w}_u) = \mathbf{1}\big(\sum_{v \in N(u) \cap Z} w_{uv} \geq k_u\big)$ denote the local influence at u for set Z of previously influence nodes. Let $\widehat{\alpha}_{1,u}(\mathbf{w}_u) = \frac{1}{m}\sum_{i=1}^{m} \mathbf{1}\big(\chi_u(Y^i_1) \neq f_u(X^i; \mathbf{w}_u)\big)$ and $\widehat{\alpha}_{t,u}(\mathbf{w}_u) = \frac{1}{m}\sum_{i=1}^{m} \mathbf{1}\big(\chi_u(Y^i_t) \neq f_u(Y^i_{t-1}; \mathbf{w}_u)\big)$, $t \geq 2$, that given the set of nodes Y^i_{t-1}

[2] By a strong signal, we mean a large positive/negative connection weight which will outweigh signals from other connections. Indeed such connections can be created when the weights are all bounded.

influenced at time $t-1$, measures the local prediction error at time t. Since the training sample was generated by a LT model, there always exists parameters such that $\widehat{\alpha}_{t,u}(\mathbf{w}_u) = 0$ for each t and u, which also implies that the overall training error is 0. Such a set of parameters can be obtained by formulating a suitable LP that can be solved in polynomial time. The details are in Appendix B.2.

4 The Independent Cascade model

We now address the question of learnability in the Independent Cascade (IC) model. Since the influence functions here have probabilistic outputs, the proof techniques we shall use will be different from the previous section, and will rely on arguments based on covering numbers. In this case, we use the squared loss which for any $Y \subseteq V$ and $\mathbf{q} \in [0,1]^n$, is given by: $\ell_{\text{sq}}(Y, \mathbf{q}) = \frac{1}{n}\sum_{u=1}^{n}[\chi_u(Y)(1-q_u)^2 + (1-\chi_u(Y))q_u^2]$. We shall make a mild assumption that the edge probabilities are bounded away from 0 and 1, i.e. $\mathbf{w} \in [\lambda, 1-\lambda]^r$ for some $\lambda \in (0, 0.5)$.

Theorem 2 (PAC learnability under IC model). *The class of influence functions under the IC model is PAC learnable w.r.t. ℓ_{sq} and the sample complexity is $m = \widetilde{O}\big(\epsilon^{-2}n^3r\big)$. Furthermore, in the full observation setting, under additional assumptions (see Assumption 1), the influence functions can be learned in polynomial time with sample complexity $\widetilde{O}(\epsilon^{-2}nr^3)$.*

The proof is given in Appendix C. As noted earlier, an IC influence function can be sensitive to errors in estimated parameters. Hence before discussing our algorithms and analysis, we seek to understand the extent to which changes in the IC parameters can produce changes in the influence function, and in particular, check if the function is Lipschitz. For this, we use the closed-form interpretation of the IC function as an expectation of an indicator term over a randomly drawn subset of edges from the network (see [2]). More specifically, the IC cascade process can be seen as activating a subset of edges in the network; since each edge can be activated at most once, the active edges can be seen as having been chosen apriori using independent Bernoulli draws. Consider a random subgraph of active edges obtained by choosing each edge $(u,v) \in E$ independently with probability w_{uv}. For a given subset of such edges $A \subseteq E$ and seed set $X \subseteq V$, let $\sigma_u(A, X)$ be an indicator function that evaluates to 1 if u is reachable from a node in X via edges in A and 0 otherwise. Then the IC influence function can be written as an expectation of σ over random draw of the subgraph:

$$F_u^{\mathbf{w}}(X) \;=\; \sum_{A \subseteq E} \prod_{(a,b)\in A} w_{ab} \prod_{(a,b)\notin A} (1-w_{ab})\; \sigma_u(A, X). \tag{2}$$

While the above definition involves an exponential number of terms, it can be verified that the corresponding gradient is bounded, thus implying that the IC function is Lipschitz.[3]

Lemma 3. *Fix $X \subseteq V$. For any $\mathbf{w}, \mathbf{w}' \in \mathbb{R}^r$ with $\|\mathbf{w} - \mathbf{w}'\|_1 \le \epsilon$, $\big|F_u^{\mathbf{w}}(X) \;-\; F_u^{\mathbf{w}'}(X)\big| \;\le\; \epsilon$.*

This result tells us how small the parameter errors need to be to obtain accurate influence predictions and will be crucially used in our learnability results. Note that for the chain example in Section 2, this tells us that the errors need to be less than $1/n$ for meaningful influence predictions.

We are now ready to provide the PAC learning algorithm for the partial observation setting with sample $S = \{(X^1, Y^1), \ldots, (X^m, Y^m)\}$; we shall sketch the proof here. The full observation case is outlined in Section 4.1, where we shall make use of the a different approach based on local estimation. Let $F^{\mathbf{w}}$ denote the IC influence function with parameters $\mathbf{w}$. The algorithm that we consider for partial observation resorts to a maximum likelihood (ML) estimation of the (global) IC function. Let $\chi_u(Y) = \mathbf{1}(u \in Y)$. Define the (global) log-likelihood for a cascade (X, Y) as:

$$\mathcal{L}(X, Y; \mathbf{w}) \;=\; \sum_{u=1}^{n} \chi_u(Y) \ln\big(F_u^{\mathbf{w}}(X)\big) \;+\; (1-\chi_u(Y)) \ln\big(1 - F_u^{\mathbf{w}}(X)\big),$$

The prescribed algorithm then solves the following optimization problem, and outputs an IC influence function $F^{\overline{\mathbf{w}}}$ from the solution $\overline{\mathbf{w}}$ obtained.

$$\max_{\mathbf{w} \in [\lambda, 1-\lambda]^r} \;\sum_{i=1}^{m} \mathcal{L}(X^i, Y^i; \mathbf{w}). \tag{3}$$

[3] In practice, IC influence functions can be computed through suitable sampling approaches. Also, note that a function class can be PAC learnable even if the individual functions cannot be computed efficiently.

To provide learnability guarantees for the above ML based procedure, we construct a finite ϵ-cover over the space of IC influence functions, i.e. show that the class can be approximated to a factor of ϵ (in the infinity norm sense) by a finite set of IC influence functions. We first construct an ϵ-cover of size $O((r/\epsilon)^r)$ over the space of parameters $[\lambda, 1-\lambda]^r$, and use Lipschitzness to translate this to an ϵ-cover of same size over the IC class. Following this, standard uniform convergence arguments [20] can be used to derive a sample complexity guarantee on the expected likelihood with a logarithmic dependence on the cover size; this then implies the desired learnability result w.r.t. ℓ_{sq}:

Lemma 4 (Sample complexity guarantee on the log-likelihood objective). *Fix $\epsilon, \delta \in (0,1)$ and $m = \widetilde{O}\big(\epsilon^{-2} n^3 r\big)$. Let $\overline{\mathbf{w}}$ be the parameters obtained from ML estimation. Then w.p. $\geq 1-\delta$,*

$$\sup_{\mathbf{w} \in [\lambda, 1-\lambda]^r} \mathbf{E}\left[\frac{1}{n}\mathcal{L}(X, Y; \mathbf{w})\right] \,-\, \mathbf{E}\left[\frac{1}{n}\mathcal{L}(X, Y; \overline{\mathbf{w}})\right] \,\leq\, \epsilon.$$

Compared to results for the LT model, the sample complexity in Theorem 2 has a square dependence on $1/\epsilon$. This is not surprising, as unlike the LT model, where the optimal 0-1 error is zero, the optimal squared error here is non-zero in general; in fact, there are standard sample complexity lower bound results that show that for similar settings, one cannot obtain a tighter bound in terms of $1/\epsilon$ [20].

We wish to also note that the approach of Du et al. (2014) for learning influence under partial observation [13] uses the same interpretation of the IC influence function as in Eq. (2), but rather than learning the parameters of the model, they seek to learn the weights on the individual indicator functions. Since there are exponentially many indicator terms, they resort to constructing approximations to the influence function, for which a strong technical condition needs to be satisfied; this condition need not however hold in most settings. In contrast, our result applies to general settings.

4.1 Efficient computation

Partial observation. The optimization problem in Eq. (3) that we need to solve for the partial observation case is non-convex in general. Of course, in practice, this can be solved approximately using gradient-based techniques, using sample-based gradient computations to deal with the exponential number of terms in the definition of $F^{\mathbf{w}}$ in the objective (see Appendix C.5).

Full observation. On the other hand, when training sample $S = \{(X^1, Y^1_{1:n}), \ldots, (X^m, Y^m_{1:n})\}$ contains fully observed cascades, we are able to show polynomial time learnability. For the LT model, we were assured of a set of parameters that would yield zero 0-1 error on the training sample, and hence the same procedure prescribed for partial information could be implemented under the full observation in polynomial time by reduction to local computations. This is not the case with the IC model, where we resort to the common approach of learning influence by estimating the model parameters through a *local* maximum likelihood (ML) estimation technique. This method is similar to the maximum likelihood procedure used in [9] for solving a different problem of recovering the structure of an unknown network from cascades. For the purpose of showing learnability, we find it sufficient to apply this procedure to only the first time step of the cascade.

Our analysis first provides guarantees on the estimated parameters, and uses the Lipschitz property in Lemma 3 to translate them to guarantees on the influence function. Since we now wish to give guarantees in the parameter space, we will require that there exists unique set of parameters that explains the IC cascade process; for this, we will need stricter assumptions. We assume that all edges have a minimum influence strength, and that even when all neighbors of a node u are influenced in a time step, there is a small probability of u not being influenced in the next step; we consider a specific seed distribution, where each node has a non-zero probability of (not) being a seed node.

Assumption 1. *Let $\mathbf{w}^*$ denote the parameters of the underlying IC model. Then there exists $\lambda \geq \gamma \in (0, 0.5)$ such that $w^*_{uv} \geq \lambda$ for all $(u,v) \in E$ and $\prod_{v \in N(u)}(1 - w_{uv}) \,\geq\, \gamma$ for all $u \in V$. Also, each node in V is chosen independently in the initial seed set with probability $\kappa \in (0,1)$.*

We first define the local log-likelihood for given seed set X and nodes Y_1 influenced at $t = 1$:

$$\mathcal{L}(X, Y_1; \boldsymbol{\beta}) \,=\, \sum_{u \notin X} \left[\chi_u(Y_1) \ln\left(1 - \exp\left(-\sum_{v \in N(u) \cap X} \beta_{uv}\right)\right) \,-\, (1 - \chi_u(Y_1)) \sum_{v \in N(u) \cap X} \beta_{uv}\right],$$

where we have used log-transformed parameters $\beta_{uv} = -\ln(1 - w_{uv})$, so that the objective is concave in $\boldsymbol{\beta}$. The prescribed algorithm then solves the following maximization problem over all

parameters that satisfy Assumption 1 and constructs an IC influence function from the parameters.

$$\max_{\boldsymbol{\beta} \in \mathbb{R}_+^r} \sum_{i=1}^{m} \mathcal{L}(X^i, Y_1^i; \boldsymbol{\beta}) \quad \text{s.t.} \quad \forall (u,v) \in E,\ \beta_{uv} \geq \ln\left(\frac{1}{1-\lambda}\right),\ \forall u \in V,\ \sum_{v \in N(u)} \beta_{uv} \geq \ln\left(\frac{1}{\gamma}\right).$$

This problem breaks down into smaller convex problems and can be solved efficiently (see [9]).

Proposition 5 (**PAC learnability under IC model with full observation**). *Under full observation and Assumption 1, the class of IC influence functions is PAC learnable in polynomial time through local ML estimation. The corresponding sample complexity is* $\widetilde{O}\big(nr^3(\kappa^2(1-\kappa)^4\lambda^2\gamma^2\epsilon^2)^{-1}\big)$.

The proof is provided in Appendix C.6 and proceeds through the following steps: (1) we use covering number arguments to show that the local log-likelihood for the estimated parameters is close to the optimal value; (2) we then show that under Assumption 1, the expected log-likelihood is *strongly concave*, which gives us that closeness to the true model parameters in terms of the likelihood also implies closeness to the true parameters in the parameter space; (3) we finally use the Lipschitz property in Lemma 3 to translate this to guarantees on the global influence function.

Note that the sample complexity here has a worse dependence on the number of edges r compared to the partial observation case; this is due to the two-step approach of requiring guarantees on the individual parameters, and then transferring them to the influence function. The better dependence on the number of nodes n is a consequence of estimating parameters locally. It would be interesting to see if tighter results can be obtained by using influence information in all time steps, and making different assumptions on the model parameters (e.g. correlation decay assumption in [9]).

5 The Voter model

Before closing, we sketch of our learnability results for the Voter model, where unlike previous models the graph is undirected (with self-loops). Here we shall be interested in learning influence for a fixed number of K time steps as the cascades can be longer than n. With the squared loss again as the loss function, this problem almost immediately reduces to linear least squares regression.

Let $\mathbf{W} \in [0,1]^{n \times n}$ be a matrix of normalized edge weights with $W_{uv} = w_{uv} / \sum_{v \in N(u) \cup \{u\}} w_{uv}$ if $(u,v) \in E$ and 0 otherwise. Note that $\mathbf{W}$ can be seen as a one-step probability transition matrix. Then for an initial seed set $Z \subseteq V$, the probability of a node u being influenced under this model after one time step can be verified to be $\mathbf{1}_u^\top \mathbf{W} \mathbf{1}_X$, where $\mathbf{1}_X \in \{0,1\}^n$ is a column vector containing 1 in entries corresponding to nodes in X, and 0 everywhere else. Similarly, for calculating the probability of a node u being influenced after K time steps, one can use the K-step transition matrix: $F_u(X) = \mathbf{1}_u^\top (\mathbf{W}^K) \mathbf{1}_X$. Now setting $\mathbf{b} = (\mathbf{W}^K)^\top \mathbf{1}_u$, we have $F_u(X) = \mathbf{b}^\top \mathbf{1}_X$ which is essentially a linear function parametrized by n weights.

Thus learning influence in the Voter model (for fixed cascade length) can be posed as linear regression with n^2 coefficients (n coefficients for each node). This can be solved in polynomial time even with partially observed data. We then have the following from standard results [20].

Theorem 6 (**PAC learnability under Voter model**). *The class of influence functions under the Voter model is PAC learnable w.r.t.* ℓ_{sq} *in polynomial time and the sample complexity is* $\widetilde{O}\big(\epsilon^{-2} n^2\big)$.

6 Conclusion

We have established PAC learnability of some of the most celebrated models of influence in social networks. Our results point towards interesting connections between learning theory and the literature on influence in networks. Beyond the practical implications of the ability to learn influence functions from cascades, the fact that the main models of influence are PAC learnable, serves as further evidence of their potent modeling capabilities. It would be interesting to see if our results extend to generalizations of the LT and IC models, and to investigate sample complexity lower bounds.

Acknowledgements. Part of this work was carried out while HN was visiting Harvard as a part of a student visit under the Indo-US Joint Center for Advanced Research in Machine Learning, Game Theory & Optimization supported by the Indo-US Science & Technology Forum. HN thanks Kevin Murphy, Shivani Agarwal and Harish G. Ramaswamy for helpful discussions. YS and DP were supported by NSF grant CCF-1301976 and YS by CAREER CCF-1452961 and a Google Faculty Research Award.

References

[1] Pedro Domingos and Matthew Richardson. Mining the network value of customers. In *KDD*, 2001.

[2] David Kempe, Jon M. Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *KDD*, 2003.

[3] Amit Goyal, Francesco Bonchi, and Laks VS Lakshmanan. Learning influence probabilities in social networks. In *KDD*, 2010.

[4] Manuel Gomez-Rodriguez, David Balduzzi, and Bernhard Schölkopf. Uncovering the temporal dynamics of diffusion networks. In *ICML*, 2011.

[5] Nan Du, Le Song, Alexander J. Smola, and Ming Yuan. Learning networks of heterogeneous influence. In *NIPS*, 2012.

[6] Manuel Gomez-Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. *ACM Transactions on Knowledge Discovery from Data*, 5(4):21, 2012.

[7] Nan Du, Le Song, Manuel Gomez-Rodriguez, and Hongyuan Zha. Scalable influence estimation in continuous-time diffusion networks. In *NIPS*, 2013.

[8] Abir De, Sourangshu Bhattacharya, Parantapa Bhattacharya, Niloy Ganguly, and Soumen Chakrabarti. Learning a linear influence model from transient opinion dynamics. In *CIKM*, 2014.

[9] Praneeth Netrapalli and Sujay Sanghavi. Learning the graph of epidemic cascades. In *SIGMETRICS*, 2012.

[10] Hadi Daneshmand, Manuel Gomez-Rodriguez, Le Song, and Bernhard Schölkopf. Estimating diffusion network structures: Recovery conditions, sample complexity & soft-thresholding algorithm. In *ICML*, 2014.

[11] Jean Pouget-Abadie and Thibaut Horel. Inferring graphs from cascades: A sparse recovery framework. *ICML*, 2015.

[12] Bruno D. Abrahao, Flavio Chierichetti, Robert Kleinberg, and Alessandro Panconesi. Trace complexity of network inference. In *KDD*, 2013.

[13] Nan Du, Yingyu Liang, Maria-Florina Balcan, and Le Song. Influence function learning in information diffusion networks. In *ICML*, 2014.

[14] Leslie G. Valiant. A theory of the learnable. *Commununications of the ACM*, 27(11):1134–1142, 1984.

[15] Elchanan Mossel and Sébastien Roch. On the submodularity of influence in social networks. In *STOC*, 2007.

[16] Eyal Even-Dar and Asaf Shapira. A note on maximizing the spread of influence in social networks. *Information Processing Letters*, 111(4):184–187, 2011.

[17] Maria-Florina Balcan and Nicholas J.A. Harvey. Learning submodular functions. In *STOC*, 2011.

[18] Vitaly Feldman and Pravesh Kothari. Learning coverage functions and private release of marginals. In *COLT*, 2014.

[19] Jean Honorio and Luis Ortiz. Learning the structure and parameters of large-population graphical games from behavioral data. *Journal of Machine Learning Research*, 16:1157–1210, 2015.

[20] Martin Anthony and Peter L. Bartlett. *Neural network learning: Theoretical foundations*. Cambridge University Press, 1999.

[21] Peter L. Bartlett and Wolfgang Maass. Vapnik Chervonenkis dimension of neural nets. *Handbook of Brain Theory and Neural Networks*, pages 1188–1192, 1995.

[22] Tong Zhang. Statistical behaviour and consistency of classification methods based on convex risk minimization. *Annals of Mathematical Statistics*, 32:56–134, 2004.

Learnability of Influence in Networks

Appendix

A Proper PAC Learning of Influence Functions

Our goal in this work is to develop algorithms that can provably learn the influence function for a given influence model, using cascades generated from the model. In other words, we are interested in *proper* PAC learnability of influence functions, and only consider algorithms that output a function from the specified influence class. This is in contrast with previous work on learnability of coverage functions [18]. While the influence functions in the Linear Threshold (LT) and Independent Cascade (IC) model can be seen as coverage functions (with an exponential large universe), the PAC algorithms [18] that have a polynomial sample complexity for these classes turn out to be improper learning algorithms. In other words, these algorithms are allowed to learn functions that do not necessarily have the functional form of the desired influence functions (the proper learning algorithms developed in [18] do not have polynomial sample complexity guarantees for coverage functions with exponential size).[4]

Indeed by construction, our PAC learning algorithms for the Linear Threshold and Independent Cascade models are proper learning algorithms, as they directly search over the parameter space of the influence model. Our algorithm for the Voter model works slightly differently, and searches over a transformed/aggregated parameter space based on the length of the given cascades. However, the function learned here still has the form of a Voter influence function, except that it is defined in terms transformed parameters. Hence, we also consider the prescribed PAC learning algorithm for the Voter model as a proper learning algorithm.

B Proofs/Additional Material for Section 3

Recall that in the LT model, given a set of nodes Z influenced at a previous time step, the local influence for a node u (that has not been influenced so far) is given by $\mathbf{1}\big(\sum_{v\in N(u)\cap Z} w_{uv} \geq k_u\big)$, where $\mathbf{w}$ denotes a vector of edge weights and threshold parameters of the model. Let us use the notation $f_u^{\mathbf{w}}$ to denote this local influence function at u (for ease of notation, we superscript f with $\mathbf{w}$ though the function is defined on only a subset of indices of $\mathbf{w}$ relevant to u; note that the notation here is slightly different from the one in the main text). In the following we shall sometimes overload notation and allow influence functions to take boolean membership vector in $\{0,1\}^n$ as inputs (rather than sets) with each entry u in the vector indicating whether node u is present in the seed set. We shall use $\text{VCdim}(\mathcal{F})$ to denote the VC-dimension of a (binary) function class $\mathcal{F}$. As before for any $Z \subseteq V$, we use the membership indicator $\chi_u(Z) = 1(u \in Z)$.

B.1 Proof of Theorem 1

We provide here the proof for the partial observation setting. The proof for the full observation setting is the same for most part, except that the training set contains more fine-grained information about the set of nodes influenced in each time step of a cascade $Y_1, \ldots, Y_n$, which for the purpose of the proof can all be aggregated into one set: $\bigcup_{t=1}^{n} Y_t$. Recall that the learning algorithm here simply picks an influence function with zero training error. To show that this procedure PAC learns from the LT class, we start by bounding the VC-dimension of the class of LT influence functions $F_u^{\mathbf{w}}$ for a given node u. The proof then follows from standard uniform convergence arguments for function classes with finite VC-dimension.

Lemma 7 (VC-dimension of global LT influence functions). *Fix node u. The class of all LT influence functions $F_u : 2^V \to \{0,1\}$ has a VC-dimension of at most $\widetilde{O}(r+n)$.*

Proof. We shall describe how the influence function $F_u^{\mathbf{w}}$ can be seen as a neural network and then extend classic results on the VC-dimension of neural networks to derive the VC-dimension of the class of all influence functions for node u. To build intuition, let us start with the neural network

[4] It is also important to note that the LT influence function is a coverage function only when the threshold is chosen uniformly at random at each node. In our setup, we consider deterministic thresholds.

construction for a simpler setting where a node, once influenced, can influence its neighbors in all subsequent time steps. While the resulting influence process can now last for more than n steps, we describe the construction for only n time steps. We shall then extend this network to the setting considered in this paper where a node can influence its neighbors only once.

1. **Local influence as a two-layer neural network.** Recall that the (local) influence at a node u is given by $f_u^{\mathbf{w}}(Z) = \mathbf{1}\big(\sum_{v \in N(u) \cap Z} w_{uv} \geq k_u\big)$. This function can be modeled as a linear (binary) classifier, or equivalently as a two-layer NN with linear threshold activations. Here the input layer contains a unit for each node in the social network and takes a binary value indicating whether the node is present in Z; the output layer contains a binary unit indicating whether u is influenced after one time step; the connections between the two layers correspond to the edges between u and other nodes; and the threshold term on the output unit is the threshold parameter k_u. Thus the first step of the influence process can be modeled using a NN with two n-node layers (the input layer takes information about the seed set, and output is a binary vector indicating which nodes got influenced).

2. **From local to global: the multilayer network.** The two-layer network can be extended to multiple time steps by replicating the second layer described above once for each step, along with the associated connections and thresholds. Additionally, let us add an edge from each node u to itself with a weight that exceeds threshold k_u. Thus once a node u is activated in a layer, it remains active thereafter. The LT influence function $F_u^{\mathbf{w}}$ (which outputs for any seed set, whether or not node u will be influenced in the corresponding cascade) is given by the status of node u in the last layer.

Thus $F_u^{\mathbf{w}}$ can be represented as a neural network with $n+1$ layers, with each layer containing $r+n$ parameters. If we ignore for a moment that the same parameters repeat across layers, an application of classic VC-dimension results for neural networks with $n(r+n)$ parameters, will give us that the VC-dimension of the class of all functions $F_u^{\mathbf{w}}$ for node u is at most $O\big((n(r+n))\log(n(r+n))\big)$. However, using a more careful analysis one can get a tighter bound of $O((r+n)\log(r+n))$. This is because with each new layer with the same connection weights, the ability of a neural network to shatter a subset of points can only reduce.

To see this, let us denote by $F_{t,u}^{\mathbf{w}} : \{0,1\}^n \to \{0,1\}$ the function computed at node u in layer $t+1$ for a given seed set encoded as binary vector in $\{0,1\}^n$ (recall that layer 1 is the input layer, and hence we only consider layer two onwards). Clearly, the function computed in the second layer $F_{1,u}^{\mathbf{w}}$ is equivalent to the local LT influence function $f_u^{\mathbf{w}}$, and that computed in the $(n+1)^{\text{th}}$ layer $F_{n,u}^{\mathbf{w}}$ is the required global influence function $F_u^{\mathbf{w}}$. Let $\mathcal{F}_{t,u}$ denote the class of all functions $F_{t,u}^{\mathbf{w}}$ under the LT model for different parameters $\mathbf{w} \in \mathbb{R}_+^{r+n}$. It is easy to see $\mathcal{F}_{1,u}$ is a class of linear binary classifiers with $r+1$ parameters, and hence we have from standard results that the $\text{VCdim}(\mathcal{F}_{1,u}) = r+1$. Similarly, $\mathcal{F}_{2,u}$ can be seen as a class of neural networks (linear threshold activations) with $O(r+n)$ parameters, and we have $\text{VCdim}(\mathcal{F}_{2,u}) = O((r+n)\ln(r+n))$. We shall now prove that $\text{VCdim}(\mathcal{F}_{t,u}) \leq O((r+n)\ln(r+n))$ for all $t \geq 3$. Consider a set of points $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subseteq \{0,1\}^n$ shattered by $\mathcal{F}_{t,k}$, $t \geq 3$. In other words, consider points such that

$$\begin{aligned}
2^N &= |\{F_{t,u}^{\mathbf{w}}(\mathbf{x}_1), \ldots, F_{t,u}^{\mathbf{w}}(\mathbf{x}_N) \mid \mathbf{w} \in \mathbb{R}_+^{r+n}\}| \\
&= |\{F_{t-1,u}^{\mathbf{w}}(F_{1,1}^{\mathbf{w}}(\mathbf{x}_1), \ldots, F_{1,n}^{\mathbf{w}}(\mathbf{x}_1)), \ldots, F_{t-1,u}^{\mathbf{w}}(F_{1,1}^{\mathbf{w}}(\mathbf{x}_N), \ldots, F_{1,N}^{\mathbf{w}}(\mathbf{x}_N)) \mid \mathbf{w} \in \mathbb{R}_+^{r+n}\}| \\
&= |\{F_{t-1,u}^{\mathbf{w}}(\mathbf{z}_1), \ldots, F_{t-1,u}^{\mathbf{w}}(\mathbf{z}_N) \mid \mathbf{w} \in \mathbb{R}_+^{r+n}\}|,
\end{aligned}$$

where $\mathbf{z}_1 = [F_{1,1}^{\mathbf{w}}(\mathbf{x}_1), \ldots, F_{1,n}^{\mathbf{w}}(\mathbf{x}_1)]$, …, $\mathbf{z}_N = [F_{1,1}^{\mathbf{w}}(\mathbf{x}_N), \ldots, F_{1,n}^{\mathbf{w}}(\mathbf{x}_N)] \in \{0,1\}^n$. Since $|\{F_{t-1,u}^{\mathbf{w}}(\mathbf{z}_1), \ldots, F_{t-1,u}^{\mathbf{w}}(\mathbf{z}_N) \mid \mathbf{w} \in \mathbb{R}_+^{r+n}\}| = 2^N$, it is necessarily the case that $\mathbf{z}_1, \ldots, \mathbf{z}_N$ are different (if not, not all binary assignments in $\{0,1\}^N$ can be realized). This implies that the set of points $\{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$ is shattered by $\mathcal{F}_{t-1,u}$. Thus for any set of points of a given size shattered by $\mathcal{F}_{t,u}$, there exists a set of points of the same size shattered by $\mathcal{F}_{t-1,u}$. This gives us that the VC-dimension of $\mathcal{F}_{t,u}$ is no greater than the VC-dimension of $\mathcal{F}_{t-1,u}$, i.e. $\text{VCdim}(\mathcal{F}_{t,u}) \leq \text{VCdim}(\mathcal{F}_{t-1,u})$ for all $t \geq 3$; applying this argument recursively, we have $\text{VCdim}(\mathcal{F}_{t,u}) \leq \text{VCdim}(\mathcal{F}_{2,u}) = O((r+n)\ln(r+n))$. Thus $\text{VCdim}(\mathcal{F}_{n,u}) \leq O((r+n)\ln(r+n))$.

The above result is for a simpler setting where a node, once influenced, can influence its neighbors in all subsequent time steps. In the setting that we consider in this paper, a node gets influenced

only by neighbors who were influenced in the previous time step, and moreover, a node cannot get influenced more than once during a cascade. To incorporate this additional constraint in the neural network structure, we introduce an additional binary unit u' for each node u in a layer, which will record whether node u was influenced in previous time steps. In particular, whenever node u is influenced in a layer, a strong positive signal is sent to activate u' in the next layer, which in turn will send out strong negative signals to ensure u is never activated in subsequent layer; we use additional connections to ensure that u' remains active there after. In the resulting neural network, a node u is activated in layer $t+1$ whenever u is influenced exactly at time step t; a node is never activated again in subsequent time steps (see Figure 1). Hence, if $F^{\mathbf{w}}_{t,u} : 2^V \to \{0,1\}$ is the function computed at node u in layer $t+1$, then the global LT influence function is given by $F^{\mathbf{w}}_u(X) = \sum_{t=1}^{n} F^{\mathbf{w}}_{t,u}(X)$. It can be verified that $F^{\mathbf{w}}_u$ can also be modeled as a neural network with $n+1$ layers and $r+n$ parameters. The same analysis used above can be retraced to show that the VC-dimension of all functions $F^{\mathbf{w}}_u$ for node u is $O((r+n)\ln(r+n))$.[5] □

We are now ready to prove our theorem.

Proof of Theorem 1. As before, μ denotes the distribution over the initial seed sets and $\mathbf{w}^*$ denotes the parameters of the underlying model; note that in this setting, $\inf_{\mathbf{w}\in\mathbb{R}^{r+n}_+} \text{err}^{0\text{-}1}[F^{\mathbf{w}}] = \text{err}^{0\text{-}1}[F^{\mathbf{w}^*}] = 0$; this also means that $\mathbf{E}\big[\mathbf{1}(\chi_u(Y)) \neq F^{\mathbf{w}^*}_u(X)\big] = 0 \ \forall u \in V$. Also, let $\overline{\mathbf{w}}$ denote the parameters obtained from Eq. (1); since $\overline{\mathbf{w}}$ minimizes the training error, we have for all u, $\frac{1}{m}\sum_{i=1}^{m} \mathbf{1}\big(\chi_u(Y^i) \neq F^{\overline{\mathbf{w}}}_u(X^i)\big) = 0$. We also know from Lemma 7, that for each u, the class of all influence functions $F^{\mathbf{w}}_u : 2^V \to \{0,1\}$ has a VC-dimension of $O((r+n)\ln(r+n))$. We can then use standard VC-dimension based learnability results for empirical risk minimization in settings where there is a function in the given function class that correctly labels all examples [20]. In particular, we have for any $\epsilon, \delta \in (0,1)$, and $m = O\Big(\dfrac{(r+n)\ln(r+n)\ln(1/\epsilon) + \ln(1/\delta)}{\epsilon}\Big)$, with probability at least $1-\delta$ (over draw of the training sample), $\mathbf{E}\big[\mathbf{1}(\chi_u(Y)) \neq F^{\overline{\mathbf{w}}}_u(X)\big] \leq \epsilon$. Taking a union bound over all of n nodes now gives us that when $m = O\Big(\dfrac{(r+n)\ln(r+n)\ln(1/\epsilon) + \ln(n/\delta)}{\epsilon}\Big)$, with probability at least $1-\delta$,

$$\text{err}^{0\text{-}1}[F^{\overline{\mathbf{w}}}] \;=\; \frac{1}{n}\sum_{u=1}^{n} \mathbf{E}\big[\mathbf{1}\big(\chi_u(Y) \neq F^{\overline{\mathbf{w}}}_u(X)\big)\big] \;\leq\; \epsilon.$$

This completes the proof. □

B.2 Formulating the learning problem as a LP under full observation

Let $n_u = |N(u)|$. Under full observation, the problem of obtaining parameters for which the local prediction error is zero for a given node $u \in V$ can be equivalently framed as the following linear program. Here the optimization is over $\mathbf{w}_u \in \mathbb{R}^{n_u+1}_+$ and over slack variables $\xi_{i,t}$ for each cascade i and time step t, subject to 'margin' constraints enforcing that the predicted influence status agrees with the true status of a node for each time step and training cascade.

$$\min_{\mathbf{w}_u \in \mathbb{R}^{n_u+1}_+,\, \xi_{i,t} \geq 0} \;\; \sum_{i=1}^{m}\sum_{t=1}^{n} \xi_{i,t}$$

$$\big(2\chi_u(Y^i_t) - 1\big)\Bigg(\sum_{v \in N(u) \cap Y^i_{t-1}} w_{uv} - k_u\Bigg) \;\geq\; 1 - \xi_{i,t}, \;\; \forall\, i \in [m],\, t \in [T].$$

Let $\mathbf{w}^* \in \mathbb{R}^{r+n}_+$ denote the parameters of the LT model from which the training sample was generated and $\mathbf{w}^*_u \in \mathbb{R}^{n_u+1}_+$ denote the parameters corresponding to node $u \in V$. Then $\mathbf{w}^*_u$ yields zero prediction error for node u; this also means that there exists a scaled version of $\mathbf{w}^*_u$ which yields optimal slack values $\xi^*_{i,t} = 0$ in the above problem. Clearly, solving the above LP will recover a scaled version of $\mathbf{w}^*_u$.

[5] Note that even with auxiliary connections with constant weights, the VC-dimension of the given class of neural networks is at most $O((r+n)\ln(r+n))$.

C Proofs/Additional Details for Section 4

Here, given a set of nodes $Z \subseteq V$ influenced at a time step, the probability of node u (that has not been influenced so far) being influenced in the next time step is $f_u^{\mathbf{w}}(Z) = 1 - \prod_{v \in N(u) \cap Z}(1 - w_{uv})$. As before, for any $Z \subseteq V$, $\chi_u(Z) = \mathbf{1}(u \in Z)$.

C.1 Proof of Theorem 2

We deal with the partial observation setting here. The full observation case is handled in the proof of Proposition 5 in Section C.6. The algorithm prescribed for the partial observation setting is a global maximum likelihood estimation described in Section 4; the specific optimization problem that needs to be solved is given in Eq. (3).

We start with an outline of the proof:

- We first show that the IC influence function $F_u^{\mathbf{w}}$ is 1-Lipschitz w.r.t. the L_1 norm (i.e. bounded changes in parameters only produce bounded changes in the function values). This was stated in Lemma 3 in the main text (restated below).

  Lemma 3. **(Lipschitzness of IC influence function w.r.t. L_1 norm).** Fix $X \subseteq V$. For any $\mathbf{w}, \mathbf{w}' \in \mathbb{R}^r$ with $\|\mathbf{w} - \mathbf{w}'\|_1 \leq \epsilon$, $\left|F_u^{\mathbf{w}}(X) - F_u^{\mathbf{w}'}(X)\right| \leq \epsilon$.

  Proof. See Section C.2. □

- We then establish an ϵ-cover over the space of parameters $[0,1]^r$ and translate this using the above Lipschitz property to a ϵ-cover over the space of IC influence functions, thus obtaining a bound on the covering number of this space.

  Lemma 8 (**Covering number of IC influence functions**). *The L_∞ covering number of the class of all IC influence functions F_u for radius ϵ is $O((r/\epsilon)^r)$.*

  Proof. See Section C.3. □

- Next, we appeal to standard uniform convergence arguments based on covering numbers [20] to bound the difference between the expected log-likelihood for the estimated parameters $\overline{\mathbf{w}}$ and that for the true parameters $\mathbf{w}^*$. This was stated in Lemma 4 in the main text (restated below).

  Lemma 4 (**Sample complexity guarantee on the log-likelihood objective**). Fix $\epsilon, \delta \in (0,1)$ and $m = \widetilde{O}\big(\epsilon^{-2} n^3 r\big)$. Let $\overline{\mathbf{w}}$ be the parameters obtained from global ML estimation. With probability at least $1 - \delta$ (over draw of the training sample), we have that

  $$\sup_{\mathbf{w} \in [\lambda, 1-\lambda]^r} \mathbf{E}\left[\frac{1}{n}\mathcal{L}(X, Y; \mathbf{w})\right] - \mathbf{E}\left[\frac{1}{n}\mathcal{L}(X, Y; \overline{\mathbf{w}})\right] \leq \epsilon.$$

  Proof. See Section C.4. □

- Finally, the above guarantee is translated into a bound on the difference between the expected squared error for $\overline{\mathbf{w}}$ and that for $\mathbf{w}^*$, as we shall see below.

Proof of Theorem 2. For PAC learnability in this setting, we need to show that $\text{err}^{\text{sq}}\big[F^{\overline{\mathbf{w}}}\big] - \inf_{\mathbf{w} \in \mathbb{R}_+^r} \text{err}^{\text{sq}}\big[F^{\mathbf{w}}\big] = \text{err}^{\text{sq}}\big[F^{\overline{\mathbf{w}}}\big] - \text{err}^{\text{sq}}\big[F^{\mathbf{w}^*}\big]$ can be made arbitrarily small w.h.p. Expanding this, we have

$$\begin{aligned}
&\text{err}^{\text{sq}}\big[F^{\overline{\mathbf{w}}}\big] - \text{err}^{\text{sq}}\big[F^{\mathbf{w}^*}\big] \\
&\quad = \mathbf{E}_{X,Y}\big[\ell_{\text{sq}}\big(Y, F^{\overline{\mathbf{w}}}(X)\big)\big] - \mathbf{E}_{X,Y}\big[\ell_{\text{sq}}\big(Y, F^{\mathbf{w}^*}(X)\big)\big] \\
&\quad = \mathbf{E}_{X,Y}\big[\ell_{\text{sq}}\big(Y, F^{\overline{\mathbf{w}}}(X)\big) - \ell_{\text{sq}}\big(Y, F^{\mathbf{w}^*}(X)\big)\big], \\
&\quad = \frac{1}{n}\sum_{u=1}^{n} \mathbf{E}_{X,Y}\Big[\chi_u(Y)(1 - F_u^{\overline{\mathbf{w}}}(X))^2 + (1 - \chi_u(Y))F_u^{\overline{\mathbf{w}}}(X)^2 \\
&\qquad\qquad - \chi_u(Y)(1 - F_u^{\mathbf{w}^*}(X))^2 - (1 - \chi_u(Y))F_u^{\mathbf{w}^*}(X)^2\Big]
\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{n}\sum_{u=1}^{n}\mathbf{E}_X\Big[\mathbf{E}_{Y|X}\Big[\chi_u(Y)(1-F_u^{\overline{\mathbf{w}}}(X))^2+(1-\chi_u(Y))F_u^{\overline{\mathbf{w}}}(X)^2 \\
&\qquad\qquad -\ \chi_u(Y)(1-F_u^{\mathbf{w}^*}(X))^2-(1-\chi_u(Y))F_u^{\mathbf{w}^*}(X)^2\Big]\Big] \\
&= \frac{1}{n}\sum_{u=1}^{n}\mathbf{E}_X\Big[F_u^{\mathbf{w}^*}(X)(1-F_u^{\overline{\mathbf{w}}}(X))^2+(1-F_u^{\mathbf{w}^*}(X))F_u^{\overline{\mathbf{w}}}(X)^2 \\
&\qquad\qquad -\ F_u^{\mathbf{w}^*}(X)(1-F_u^{\mathbf{w}^*}(X))^2-(1-F_u^{\mathbf{w}^*}(X))F_u^{\mathbf{w}^*}(X)^2\Big] \\
&= \frac{1}{n}\sum_{u=1}^{n}\mathbf{E}_X\Big[\big(F_u^{\overline{\mathbf{w}}}(X)\ -\ F_u^{\overline{\mathbf{w}}}(X)\big)^2\Big], \qquad (4)
\end{aligned}
$$

where the fifth step follows from the fact that for any X, $\mathbf{E}_Y[\chi_u(Y)\,|\,X]=F_u^{\mathbf{w}^*}(X)$.

We already have from Lemma 4 that when the number of training examples $m=\widetilde{O}\big(\epsilon^{-2}n^3r\big)$, we have with probability at least $1-\delta$ (over draw of training sample),

$$
\mathbf{E}_{X,Y}\left[\frac{1}{n}\mathcal{L}(X,Y;\mathbf{w}^*)\right]\ -\ \mathbf{E}_{X,Y}\left[\frac{1}{n}\mathcal{L}(X,Y;\overline{\mathbf{w}})\right]\ \le\ \epsilon.
$$

Expanding the left-hand side of the above inequality,

$$
\begin{aligned}
&\frac{1}{n}\mathbf{E}_{X,Y}\big[\mathcal{L}(X,Y;\mathbf{w}^*)\ -\ \mathcal{L}(X,Y;\overline{\mathbf{w}})\big] \\
&= \frac{1}{n}\sum_{u=1}^{n}\mathbf{E}_{X,Y}\big[\chi_u(Y)\ln(F_u^{\mathbf{w}^*}(X))\ +\ (1-\chi_u(Y))\ln(1-F_u^{\mathbf{w}^*}(X)) \\
&\qquad\qquad -\ \chi_u(Y)\ln(F_u^{\overline{\mathbf{w}}}(X))\ -\ (1-\chi_u(Y))\ln(1-F_u^{\overline{\mathbf{w}}}(X))\big] \\
&= \frac{1}{n}\sum_{u=1}^{n}\mathbf{E}_X\big[F_u^{\mathbf{w}^*}(X)\ln(F_u^{\mathbf{w}^*}(X))\ +\ (1-F_u^{\mathbf{w}^*}(X))\ln(1-F_u^{\mathbf{w}^*}(X)) \\
&\qquad\qquad -\ F_u^{\mathbf{w}^*}(X)\ln(F_u^{\overline{\mathbf{w}}}(X))\ -\ (1-F_u^{\mathbf{w}^*}(X))\ln(1-F_u^{\overline{\mathbf{w}}}(X))\big] \\
&= \frac{1}{n}\sum_{u=1}^{n}\mathbf{E}_X\big[L_{\log}\big(F_u^{\mathbf{w}^*}(X),F_u^{\mathbf{w}^*}(X)\big)\ -\ L_{\log}\big(F_u^{\mathbf{w}^*}(X),F_u^{\overline{\mathbf{w}}}(X)\big)\big] \\
&\ge \frac{1}{n}\sum_{u=1}^{n}\mathbf{E}_X\big[2\big(F_u^{\mathbf{w}^*}(X)-F_u^{\overline{\mathbf{w}}}(X)\big)^2\big],
\end{aligned}
$$

where the second equality follows from $E[\chi_u(Y)|X]=F_u^{\mathbf{w}^*}(X)$; in the second-last step last, we denote for any $\eta,\eta'\in[0,1]$, $L_{\log}(\eta',\eta)=\eta'\ln\big(\eta\big)+(1-\eta')\ln\big(1-\eta\big)$; the last step follows from the fact $L_{\log}(\eta',\eta')-L_{\log}(\eta',\eta)\ \ge\ 2(\eta-\eta')^2$ (this is easy to show; see e.g. Eq. (12) in [22]). Plugging this back into Eq. (4), the above implies that with probability at least $1-\delta$,

$$
\mathrm{err}^{\mathrm{sq}}\big[F^{\overline{\mathbf{w}}}\big]-\inf_{\mathbf{w}\in[\lambda,1-\lambda]}\mathrm{err}^{\mathrm{sq}}\big[F^{\mathbf{w}}\big]\ \le\ 0.5\epsilon,\qquad\text{as desired.}
$$

□

C.2 Proof of Lemma 3

Proof. We bound the L_∞ norm of the gradient of $F_u^{\mathbf{w}}$ by 1, which would imply that the function is 1-Lipschitz w.r.t. the L_1 norm. We have from Eq. (2), for any $(c,d)\in E$

$$
\begin{aligned}
&\left|\frac{\partial F_u^{\mathbf{w}}(Z)}{\partial w_{cd}}\right| \\
&= \left|\frac{\partial}{\partial w_{cd}}\Big[w_{cd}\sum_{A\subseteq E\setminus\{(c,d)\}}\prod_{(a,b)\in A}w_{ab}\prod_{(a,b)\notin A,(a,b)\neq(c,d)}(1-w_{ab})\sigma_u(A\cup\{(c,d)\},Z)\right. \\
&\qquad\qquad \left.+(1-w_{cd})\sum_{A\subseteq E\setminus\{(c,d)\}}\prod_{(a,b)\in A}w_{ab}\prod_{(a,b)\notin A,(a,b)\neq(c,d)}(1-w_{ab})\sigma_u(A,Z)\Big]\right|
\end{aligned}
$$

$$
\begin{aligned}
&= \Bigg| \sum_{A \subseteq E \setminus \{(c,d)\}} \prod_{(a,b) \in A} w_{ab} \prod_{(a,b) \notin A, (a,b) \neq (c,d)} (1 - w_{ab}) \sigma_u(A \cup \{(c,d)\}, Z) \\
&\qquad\qquad - \sum_{A \subseteq E \setminus \{(c,d)\}} \prod_{(a,b) \in A} w_{ab} \prod_{(a,b) \notin A, (a,b) \neq (c,d)} (1 - w_{ab}) \sigma_u(A, Z) \Bigg| \\
&\leq \Bigg| \sum_{A \subseteq E \setminus \{(c,d)\}} \prod_{(a,b) \in A} w_{ab} \prod_{(a,b) \notin A, (a,b) \neq (c,d)} (1 - w_{ab}) \Bigg| \\
&= 1,
\end{aligned}
$$

where the second last step follows from $0 \leq \sigma_u(A, Z) \leq 1$. Clearly $\|\nabla_{\mathbf{w}} F_u^{\mathbf{w}}(X)\|_\infty \leq 1$, which completes the proof of Lipschitzness of $F_u^{\mathbf{w}}$. □

C.3 Proof of Lemma 8

Proof. Note that the space of all parameters $\mathbf{w} \in [0,1]^r$ is bounded and can be covered by $(r/\epsilon)^r$ L_1-balls of radius ϵ. Further, from the above lemma we known that $F_u^{\mathbf{w}}$ is 1-Lipschitz w.r.t. the L_1 norm; we then have for any $\mathbf{w}, \mathbf{w}' \in [0,1]^r$:

$$
\max_{Z \subseteq V} \left| F_u^{\mathbf{w}}(Z) - F_u^{\mathbf{w}'}(Z) \right| \leq \|\mathbf{w} - \mathbf{w}'\|_1.
$$

This says that if the parameters of two influence functions are separated by a distance of ϵ in the L_1 space, the influence functions are also within an L_∞ distance of ϵ from each other. Clearly, an L_1 cover of radius ϵ over the parameter space can be translated to a L_∞ cover of the over the space of all influence functions for node u. In particular, if the parameter space is covered by R L_1-balls of radius ϵ and centers $\mathbf{w}_1, \ldots, \mathbf{w}_R$, then the influence functions $F^{\mathbf{w}_1}, \ldots, F^{\mathbf{w}_R}$ form a L_∞ cover of the space of influence functions, with the same radius. Thus the number of L_∞-balls of radius ϵ required to cover the space of influence functions is at most $O\big((r/\epsilon)^r\big)$. □

C.4 Proof of Lemma 4

The proof makes use of standard covering number based uniform convergence result for empirical risk minimization (or equivalently for log-likelihood maximization) over a real-valued function class [20]. To apply these standard results, we must ensure the log-likelihood is bounded and Lipschitz. We shall first establish this.

We once again use $n_u = |N(u)|$. Define for any $Z \subseteq V$, $y \in \{0,1\}$, $\mathbf{w} \in [\lambda, 1-\lambda]^r$ and $u \in V$, a function: $g_u(Z, y; \mathbf{w}) = y \ln\big(F_u^{\mathbf{w}}(Z)\big) + (1-y) \ln\big(1 - F_u^{\mathbf{w}}(Z)\big)$. Note that for a cascade (X, Y), $\mathcal{L}(X, Y; \mathbf{w}) = \frac{1}{n} \sum_{u=1}^n g_u(X, \chi_u(Y); \mathbf{w})$. In the following lemma, whenever we refer to a subset $Z \subseteq V$ in the context of a node u, *we shall assume that $u \notin Z$ and that there exists a path in the graph from a node in Z to u*; cases where this assumption fails can be easily handled, but have been ignored here to make the proof easier to follow. Below, we show that g_u is bounded and Lipschitz for any u.

Lemma 9 (Boundedness and Lipschitz continuity of log-likelihood function). *Fix parameters $\mathbf{w} \in [\lambda, 1-\lambda]^r$. Then*

1. $\lambda^n \leq F_u^{\mathbf{w}}(Z) \leq 1 - \lambda^n$.
2. $|g_u(Z, y; \mathbf{w})| \leq n \ln(1/\lambda)$.
3. $g_u(Z, y; \mathbf{w})$ *is $1/\lambda^n$-Lipschitz in $\mathbf{w}$ w.r.t. the L_1 norm.*

Proof.

1. Starting with the lower bound, recall the interpretation of the IC influence function in Eq. (2) as an expectation of an indicator term over random draw of a subgraph. From this interpretation, it is clear that the probability of node u being influenced is at least the probability that all edges in a path from a node in Z to u get activated. Since the minimum probability on any edge is λ and the length of any path can be at most n, we have

$F_u^{\mathbf{w}}(Z) \geq \lambda^n$. For the upper bound, note that the probability of u *not* being influenced in any of n time steps for a seed set Z is at least the probability that none of the neighbors of u ever influenced it (i.e. none of the incoming edges incident on u got activated): $\prod_{v \in N(u)}(1 - w_{uv}) \geq (1 - (1 - \lambda))^{n_u} \geq \lambda^n$. Hence $F_u^{\mathbf{w}}(Z) \leq 1 - \lambda^n$.

2. $|g_u(Z, y; \mathbf{w})| = |y \ln\left(F_u^{\mathbf{w}}(Z)\right) + (1-y) \ln\left(1 - F_u^{\mathbf{w}}(Z)\right)| \leq |y \ln(\lambda^n) + (1-y)\ln(\lambda^n)| \leq n \ln(\lambda)$ (from lower and upper bounds on $F_u^{\mathbf{w}}$ derived above and from $\lambda < 1$).

3. To show Lipschitzness of g_u w.r.t. L_1 norm, we bound the L_∞ norm of its gradient. In particular, $g_u(Z, y; \mathbf{w}) = y \ln\left(1 - F_u^{\mathbf{w}}(Z)\right) - (1 - y) \ln\left(F_u^{\mathbf{w}}(Z)\right)$ and

$$\nabla_{\mathbf{w}}\, g_u(Z, y; \mathbf{w}) \quad = \quad \left[\frac{y}{F_u^{\mathbf{w}}(Z)} - \frac{1-y}{1 - F_u^{\mathbf{w}}(Z)}\right] \nabla_{\mathbf{w}} F_u^{\mathbf{w}}(Z).$$

Since $1 - \lambda^n \geq F_u^{\mathbf{w}}(Z) \geq \lambda^n$, we have

$$\left|\frac{y}{F_u^{\mathbf{w}}(Z)} - \frac{1-y}{1 - F_u^{\mathbf{w}}(Z)}\right| \leq \frac{1}{\lambda^n}.$$

In addition, from the Lipschitz property of the IC influence function in Lemma 3, we know its gradient norm is bounded by 1,

$$\left\|\nabla_{\mathbf{w}}\, g_u(Z, y; \mathbf{w})\right\|_\infty \leq \frac{1}{\lambda^n} \left\|\nabla_{\mathbf{w}} F_u^{\mathbf{w}}(Z)\right\|_\infty \leq \frac{1}{\lambda^n}(1).$$

Hence g_u is $1/\lambda^n$-Lipschitz in $\mathbf{w}$ w.r.t. the L_1 norm.

□

Proof of Lemma 4. Let $\overline{\mathbf{w}}$ be the parameters obtained by solving Eq. (3). Similarly, let $\mathbf{w}^* \in [\lambda, 1 - \lambda]^r$ be the the underlying model parameters. Since the cascades are generated from an IC model defined by $\mathbf{w}^*$, one can verify that maximizing the expected log-likelihood $\mathbf{E}_{X,Y}\left[\mathcal{L}(X, Y; \mathbf{w})\right]$ over all $\mathbf{w} \in [\lambda, 1 - \lambda]^r$ yields $\mathbf{w}^*$. As mentioned above, the proof involves an application of standard covering number based uniform convergence arguments[20]; we shall make use of the covering number result in Lemma 8 and the Lipschitzness and boundedness of the likelihood shown in Lemma 9.

First, let us write the likelihood objective in Eq. (3) in terms of g_u.

$$\frac{1}{mn} \sum_{i=1}^{m} \mathcal{L}(X^i, Y^i; \mathbf{w}) = \frac{1}{n} \sum_{u=1}^{n} \underbrace{\frac{1}{m} \sum_{i=1}^{m} g_u(X, \chi_u(Y^i); \mathbf{w})}_{\widehat{G}_u(\mathbf{w})}. \tag{5}$$

Similarly, the expected log-likelihood can be written as

$$\frac{1}{n} \mathbf{E}_{X,Y}\left[\mathcal{L}(X, Y; \mathbf{w})\right] = \frac{1}{n} \sum_{u=1}^{n} \underbrace{\mathbf{E}_{X,Y}\left[g_u(X, \chi_u(Y); \mathbf{w})\right]}_{G_u(\mathbf{w})}. \tag{6}$$

We proceed by bounding the difference between the expected and empirical log-likelihood objective for any model vector, and use this to bound the difference between the optimal likelihood and the likelihood value of $\overline{\mathbf{w}}$.

We know from Lemma 9 that g_u is bounded by $n \ln(1/\lambda)$ and is $1/\lambda^n$-Lipschitz in $\mathbf{w}$. We can then invoke standard uniform convergence arguments based on the covering number result in Lemma 8, followed by a union bound over all nodes, to bound the difference between G_u and $\widehat{G}_u$. In particular, when $m = O\left(n^2 \ln(1/\lambda)^2 \frac{r \ln(r/\epsilon) + nr \ln(1/\lambda) + \ln(n/\delta)}{\epsilon^2}\right)$, with probability at least $1 - \delta$ (over draw of training sample), for each $u \in V$ and all $\mathbf{w} \in [\lambda, 1 - \lambda]^r$,

$$|G_u(\mathbf{w}) - \widehat{G}_u(\mathbf{w})| \leq \epsilon/2.$$

Substituting this back into Eq. (5) and (6), gives us with probability at least $1-\delta$, for all $\mathbf{w} \in [\lambda, 1-\lambda]^r$,

$$\left| \mathbf{E}_{X,Y}\left[\frac{1}{n}\mathcal{L}(X,Y;\mathbf{w})\right] - \frac{1}{mn}\sum_{i=1}^{m}\mathcal{L}(X^i,Y^i;\mathbf{w}) \right| \;\leq\; \frac{1}{n}\sum_{u=1}^{n}\epsilon/2 \;=\; \epsilon/2. \tag{7}$$

The above bound will then allow us to in turn bound the difference between the optimal log-likelihood and the log-likelihood of $\overline{\mathbf{w}}$, as shown below:

$$\begin{aligned}
&\sup_{\mathbf{w}\in[\lambda,1-\lambda]^r} \mathbf{E}_{X,Y}\left[\frac{1}{n}\mathcal{L}(X,Y;\mathbf{w})\right] - \mathbf{E}_{X,Y}\left[\frac{1}{n}\mathcal{L}(X,Y;\overline{\mathbf{w}})\right] \\
&= \mathbf{E}_{X,Y}\left[\frac{1}{n}\mathcal{L}(X,Y;\mathbf{w}^*)\right] - \mathbf{E}_{X,Y}\left[\frac{1}{n}\mathcal{L}(X,Y;\overline{\mathbf{w}})\right] \\
&= \mathbf{E}_{X,Y}\left[\frac{1}{n}\mathcal{L}(X,Y;\mathbf{w}^*)\right] - \frac{1}{mn}\sum_{i=1}^{m}\mathcal{L}(X^i,Y^i;\overline{\mathbf{w}}) \\
&\qquad\qquad + \frac{1}{mn}\sum_{i=1}^{m}\mathcal{L}(X^i,Y^i;\overline{\mathbf{w}}) - \mathbf{E}_{X,Y}\left[\frac{1}{n}\mathcal{L}(X,Y;\overline{\mathbf{w}})\right] \\
&\leq \left[\mathbf{E}_{X,Y}\left[\frac{1}{n}\mathcal{L}(X,Y;\mathbf{w}^*)\right] - \frac{1}{mn}\sum_{i=1}^{m}\mathcal{L}(X^i,Y^i;\mathbf{w}^*)\right] \\
&\qquad\qquad + \left[\frac{1}{mn}\sum_{i=1}^{m}\mathcal{L}(X^i,Y^i;\overline{\mathbf{w}}) - \mathbf{E}_{X,Y}\left[\frac{1}{n}\mathcal{L}(X,Y;\overline{\mathbf{w}})\right]\right] \\
&\leq \epsilon/2 + \epsilon/2 \;=\; \epsilon,
\end{aligned}$$

where the second-last step uses the fact that $\overline{\mathbf{w}}$ is the empirical maximizer of the log-likelihood, and the last step follows from Eq. (7). This completes the proof. □

C.5 Gradient Computation for Likelihood in Eq. (3)

We prescribe that the optimization problem in Eq. (3) be solved using a suitable gradient-based solver. We describe here how the gradient for the objective can be computed approximately by sampling subgraphs from G. In particular, for any $Z, Y \subseteq V$, and $(c,d) \in E$

$$\begin{aligned}
\frac{\partial \mathcal{L}(Z,Y;\mathbf{w})}{\partial w_{cd}} &= \frac{\partial}{\partial w_{cd}}\left[\sum_{u=1}^{n}\chi_u(Y)\ln\left(F_u^{\mathbf{w}}(X)\right) + (1-\chi_u(Y))\ln\left(1-F_u^{\mathbf{w}}\right)\right] \\
&= \sum_{u=1}^{n}\left[\frac{\chi_u(Y)}{F_u^{\mathbf{w}}(X)} - \frac{1-\chi_u(Y)}{1-F_u^{\mathbf{w}}(X)}\right]\frac{\partial F_u^{\mathbf{w}}(Z)}{\partial w_{cd}}.
\end{aligned}$$

Further,

$$\begin{aligned}
\frac{\partial F_u^{\mathbf{w}}(Z)}{\partial w_{cd}} &= \frac{\partial}{\partial w_{cd}}\Bigg[w_{cd}\sum_{A\subseteq E\setminus\{(c,d)\}}\prod_{(a,b)\in A} w_{ab}\prod_{(a,b)\notin A,\,(a,b)\neq(c,d)}(1-w_{ab})\sigma_u(A\cup\{(c,d)\},Z) \\
&\qquad + (1-w_{cd})\sum_{A\subseteq E\setminus\{(c,d)\}}\prod_{(a,b)\in A} w_{ab}\prod_{(a,b)\notin A,\,(a,b)\neq(c,d)}(1-w_{ab})\sigma_u(A,Z)\Bigg] \\
&= \sum_{A\subseteq E\setminus\{(c,d)\}}\prod_{(a,b)\in A} w_{ab}\prod_{(a,b)\notin A,\,(a,b)\neq(c,d)}(1-w_{ab})\sigma_u(A\cup\{(c,d)\},Z) \\
&\qquad - \sum_{A\subseteq E\setminus\{(c,d)\}}\prod_{(a,b)\in A} w_{ab}\prod_{(a,b)\notin A,\,(a,b)\neq(c,d)}(1-w_{ab})\sigma_u(A,Z) \qquad (8) \\
&= \underbrace{\sum_{A\subseteq E\setminus\{(c,d)\}}\mathbf{P}_{(c,d)}[A]\,\sigma_u(A\cup\{(c,d)\},Z)}_{\text{term}_1} - \underbrace{\sum_{A\subseteq E\setminus\{(c,d)\}}\mathbf{P}_{(c,d)}[A]\,\sigma_u(A,Z)}_{\text{term}_2},
\end{aligned}$$

where $\mathbf{P}_{(c,d)}[A]$ denotes the probability of sampling the edge subset A when each edge $(u,v) \neq (c,d)$ is chosen independently with probability w_{uv}. Thus to compute the gradient of optimization objective in Eq. (3), we will need to evaluate the values of $F_u^{\mathbf{w}}$, term$_1$ and term$_2$ for every node u and training example. While each of these involve a summation over an exponential number of subgraphs, they can essentially be seen as expectations and estimated through suitable sampling-based approaches.

C.6 Proof of Proposition 5

We now move to the fully observation setting. Here the algorithm that we analyze performs local maximum likelihood estimation to estimate the parameters of the IC model (see Section 4.1). The specific objective optimized is restated below:

$$\begin{aligned}\sum_{i=1}^{m} \mathcal{L}(X^i, Y_1^i; \mathbf{w}) &= \sum_{i=1}^{m} \sum_{u \notin X^i} \left[\chi_u(Y_1^i) \ln\left(f_u^{\mathbf{w}}(X^i)\right) + (1-\chi_u(Y_1^i)) \ln\left(1 - f_u^{\mathbf{w}}(X^i)\right)\right] \\ &= \sum_{i=1}^{m} \sum_{u=1}^{n} \left[\chi_u(Y_1^i) \ln\left(f_u^{\mathbf{w}}(X^i)\right) + (1-\chi_u(Y_1^i)) \ln\left(1 - f_u^{\mathbf{w}}(X^i)\right)\right] \mathbf{1}(u \notin X^i),\end{aligned} \tag{9}$$

where notice that the likelihood is not evaluated on nodes that are already present in the seed set. Note that we did not have this issue with the partial observation case, as there the global influence function $F_u(X)$, by definition, would evaluate to 1 whenever X contains u (as u is influenced even before the cascade begin; see Eq. (2)). On the other hand, the local influence function $f_u^{\mathbf{w}}(X)$ need not evaluate to 1 when $u \in X$ and hence this case is ignored in the above objective.

Our analysis involves first showing guarantees on the estimated parameters, and transferring them to guarantees on the global IC influence function. Unlike the partial observation case, here we seek to derive optimality guarantees on the parameters themselves, and require stricter assumptions.

Discussion on Assumption 1 In particular, the following are the assumptions we make:

1. All edges have a minimum influence strength of λ. Note that the graph can still contain a node that has no influence on its neighbor, by not having an edge between the two nodes.
2. Even when all neighbors of a node are influenced in a time step, there is a small probability $\gamma > 0$ of the node not being influenced in the next step. Thus expect for the case where none of a node's neighbors are present in the seed set, there is always a small probability of the node not being influenced in the first time step.
3. The seed distribution is such that each node is chosen independently with probability $\kappa \in (0,1)$.

The first and second assumptions ensure that the IC influence function and hence the log-likelihood function is bounded, a property which is crucial to guarantee learnability. The third assumption avoids pathological cases where the support of the seed distribution only covers a subset of nodes (in which case, we will not be able to learn anything about the remaining nodes), or has its entire probability mass concentrated on the full set V (in which case, we again learn nothing about the individual edge probabilities). Indeed our analysis will go through if in place of the second assumption, we just restricted the edge probabilities to be upper bounded by a value below 1, and the third assumption allows for more general distributions with appropriate support. We have retained these slightly stricter assumptions so that analysis is cleaner and easier to follow.

We begin rewriting the local influence functions $f_u^{\mathbf{w}}$ in terms of transformed parameters $\beta_{uv} = -\ln(1-w_{uv})$: $f_u^{\beta}(Z) = 1 - \exp\left(-\sum_{v \in N(u) \cap Z} \beta_{uv}\right)$, where $\sigma(s) = 1 - \exp(-s)$. Let us use the notation $\boldsymbol{\beta}_u$ to denote the vector of parameters β_{uv}, $v \in N(u)$. Also, recall that the prescribed local estimation procedure solves an optimization over all parameters that satisfy Assumption 1. Due to this, in our analysis, we can safely assume that the parameters are bounded in a certain range. In particular, it is clear that for all $(u,v) \in E$, $w_{uv} \geq \lambda$. One can also derive an upper bound from Assumption 1 as follows: for any $(u,v) \in E$, $w_{uv} = 1 - (1 - w_{uv}) \leq 1 - \prod_{v' \in N(u)}(1 - w_{uv'}) \leq 1 - \gamma$. Translating these bounds to the log-transformed space, we conclude that the log-transformed parameters $\boldsymbol{\beta} \in \mathbb{R}_+^r$ satisfy: $-\ln(1-\lambda) \leq \beta_{uv} \leq -\ln(\gamma)$.

We are now ready to sketch the proof of Proposition 5. Let $\overline{\mathbf{w}}$ be the parameters obtained by local ML estimation. We shall show guarantees on $\overline{\mathbf{w}}$ and translate them to guarantees on $F^{\overline{\mathbf{w}}}$.

- We first establish an ϵ-cover of local IC influence functions $f_u^{\boldsymbol{\beta}}$ or $f_u^{\mathbf{w}}$, and obtain a bound on the covering number of this space.

  Lemma 10 (Covering number of local IC influence functions). *Under Assumption 1, the L_∞ covering number of the class of all local IC influence functions $f_u^{\boldsymbol{\beta}}$ for a node u with n_u parameters and radius ϵ is $O((\ln(1/\gamma)/\epsilon)^{n_u})$.*

  Proof. See Section C.7. □

- The covering number result allows us to invoke standard uniform convergence arguments to prove that the log-likelihood of the $\overline{\mathbf{w}}$ can be taken arbitrarily close to the optimal value. But, this does not imply that the estimated parameters are themselves close to the optimal parameters. For this, we show that under Assumption 1, the expected log-likelihood objective is strongly concave in the IC parameters, or equivalently that the negative likelihood is strongly convex, which then implies the desired result.

  Lemma 11 (Guarantees on parameters obtained by local ML estimation). *Let $\mathbf{w}^* \in [0,1]^r$ be the true IC parameters and $\overline{w}_{uv} = 1 - \exp(-\beta_{uv})$ be obtained by local ML estimation. Fix $\epsilon, \delta \in (0,1)$. Under Assumption 1, if $m = \widetilde{O}(nr(\kappa^2(1-\kappa)^4\lambda^2\gamma^2\epsilon^2)^{-1})$, with probability at least $1-\delta$ (over draw of the training sample), $\forall (u,v) \in E,\ \|\mathbf{w}^* - \overline{\mathbf{w}}\|_2^2 \leq \epsilon$.*

  Proof. See Section C.8. □

- Given that that the global IC influence function is Lipschitz (see Lemma 3), the above guarantees translate to the following sample complexity guarantee on the IC influence function.

  Lemma 12 (Translation to global influence function). *Under the statement of Lemma 11, we have with probability at least $1-\delta$, $\left(F_u^{\mathbf{w}^*}(X) - F_u^{\overline{\mathbf{w}}}(X)\right)^2 \leq r\epsilon,\ \forall u \in V,\ X \subseteq V$.*

  Proof. See Section C.9 □

Proposition 5 then directly follows from the above sequence of results.

Proof of Proposition 5. Let $\mathbf{w}^* \in \mathbb{R}_+^r$ be the parameters of the underlying IC model satisfying Assumption 1. Fix $\epsilon, \delta \in (0,1)$, and let $\overline{\mathbf{w}}$ be the parameters obtained from the local maximum likelihood estimation. From Lemma 12, when the number of training examples $m = \widetilde{O}(nr(\kappa^2(1-\kappa)^4\lambda^2\gamma^2\epsilon^2)^{-1})$, we have with probability at least $1-\delta$ (over draw of training sample), for each node $u \in V$, and seed set $X \subseteq V$:

$$\left(F_u^{\mathbf{w}^*}(X) - F_u^{\overline{\mathbf{w}}}(X)\right)^2 \leq r\epsilon,$$

As in the proof of Theorem 2 (see Eq. (4)), we can show from this that with probability at least $1-\delta$,

$$\text{err}^{\text{sq}}\left[F^{\overline{\mathbf{w}}}\right] - \inf_{\mathbf{w} \in \mathbb{R}_+^r} \text{err}^{\text{sq}}\left[F^{\mathbf{w}}\right] \leq \frac{1}{n}\sum_{u=1}^{n} \mathbf{E}_X[r\epsilon] \leq r\epsilon.$$

Absorbing r on the right hand side into the sample complexity bound, gives us the desired result. □

C.7 Proof of Lemma 10

Proof. The local IC influence function for any $Z \subseteq V$ is $f_u^{\beta}(Z) = 1 - \exp\left(-\sum_{v \in N(u) \cap Z} \beta_{uv}\right) = \sigma(\sum_{v \in N(u) \cap Z} \beta_{uv})$, which is a linear function composed with link function $\sigma(s) = 1 - \exp(-s)$. It is well-known that the class of all linear functions with n_u parameters in a bounded range $[a,b]$, can be covered with at most $O(((b-a)/\epsilon)^{n_u})$ L_1-balls of radius ϵ [20]. In our case, each $\boldsymbol{\beta}_{uv} \in [-\ln(1-\lambda), -\ln(\gamma)]$, and the number of L_1-balls to cover the space of all linear functions defined by parameters in this range is at most $O((\ln(1/\gamma)/\epsilon)^n)$ as $1-\lambda < 1$. Let $\mathbf{a}_1, \ldots, \mathbf{a}_R$ be the corresponding centers. Now, since σ is 1-Lipschitz on the positive real-line (follows from $\sigma'(s) = \exp(-s) \leq 1$ for all $s \geq 0$), a set of L_∞-balls of radius ϵ with centers $f_u^{\mathbf{a}_1}, \ldots, f_u^{\mathbf{a}_R}$ would then constitute an ϵ-cover over the class of all local IC influence functions. Thus the L_∞ covering number of this function class for radius ϵ is at most $O\left(\left(\ln(1/\gamma)/\epsilon\right)^{n_u}\right)$. □

C.8 Proof of Lemma 11

As with the partial observation setting, the proof makes use of standard covering number based uniform convergence result for empirical risk minimization (or equivalently for log-likelihood maximization) over a real-valued function class [20]. To apply these standard results, we must ensure the local log-likelihood is bounded and Lipschitz. We do this below.

As before, let $n_u = |N(u)|$ and define for any $Z \subseteq V$, $y \in \{0,1\}$, and $u \in V$, the local log-likelihood g_u for parameters $\boldsymbol{\beta}$ as $g_u(Z, y; \boldsymbol{\beta}) = \big[y \ln\big(f_u^{\boldsymbol{\beta}}(Z)\big) + (1-y)\ln\big(1 - f_u^{\boldsymbol{\beta}}(Z)\big)\big]\mathbf{1}(u \notin Z)$; the indicator term automatically ignore cases where u is already present in seed set Z. Note that for a cascade (X, Y), the local log-likelihood $\mathcal{L}(X, Y_1; \boldsymbol{\beta}) = \frac{1}{n}\sum_{u=1}^{n} g_u(X, \chi_u(Y_1); \boldsymbol{\beta})$. Whenever we refer to a subset $Z \subseteq V$ in the context of a node u, *we shall assume that Z contains a neighbor of u*; cases where this assumption fails can be easily handled, but have been ignored here to make the proof more accessible. Below, we show that g_u is bounded and Lipschitz for any u.

Lemma 13 (Boundedness and Lipschitz continuity of log-likelihood function). *Let $\boldsymbol{\beta}$ be obtained from edge weights that satisfy Assumption 1. Then*

1. $\lambda \leq f_u^{\boldsymbol{\beta}}(Z) \leq 1 - \gamma$.
2. $|g_u(Z, y; \boldsymbol{\beta})| \leq \ln(1/\gamma)$.
3. *$g_u(Z, y; \boldsymbol{\beta})$ is $1/\lambda$-Lipschitz in $\boldsymbol{\beta}_u$ w.r.t. the L_1 norm.*

Proof.

1. Starting with the upper bound, we have $f_u^{\boldsymbol{\beta}}(Z) = 1 - \exp\big(-\sum_{v \in N(u)\cap Z}\beta_{uv}\big) \leq 1 - \exp\big(-\sum_{v\in N(u)}\beta_{uv}\big) \leq 1 - \gamma$ (by Assumption 1). For the lower bound, $f_u^{\boldsymbol{\beta}}(Z) = 1 - \exp\big(-\sum_{v\in N(u)\cap Z}\beta_{uv}\big) \geq 1 - \exp\big(-\beta_{uv'}\big) \geq \lambda$, where u' is some neighbor of u in Z, which we have assumed exists.
2. Using the above result, $|g_u(Z, y; \boldsymbol{\beta})| \leq |y\ln\big(f_u^{\boldsymbol{\beta}}(Z)\big) + (1-y)\ln\big(1 - f_u^{\boldsymbol{\beta}}(Z)\big)| \leq |y\ln(\lambda) + (1-y)\ln(\gamma)| \leq |\ln(\gamma)| = \ln(1/\gamma)$, where we have used $0 < \gamma \leq \lambda < 0.5$.
3. To show Lipschitzness of g_u w.r.t. L_1 norm, we bound the L_∞ norm of its gradient w.r.t. $\boldsymbol{\beta}_u$. Let $\widetilde{\mathbf{Z}} \in \{0,1\}^{n_u}$ be a boolean vector whose entries are $\mathbf{1}(v \in Z)$ for each neighbor $v \in N(u)$. Then $g_u(Z, y; \boldsymbol{\beta}) = \big[y\ln\big(1 - \exp(-\widetilde{\mathbf{Z}}^\top\boldsymbol{\beta}_u)\big) - (1-y)\widetilde{\mathbf{Z}}^\top\boldsymbol{\beta}_u\big]\mathbf{1}(u \notin Z)$ and

$$\begin{aligned}\nabla_{\boldsymbol{\beta}_u}\big[g_u(Z, y; \boldsymbol{\beta})\big] &= \mathbf{1}(u \notin Z)\left[\frac{y\exp(-\widetilde{\mathbf{Z}}^\top\boldsymbol{\beta}_u)}{1 - \exp(-\widetilde{\mathbf{Z}}^\top\boldsymbol{\beta}_u)} - (1-y)\right]\widetilde{\mathbf{Z}} \\ &= \mathbf{1}(u \notin Z)\left[\frac{y\big(1 - f_u^{\boldsymbol{\beta}}(Z)\big)}{f_u^{\boldsymbol{\beta}}(Z)} - (1-y)\right]\widetilde{\mathbf{Z}}.\end{aligned}$$

   Then we have,

$$\big\|\nabla_{\boldsymbol{\beta}_u}\big[g_u(Z, y; \boldsymbol{\beta})\big]\big\|_\infty \leq \frac{1 - f_u^{\boldsymbol{\beta}}(Z))}{f_u^{\boldsymbol{\beta}}(Z)} \max_{v\in N(u)} \mathbf{1}(v \in Z) \leq \frac{1}{\lambda},$$

   where the numerator is upper bounded by 1, and in the denominator we have used the lower bound on $f_u^{\boldsymbol{\beta}}$ shown in the part 1. Hence g_u is $1/\lambda$-Lipschitz in $\boldsymbol{\beta}_u$ w.r.t. the L_1 norm.

□

The above boundedness and Lipschitzness properties of the log-likelihood function will enable us to apply standard covering number arguments to show that the log-likelihood of the estimated parameters can be taken close to the optimal value. This does not however imply that the estimated parameters themselves converge to the optimal parameters; in order to show this, we will need the (negative) likelihood objective to additional be strongly convex. We next show that under Assumption 1, the expected (negative) log-likelihood is strongly convex. In particular, define for any $Z \subseteq V$ and $\eta \in [0,1]$, $\widetilde{g}_u(Z, \eta; \boldsymbol{\beta}) = \big[\eta\ln\big(f_u^{\boldsymbol{\beta}}(Z)\big) + (1-\eta)\ln\big(1 - f_u^{\boldsymbol{\beta}}(Z)\big)\big]\mathbf{1}(u \notin Z)$; again the indicator term automatically ignore cases where u is present in seed set Z. Then we have:

Lemma 14 (Strong convexity of negative expected log-likelihood). *Let μ be a distribution over subsets of nodes in V and $\boldsymbol{\beta}^*$ be underlying IC parameters, both satisfying Assumption 1.Let $\eta : 2^V \to [0,1]$ with $\eta(Z) \geq \lambda$. Then $\mathbf{E}_{Z\sim\mu}\big[-\widetilde{g}_u\big(Z,\eta(Z);\boldsymbol{\beta}\big)\big]$ is strongly convex in $\boldsymbol{\beta}_u$ and the strong convexity parameter is at least $\gamma\lambda\kappa(1-\kappa)^2$.*

Proof. As in the previous lemma, we use $\widetilde{\mathbf{Z}} \in \{0,1\}^{n_u}$ to denote a boolean vector whose entries are $\mathbf{1}(v \in Z)$ for each neighbor $v \in N(u)$. To show strong convexity of $-\mathbf{E}_Z\big[\widetilde{g}_u\big(Z,\eta(Z);\boldsymbol{\beta}\big)\big]$, we compute its Hessian w.r.t. $\boldsymbol{\beta}_u$ and show that the Hessian is well-conditioned or that its smallest Eigen value is bounded above zero. The Hessian is given by:

$$\begin{aligned}
\nabla^2_{\boldsymbol{\beta}_u}\Big[-\mathbf{E}_Z\big[\widetilde{g}_u\big(Z,\eta(Z);\boldsymbol{\beta}\big)\big]\Big] &= \nabla^2_{\boldsymbol{\beta}_u}\Big[-\mathbf{E}_Z\big[\mathbf{1}(u\notin Z)\big[\eta(Z)\ln\big(f_u^{\boldsymbol{\beta}}(Z)\big) \\
&\qquad + (1-\eta(Z))\ln\big(1-f_u^{\boldsymbol{\beta}}(Z)\big)\big]\big]\Big] \\
&= \mathbf{E}_Z\left[\mathbf{1}(u\notin Z)\frac{\eta(Z)\exp(-\widetilde{\mathbf{Z}}^\top\boldsymbol{\beta}_u)}{\big(1-\exp(-\widetilde{\mathbf{Z}}^\top\boldsymbol{\beta}_u)\big)^2}\widetilde{\mathbf{Z}}\widetilde{\mathbf{Z}}^\top\right] \\
&= \mathbf{E}_Z\left[\mathbf{1}(u\notin Z)\frac{\eta(Z)\big(1-f_u^{\boldsymbol{\beta}}(Z)\big)}{f_u^{\boldsymbol{\beta}}(Z)^2}\widetilde{\mathbf{Z}}\widetilde{\mathbf{Z}}^\top\right].
\end{aligned}$$

The following can then be verified to be the smallest Eigen value of the Hessian. Here $\mathbf{x} \in \mathbb{R}^{n_u}$ and we have used for any $Z \subseteq V$ and $u \in V$, $Z_u = \mathbf{1}(u \in Z)$:

$$\begin{aligned}
&\inf_{\mathbf{x}^\top\mathbf{x}=1}\mathbf{E}_Z\left[\frac{\eta(Z)\big(1-f_u^{\boldsymbol{\beta}}(Z)\big)}{f_u^{\boldsymbol{\beta}}(Z)^2}\big(\widetilde{\mathbf{Z}}^\top\mathbf{x}\big)^2\mathbf{1}(u\notin Z)\right] \\
&\geq \gamma\inf_{\mathbf{x}^\top\mathbf{x}=1}\mathbf{E}_Z\big[\eta(Z)\big(\widetilde{\mathbf{Z}}^\top\mathbf{x}\big)^2\mathbf{1}(u\notin Z)\big] \\
&= \gamma(1-\kappa)\inf_{\mathbf{x}^\top\mathbf{x}=1}\mathbf{E}_Z\big[\eta(Z)\big(\widetilde{\mathbf{Z}}^\top\mathbf{x}\big)^2\,\big|\,u\notin Z\big] \\
&= \gamma(1-\kappa)\inf_{\mathbf{x}^\top\mathbf{x}=1}\mathbf{E}_Z\left[\eta(Z)\sum_{v\in N(u)}Z_v x_v^2 + \eta(Z)\sum_{v\in N(u)}\sum_{k\in N(u)}Z_u Z_k x_v x_k\,\middle|\,u\notin Z\right] \\
&= \gamma(1-\kappa)\inf_{\mathbf{x}^\top\mathbf{x}=1}\left[\sum_{v\in N(u)}\mathbf{E}_Z\big[\eta(Z)Z_v\,\big|\,u\notin Z\big]x_v^2 + \sum_{v\in N(u)}\sum_{k\in N(u)}E_Z\big[\eta(Z)Z_vZ_k\,\big|\,u\notin Z\big]x_vx_k\right] \\
&\geq \gamma(1-\kappa)\inf_{\mathbf{x}^\top\mathbf{x}=1}\left[\sum_{v\in N(u)}\lambda\mathbf{P}\big(v\in Z\,\big|\,u\notin Z\big)x_v^2 + \sum_{v\in N(u)}\sum_{k\in N(u)}\lambda\mathbf{P}\big(v\in Z,k\in Z\,\big|\,u\notin Z\big)x_vx_k\right] \\
&= \gamma\lambda(1-\kappa)\inf_{\mathbf{x}^\top\mathbf{x}=1}\left[\kappa\sum_{v\in N(u)}x_v^2 + \kappa^2\sum_{v\in N(u)}\sum_{k\in N(u)}x_vx_k\right] \\
&= \gamma\lambda(1-\kappa)\inf_{\mathbf{x}^\top\mathbf{x}=1}\left[(\kappa-\kappa^2)\sum_{v\in N(u)}x_v^2 + \kappa^2\Big(\sum_{v\in N(u)}x_v\Big)^2\right] \\
&\geq \gamma\lambda(1-\kappa)\big[(\kappa-\kappa^2)(1) + 0\big] \\
&\geq \gamma\lambda\kappa(1-\kappa)^2,
\end{aligned}$$

where the first step follows from $f_u^{\boldsymbol{\beta}}(Z) \leq 1-\gamma < 1$, the second and sixth step follow from Z being drawn from a distribution that satisfies Assumption 1 (i.e. a distribution where each element in V is chosen independently w.p. $\kappa \in (0,1)$), and the fifth step follows from $\eta(Z) \geq \lambda$. Thus the given expected log-likelihood is strongly convex in $\boldsymbol{\beta}_u$ under the given assumptions, and the strong convexity parameter is at least $\gamma\lambda\kappa(1-\kappa)^2$. □

We now make use of both the above results to prove Lemma 11.

Proof of Lemma 11. For the parameters $\overline{\mathbf{w}}$ obtained by from local ML estimation, define log-transformed parameters $\overline{\beta}_{uv} = -\ln(1-\overline{w}_{uv})$ (these parameters satisfy Assumption 1 due to the

way we have framed the optimization problem). Similarly, let $\boldsymbol{\beta}^* \in \mathbb{R}_+^r$ be the transformed version of the underlying model parameters $\mathbf{w}^*$ (satisfying Assumption 1). We shall begin by making use of standard uniform convergence result based on covering numbers [20] and show that the expected log-likelihood of the obtained parameters $\overline{\beta}$ is close to that of the true parameters $\boldsymbol{\beta}^*$; here we will use the covering number result in Lemmas 10 and the boundedness/Lipschitz properties in Lemma 13. We will then exploit the strong convexity of the expected (negative) log-likelihood (shown in Lemma 14) to translate these bounds to guarantees on the parameters themselves.

First, let us write the empirical (local) log-likelihood objective for the first step optimized by the prescribed procedure (shown in Eq. (9)) in terms of g_u.

$$\frac{1}{mn}\sum_{i=1}^{m}\mathcal{L}(X^i, Y_1^i; \boldsymbol{\beta}) \;=\; \frac{1}{n}\sum_{u=1}^{n}\underbrace{\frac{1}{m}\sum_{i=1}^{m} g_u(X^i, \chi_u(Y_1^i); \boldsymbol{\beta})}_{\widehat{G}_u(\boldsymbol{\beta}_u)}.$$

Since each $\widehat{G}_u$ involves a different set of parameters, they can be essentially maximized independently. The maximizer of the above empirical log-likelihood is then simply a concatenation of the maximizers $\overline{\boldsymbol{\beta}}_u \in \mathbb{R}_+^{n_u}$ of each $\widehat{G}_u$. Similarly, one can write down the expected log-likelihood in terms of g_u:

$$\frac{1}{n}\mathbf{E}_{X,Y_1}\big[\mathcal{L}(X, Y_1; \boldsymbol{\beta})\big] \;=\; \frac{1}{n}\sum_{u=1}^{n}\mathbf{E}_{X,Y_1}\big[\underbrace{g_u(X, \chi_u(Y_1); \boldsymbol{\beta})}_{G_u(\boldsymbol{\beta}_u)}\big].$$

Again each G_u can be maximized independently; the optimal parameters $\boldsymbol{\beta}^*$ for the above objective is given by a concatenation of the optimal parameters $\boldsymbol{\beta}_u^*$ for each G_u.

Now from the properties stated in Lemma 13, we have that g_u is bounded by $\ln(1/\gamma)$ and is $1/\lambda$-Lipschitz in $\boldsymbol{\beta}$. We then have based on the covering number result in Lemma 10 for the class of local influence functions, followed by an application of a union bound over all nodes in V, that when $m = O\bigg(n_u \ln(1/\gamma)^2 \dfrac{\ln(\ln(1/\gamma)/\lambda\epsilon) + \ln(n/\delta)}{\epsilon^2}\bigg)$, with probability at least $1-\delta$ (over draw of training sample), for each $u \in V$

$$|G_u(\overline{\boldsymbol{\beta}}_u) \;-\; \widehat{G}_u(\overline{\boldsymbol{\beta}}_u)| \;\leq\; \epsilon,$$

where $n_u = |N(u)|$. Equivalently, when $m = O\bigg(\ln(1/\gamma)^2 \dfrac{\ln(\ln(1/\gamma)/\lambda\epsilon\sqrt{n_u}) + \ln(n/\delta)}{\epsilon^2}\bigg)$, with probability at least $1-\delta$ (over draw of training sample), for each $u \in V$

$$|G_u(\overline{\boldsymbol{\beta}}_u) \;-\; \widehat{G}_u(\overline{\boldsymbol{\beta}}_u)| \;\leq\; \epsilon\sqrt{n_u},$$

which will further give us using straight-forward algebra (see proof of Theorem 2) that with probability at least $1-\delta$,

$$G_u(\boldsymbol{\beta}_u^*) \;-\; G_u(\overline{\boldsymbol{\beta}}_u) \;\leq\; \epsilon\sqrt{n_u}. \tag{10}$$

Thus, $\boldsymbol{\beta}_u^*$ and $\overline{\boldsymbol{\beta}}$ are close in terms of their likelihood value. The rest of the proof involves using strong convexity of the (negative) expected log-likelihood (in Lemma 14) to show that similar guarantees hold for the parameters themselves. In particular, we shall use the fact that if the negative of a function $h : \mathbb{R}^d \to \mathbb{R}$ is q-strongly convex with $\mathbf{z}^* = \text{argmax}_{\mathbf{z}\in\mathbb{R}^d} h(\mathbf{z})$ the following is true for any $\mathbf{z} \in \mathbb{R}^d$: $h(\mathbf{z}^*) - h(\mathbf{z}) \geq \frac{q}{2}\|\mathbf{z} - \mathbf{z}^*\|_2^2$.

Expanding the left-hand side of the above inequality,

$$\begin{aligned}
G_u(\boldsymbol{\beta}_u^*) \;-\; \widehat{G}_u(\overline{\boldsymbol{\beta}}_u) \;&=\; \mathbf{E}_{X,Y_1}\big[g_u(X, \chi_u(Y_1); \boldsymbol{\beta}^*) \;-\; g_u(X, \chi_u(Y_1); \overline{\boldsymbol{\beta}})\big] \\
&=\; \mathbf{E}_{X,Y_1}\big[\mathbf{1}(u \notin X)\big[\chi_u(Y_1)\ln(f_u^{\boldsymbol{\beta}^*}(X)) \;+\; (1-\chi_u(Y_1))\ln(1-f_u^{\boldsymbol{\beta}^*}(X)) \\
&\qquad\qquad -\; \chi_u(Y_1)\ln(f_u^{\overline{\boldsymbol{\beta}}}(X)) \;-\; (1-\chi_u(Y_1))\ln(1-f_u^{\overline{\boldsymbol{\beta}}}(X))\big]\big] \\
&=\; \mathbf{E}_X\big[\mathbf{1}(u \notin X)\big[f_u^{\boldsymbol{\beta}^*}(X)\ln(f_u^{\boldsymbol{\beta}^*}(X)) \;+\; (1-f_u^{\boldsymbol{\beta}^*}(X))\ln(1-f_u^{\boldsymbol{\beta}^*}(X)) \\
&\qquad\qquad -\; f_u^{\boldsymbol{\beta}^*}(X)\ln(f_u^{\overline{\boldsymbol{\beta}}}(X)) \;-\; (1-f_u^{\boldsymbol{\beta}^*}(X))\ln(1-f_u^{\overline{\boldsymbol{\beta}}}(X))\big]\big] f
\end{aligned}$$

$$
\begin{aligned}
&= \mathbf{E}_X\left[\widetilde{g}_u\big(X, f_u^{\boldsymbol{\beta}^*}; \boldsymbol{\beta}^*\big) - \widetilde{g}_u\big(X, f_u^{\boldsymbol{\beta}^*}; \overline{\boldsymbol{\beta}}\big)\right] \\
&\geq \frac{\lambda\gamma\kappa(1-\kappa)^2}{2}\|\boldsymbol{\beta}_u^* - \overline{\boldsymbol{\beta}}_u\|_2^2,
\end{aligned}
$$

where the second equality follows from $E[\chi_u(Y_1)|X] = f_u^{\boldsymbol{\beta}^*}(X)$, and the fourth step follows from the strong convexity result in Lemma 13 with $\eta = f_u^{\boldsymbol{\beta}^*}$. Substituting this back in Eq. (10), we have with probability at least $1-\delta$ (over draw of the training sample), for each u

$$
\|\boldsymbol{\beta}_u^* - \overline{\boldsymbol{\beta}}_u\|_2^2 \leq \frac{2\epsilon\sqrt{n_u}}{\lambda\gamma\kappa(1-\kappa)^2}.
$$

In other words, if $m = O\left(\ln(1/\gamma)^2 \frac{\ln(\ln(1/\gamma)/\lambda\epsilon\sqrt{n}) + \ln(n/\delta)}{\kappa^2(1-\kappa)^4\lambda^2\gamma^2\epsilon^2}\right)$, then w.p. at least $1-\delta$,

$$
\|\boldsymbol{\beta}_u^* - \overline{\boldsymbol{\beta}}_u\|_2^2 \leq \epsilon\sqrt{n_u}.
$$

Summing this over all $u \in V$,

$$
\|\boldsymbol{\beta}^* - \overline{\boldsymbol{\beta}}\|_2^2 \leq \epsilon\sum_{u=1}^{n}\sqrt{n_u} \leq \epsilon\sqrt{n\sum_{u=1}^{n} n_u} = \epsilon\sqrt{nr},
$$

where the second last step follows from Jensen's inequality given that the square-root is a concave function. We thus have guarantees on the log-transformed parameters. It is straight-forward to show that the same guarantees also hold in the original parameter space, i.e. w.p. at least $1-\delta$, $\|\mathbf{w}^* - \overline{\mathbf{w}}\|_2^2 \leq \epsilon\sqrt{nr}$. Absorbing the term $\sqrt{nr}$ into the sample complexity bound completes the proof. □

C.9 Proof of Lemma 12

Proof. Recall from the Lemma 3 that $F_u^{\overline{\mathbf{w}}}$ is 1-Lipschitz in $\mathbf{w}$ w.r.t. the ℓ_1 norm. So if $\|w^* - \overline{w}\|_2^2 \leq \epsilon$ for all $(u,v) \in E$, then $\left|F_u^{\mathbf{w}^*}(X) - F_u^{\overline{\mathbf{w}}}(X)\right|^2 \leq \|\mathbf{w}^* - \overline{\mathbf{w}}\|_1^2 \leq r\|\mathbf{w}^* - \overline{\mathbf{w}}\|_2^2 \leq r\epsilon$. This, together with the result in Lemma 11, leads to the desired guarantee on $F_u^{\overline{\mathbf{w}}}$. □