Maximizing Product Adoption in Social Networks

Smriti Bhagat   Amit Goyal   Laks V. S. Lakshmanan

University of British Columbia
Vancouver, BC, Canada
{smritib,goyal,laks}@cs.ubc.ca

ABSTRACT

One of the key objectives of viral marketing is to identify a small set of users in a social network, who when convinced to adopt a product will influence others in the network leading to a large number of adoptions in an expected sense. The seminal work of Kempe et al. [13] approaches this as the problem of influence maximization. This and other previous papers tacitly assume that a user who is influenced (or, informed) about a product necessarily adopts the product and encourages her friends to adopt it. However, an influenced user may not adopt the product herself, and yet form an opinion based on the experiences of her friends, and share this opinion with others. Furthermore, a user who adopts the product may not like it and hence not encourage her friends to adopt it to the same extent as another user who adopted and liked the product. This is independent of the extent to which those friends are influenced by her. Previous works do not account for these phenomena.

We argue that it is important to distinguish product adoption from influence. We propose a model that factors in a user's experience (or projected experience) with a product. We adapt the classical Linear Threshold (LT) propagation model by defining an objective function that explicitly captures product adoption, as opposed to influence. We show that under our model, adoption maximization is NP-hard and the objective function is monotone and submodular, thus admitting an approximation algorithm. We perform experiments on three real popular social networks and show that our model is able to distinguish between influence and adoption, and predict product adoption much more accurately than the classical LT model.

Categories and Subject Descriptors H.2.8 [Database Management]: Database Applications - *Data Mining*
General Terms: Algorithms, Theory, Experimentation.
Keywords: Product adoption, Influence, Viral marketing.

1. INTRODUCTION

One of the fundamental problems in viral marketing is to identify a small set of individuals in a social network, who when convinced to adopt a product will influence others in the network through a word-of-mouth effect, leading to a large number of adoptions in an expected sense. Previous works such as [13] approached this as the problem of *influence maximization* where, given a social network represented by a directed graph with users as nodes, edges corresponding to social ties, and edge weights capturing influence probabilities, the goal is to find a *seed set* of k users such that by targeting these, the *expected influence spread* (defined as the expected number of influenced users) is maximized. Here, the expected influence spread of a seed set depends on the influence diffusion process which is captured by a model for influence propagation. While several propagation models exist, two classical models, namely Linear Threshold (LT) and Independent Cascade (IC), have been widely studied in the literature. In this work, we focus on the LT model. In the LT model, a user is either in an inactive or an active state. An active user influences each of its inactive neighbors. The activation of an inactive user depends on the influence probabilities associated with its active neighbors. Each user v picks an activation threshold θ_v uniformly at random from $[0, 1]$. At any time step, if the sum of incoming weights from its active neighbors exceeds the threshold, v becomes active. The process continues until no more activations are possible.

In the bulk of the influence maximization literature in data mining, it is assumed that once a user is active (i.e., is influenced), she will automatically and unconditionally adopt the product. In this sense, influence spread is viewed as equivalent to adoption spread (expected number of users who adopt the product). Clearly, product (or technology/innovation) adoption is the main goal in viral marketing, and influence spread is essentially used as a "proxy" for adoption. We argue that it is important to distinguish between influence and adoption, an idea well established in other domains like Sociology and Marketing. Bohlen et al. [2] in 1957 proposed five stages of product adoption in which the adoption stage is considered different than the awareness stage. Kalish [12] characterized the adoption of a new product as consisting of two steps – awareness and adoption – and argued that the awareness information spreads in an epidemic-like manner while the actual adoption depends on other factors such as price and individual's valuation of the product. Given these studies in other traditional domains, the current influence maximization work in the data mining community lags behind in modeling such phenomena. Our work is a step towards building a more realistic model of product adoption.

More concretely, previous literature (in data mining) on influence maximization makes three assumptions. First, once a user is influenced, she will immediately adopt the product. Second, it is assumed that once a user adopts a product, she encourages her friends to adopt it as well, irrespective of whether or how much she likes the product. In other words, adoption of a product implies that the user likes it regardless of her experience with it. Third, only after a user adopts a product, she shares her opinion on the product and attempts to influence her neighbors. That is, non-adopters either don't have any opinions on the product or such opinions are not visible to others. These assumptions may not hold in general as illustrated by the following examples.

Example 1. Bob buys an Amazon Kindle and dislikes the product. He posts his experience on his blog and describes the missing features. Kali, a friend of Bob, takes Bob's opinions on technology products seriously. She learns about the Kindle from the post and decides *not* to buy it. Furthermore, she blogs about the product not meeting her expectations and in turn influences the decisions of her friends. □

Example 2. Bob watches the movie "The Ring" and likes it. He tells his friend Kali that the movie is great. Kali doesn't get a chance to watch the movie, but tells her friend Charles about the movie. Charles gets influenced and watches it. □

In Example 1, Bob doesn't like the product and shares his opinion with his friends. Thus, opinions can emerge from a user experience with a product and can propagate from an adopter. Specifically, if a user does not like the product, their endorsement of the product is unlikely to be strong, a fact that may be noticed by their neighbors. Even when a user is strongly influenced by her neighbor, if the endorsement is weak, the user is less likely to adopt the product. The propagation of information to a user does not always lead to product adoption. There may be many factors that affect a user's adoption decision, including the user's interests, budget and time. Indeed, opinions can propagate from a non-adopter who is active (e.g., Kali in Example 2) and can promote adoption by others. In this example Kali acts as an "information bridge" or a *tattler*, who talks about the product without actually adopting it. How does the presence of such non-adopters who act as information bridges affect the overall adoption? Does the product adoption depend on the opinions, in addition to social influence? These are some of the questions this paper addresses.

In particular, we argue that the classical propagation models that make the three assumptions above may predict a product adoption quite different from reality, a point we will establish empirically using three real data sets. This motivates re-examining these assumptions and allowing for the following facts: (i) adopters' opinions may strongly affect the degree to which they influence their neighbors' adoption and (ii) non-adopters acting as an information bridge can contribute their own opinions, which too can affect influence propagation as well as adoption. We use user ratings of products as an abstraction of their opinions. Ratings are either provided by the user or can be predicted using collaborative filtering [14]. Therefore, the extent to which a user is influenced by her neighbors (to become active) is a function, not only of the influence probabilities but of the ratings of the neighbors as well. In this paper, we study an interesting variation of the classical influence maximization problem with an aim to *maximize the number of product adoptions* and make the following contributions.

- We study the novel problem of maximizing product adoption, a more natural goal of applications like viral marketing. We present an intuitive model called LT-C, for *LT with Colors*, that captures these intuitions. We show that the problem of product adoption maximization is NP-hard but the objective function, i.e., the expected number of product adoptions, is monotone and submodular; thus the greedy algorithm provides a $(1 - 1/e - \epsilon)$-approximation to the optimal solution. We also propose methods to learn parameters of our model (Section 3).
- We study two types of networks – movies networks (Section 4.1) and music networks (Section 4.2) using three real world datasets. We found that in almost all instances, the classical LT model significantly over-predicts the spread. Moreover, it predicts the same spread for all the products. We demonstrate that our LT-C model, by incorporating ratings, significantly improves the accuracy of these predictions.
- In all the three datasets, we found that there is a positive correlation between the number of initiators (users who first express opinions among their friend circles) and the final spread, suggesting that the network structure plays a significant role in the spread. We demonstrate that the choice of seed nodes is critical and can make a significant difference to the spread achieved. We also found that largely, the seed sets are different for different products, though some influential users are picked up consistently as seeds irrespective of the product being marketed (Section 4).

We review related work in Section 2 and present conclusions in Section 5.

2. BACKGROUND AND RELATED WORK

Background. In 2001, Domingos et al. [6] introduced the the problem of identifying influential users to the data mining community. Kempe et al. [13] formalized it as the problem of influence maximization as described above. In particular, their work focuses on two propagation models – Linear Threshold (LT) Model and Independent Cascade (IC) Model. The influence maximization problem is NP-hard under both the LT and IC models [13], and the expected influence spread function is monotone and submodular. Formally, a set function f is submodular iff $f(S+\{x\})-f(S) \geq f(T+\{x\})-f(T)$ whenever $S \subseteq T \subset V$ and $x \in V \setminus T$, where V is the set of all users. Exploiting these properties, Kempe et al. [13] showed that a simple greedy algorithm, which iteratively adds a user with the maximum marginal gain to the current seed set, guarantees a $(1 - 1/e - \epsilon)$-approximation to the optimal solution, for any $\epsilon > 0$. Our work adapts the framework of the LT model for product adoption in the presence of non-adopters who may promote or inhibit the adoption by sharing their opinions.

Related Work. One related problem is that of revenue maximization, studied by Hartline et al. [10]. They offer a family of strategies based on an *influence and exploit* paradigm that work as follows. In the first step, called the *influence* step, the seller gives the item to a chosen set of customers for free. Then in the *exploit* step, the seller visits the remaining buyers in a random sequence and offers each buyer a price that is expected to maximize her revenue. Their model allows the seller to bypass the network structure and visit arbitrary buyers at any time. While revenue maximization, in principle, is closely related to adoption maximization, and arguably should subsume the latter, there are the following key differences with our work. We explicitly model the following phenomena observed in real data sets: first, users may be activated but not necessarily adopt/buy a product and instead may tattle about it; second, opinions shared by tattlers may promote adoption by other users; third, sometimes, tattlers may choose to inhibit adoption by other users by sharing the poor ratings they gave for the product.

Another closely related problem is that influence maximization in the presence of negative opinions (see Chen et al. [3]). Their model, called IC-N, builds on the classical IC model by subdividing the active state into two substates, positive and negative, while preserving the properties of monotonicity and submodularity. They incorporate the spread of negative opinions with a parameter q that models the quality of the product. The model assumes that the parameter q is the same across all users, an assumption that is not always realistic. Unfortunately, this assumption is essential to their framework, since if the parameter q is allowed to be different for different users (indicating users like the product to differing degrees), their objective function is no longer monotone and submodular, thus, the greedy algorithm fails to provide any approximation guarantee. In our work, we argue that influence propagation depends on the extent to which a user likes the product, in addition to the influence weights among users. Furthermore, we claim that in a network, there exist information bridges, or *tattlers* who propagate the influence without adopting the product themselves. We show that these tattlers are indeed present in real data sets and their presence makes a significant difference to product adoption. A key feature of our model is that not only do we allow the probability of liking a product to be different for different users, we do so while retaining the monotonicity and submodularity of the objective function.

Tang et al. [18] recognize that influence varies with the domain/topic of an item. They present a scalable approach based on MapReduce for learning topic-specific social influence weights from a given social graph and topic distribution. By contrast, in our work, we recognize the reality that user tastes and opinions can be not just topic specific but even item specific and take that into account in our model.

In addition to social influence, *homophily* (or *selection*) has been shown to be a possible cause of users' actions. While some (e.g., [1]) focus on distinguishing the effects of influence and homophily, others (e.g., [5], [15]) showed the feedback effects that both factors have on each other. In particular, Crandall et al. [5] argued that both influence and homophily can be useful in predicting users' future behavior. In our work, we are interested in building a model that predicts the users' actions, in particular, users' adoption of products, without necessarily distinguishing the causes, and use it to develop algorithms that maximize the number of adoptions. Following [12], we argue that while information (or influence) spreads in an epidemic-like manner, actual adoption depends on various other factors. We separate the state of adoption from the state of being influenced (or active) in the LT model. Clearly, for the model to work, it is important that the underlying network structure represent the flow of information, irrespective of whether it corresponds to an explicit social graph (where the links are formed explicitly by users, e.g., friendship links in Facebook), an implicit social graph (where the information flows among users indirectly, e.g., as in recommendation engines like Movielens, Amazon etc.), or a combination of the two.

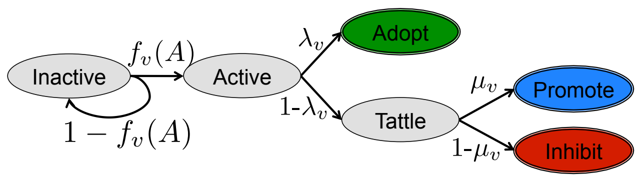


Figure 1: LT-C model with colored end states: adopt–green, promote–blue, inhibit–red

A recent work by Goyal et al. [8] also focuses on predicting the users' actions, while offering an alternative "data-based" approach that bypasses the propagation models to estimate the spread. The same authors [7] study the problem of learning influence probabilities based on the past propagation traces, paying special attention to the temporal nature of influence probabilities. In other work, Chen et al. [4] have proposed an efficient heuristic for influence maximization under the LT model, which is recently improved by [9].

3. PROPOSED FRAMEWORK

Let $G = (V, E, W)$ be a weighted, directed graph with nodes (users) V and edges (social ties) E, where the influence weights are captured by the function $W : E \to [0, 1]$. The weight $w_{u,v}$ associated with an edge $(u, v) \in E$ represents the probability of user u influencing user v, such that the sum of incoming edge weights at any node is no greater than 1. That is, $\sum_{u \in N^{in}(v)} w_{u,v} \leq 1$, where $N^{in}(v)$ is the set of in-neighbors of user v. In addition, we assume that a matrix R of Users $\times$ Products is given where each entry $r_{u,i}$ denotes the rating given by user u to the product i. We assume whenever ratings are missing, they are predicted using collaborative filtering. The higher the rating, the more positive the opinion.

Intuitively, a user's decision to adopt a product should not only depend on the influence of neighbors, but also on what they think about the product, as captured by the ratings. Next, we describe our model that we call LT-C model, for *Linear Threshold with Colors.*

3.1 LT-C Model

Figure 1 gives an overview of our model in the form of a state diagram. An INACTIVE node v has not yet formed an opinion of the new product and is open to being influenced by neighbors. A node is ACTIVE if it is influenced by its neighbors. Let A be the set of ACTIVE in-neighbors of node v and $f_v(A)$ denote the activation function of v, that is, the total influence on v from nodes in A s.t. $0 \leq f_v(A) \leq 1$. Values of $f_v(A)$ close to 1 can be thought of as a collective "recommendation" of ACTIVE neighbors of u strongly favoring the product. On the hand, values of $f_v(A)$ that are close to 0 represent that neighbors of u collectively disapprove the product. As with the LT model, an INACTIVE node v picks an *activation threshold* θ_v uniformly at random from $[0, 1]$ and if $f_v(A) \geq \theta_v$ then v becomes ACTIVE, else v remains INACTIVE. We instantiate the activation function as follows,

$$f_v(A) = \frac{\sum_{u \in A} w_{u,v}(r_{u,i} - r_{\min})}{r_{\max} - r_{\min}} \qquad (1)$$

where $r_{\max}$ and $r_{\min}$ represent the maximum and minimum ratings in the system, respectively. Intuitively, this definition corresponds to treating, for each edge (u, v) in the graph, the effective influence weight of u on v as $w_{u,v}(r_{u,i} - r_{\min})/(r_{\max} - r_{\min})$, where i is the product being marketed.

Once a node is in the ACTIVE state, it has enough information about the product and has formed an opinion. At this stage, the node can choose to *adopt* the product and rate it, or decide to share its opinion without adopting the product i.e., *tattle*. With some probability λ_v, an activated node v enters the ADOPT state (or adopts) and is colored *green*, and with probability $1 - \lambda_v$, node v enters the TATTLE state (or tattles). Intuitively, the parameter λ_v models the likelihood of a node adopting the product after it has been activated by its neighbors. There are several real world factors that can be modeled into λ_v, such as a user's budget, interests and purchase history. Note that the parameter λ_v is specific to each node v. If the node decides to ADOPT, it provides a rating $r_{v,i}$ for the product i being marketed. In Section 3.4, we propose methods to learn these parameters.

The TATTLE state is an interesting one. It represents a typical gossipmonger who has not experienced the product but is expressing an *acquired* opinion. Such a node is likely to be biased one way or the other. We say a tattler (node in the TATTLE state) can either enter the PROMOTE state (color *blue*) or the INHIBIT state (color *red*) depending on its disposition parameterized by μ_v, specific to v. In other words, v enters the PROMOTE state with probability μ_v and INHIBIT stat with probability $1 - \mu_v$. For instance, consider a user who loves (or hates) the iPhone and owns one already. When the latest model is launched, even though the user is not ready to buy it, the user has some strongly biased opinion of the product which the user shares with his friends. We model this bias in opinion by representing it with a constant rating of $r_{\max}$ for a user in the PROMOTE state and $r_{\min}$ for a user in the INHIBIT state. A node in the PROMOTE or INHIBIT state propagates information but does not directly contribute to the revenue. The node's actions however may stimulate or inhibit adoption by its neighbors, thus proving to be an important information bridge in the network. Finally, we should mention that of the states in Figure 1, only the INACTIVE, ADOPT, PROMOTE and INHIBIT states may be observable in a real data set. The intermediate states are used for modeling purposes only.

The dynamics followed by the process of opinion propagation at discrete time steps are as follows. At time $t = 0$, a set S of k nodes is chosen as the *seed set* and are said to be active, and each seed node can choose any of the three states ADOPT, PROMOTE or INHIBIT. At $t > 0$, each node u that activated (ADOPT, PROMOTE or INHIBIT state) at time t, contributes to the activation of its neighbors v where $(u, v) \in E$. Let A_t denote the neighbors of v that are ACTIVE at time t, then the total influence on node v at time t is $f_v(A_t)$. If the ACTIVE neighbors push $f_v(A_t)$ over the threshold θ_v, then v becomes ACTIVE, else it remains INACTIVE. Once in ACTIVE state, the node follows the path to enter one of ADOPT, PROMOTE or INHIBIT states and stays in that state with corresponding probabilities. Also, each ACTIVE node provides a rating of the product that is factored into the activation function. Thus, once a node becomes active, it enters one of these states. A_{t+1} is the set of all nodes that become active as a result of the influence from their neighbors in A_t. The process of node activation stops if at some time step no new nodes can be activated.

3.2 Maximizing Product Adoption

We define the *expected spread* (or *spread* for short) of the seed set S as the expected number of ADOPT nodes in the network at the end of the propagation and denote it $\sigma(S)$. We sometimes refer to $\sigma(S)$ as the *coverage* of set S.

Problem 1 (Maximizing Product Adoption)**.** *Given a social graph $G = (V, E, B)$, a parameter k, and a ratings matrix R and parameters $\lambda_v, \mu_v, \forall v \in V$, the problem of maximizing product adoption is to find a seed set S of k nodes such that by activating these nodes, the expected spread under LT-C model is maximized.*

Not surprisingly, the problem is NP-hard. However, as we will show, the spread function is monotone and submodular, so a simple greedy algorithm leads to a $(1 - 1/e - \epsilon)$-approximation to the optimum, for any $\epsilon > 0$.

Theorem 1. *Problem 1 is NP-hard.*

PROOF. Consider the restricted class of instances of the problem where $\forall v \in V, \lambda_v = 1$ and $r_{v,i} = r_{max}$. The problem of maximizing product adoption over this class of instances is equivalent to the classical problem of influence maximization under the LT model, which is known to be NP-hard [13]. The theorem follows. ☐

Theorem 2. *The spread function $\sigma(\cdot)$ under LT-C model is monotone and submodular.*

PROOF. It is straightforward to show that $\sigma(\cdot)$ is monotone. To establish submodularity, our proof outline follows that in [13]. The challenge in our case is two-fold: (1) There are four possible observable states of nodes – INACTIVE (gray), ADOPT (green), PROMOTE (blue), and INHIBIT (red) in place of just two; (2) The activation of a node depends on the state of its in-neighbors. The reason is that the influence of a node u on its neighbor v is a combination of the weight $w_{u,v}$ and the rating of u. This rating depends on u's state: it's r_{min} in the red state, r_{max} in the blue state, and a value $r_{u,i}$ obtained from R in the green state. Thus, u's state affects the extent of u's influence on its neighbors.

Borrowing the idea of a live edge model from [13], we define a timed version of the live edge selection process for the purpose of this proof. Start by activating the seed nodes S in G and color them as follows: for any $u \in S$, color u green w.p. λ_u, blue w.p. $(1-\lambda_u)\mu_u$, and red w.p. $(1-\lambda_u)(1-\mu_u)$. At any time $t > 0$, allow each uncolored out-neighbor v of a colored node to choose at most one of its in-neighbors z w.p. $p_{z,v}$ and no in-neighbor w.p. $\sum_{y \in N^{in}(v)} w_{y,v} = 1$. If the chosen in-neighbor is colored, then color v green w.p. λ_v, blue w.p. $(1 - \lambda_v)\mu_v$, and red w.p. $(1 - \lambda_v)(1 - \mu_v)$. Otherwise, don't record the choice. That is, if the chosen in-neighbor is not colored, we don't record the choice of the in-neighbor, but constrain future choices of in-neighbors by v, if any, to be outside the set of nodes colored by time $t - 1$. Term each chosen edge i.e., edge to the chosen colored neighbor as a *live edge*. Other edges are blocked/dead. A *live path* is a path made of only live edges. Stop the process when there is no change. This process produces a distribution over possible worlds. Let X be one such possible world. We will show that $\sigma(S) = \sum_X Pr[X]\sigma_X(S)$, where $\sigma_X(S)$ is the number of green nodes reachable from the seed nodes S via live paths in the possible world X. It is easy to see that $\sigma_X(.)$ is monotone and submodular, from which the theorem follows, since a non-negative linear combination of submodular functions is submodular. To show that $\sigma(S) = \sum_X Pr[X]\sigma_X(S)$, we will show that the probability distributions of sets of green/blue/red nodes obtained by running the LT-C model are identical to those obtained from the above process.

We step through the diffusion process according to the LT-C model and determine the probability with which any node v turns green, blue or red. Let each node v pick its threshold θ_v uniformly at random from $[0, 1]$. Let S_t be the set of ACTIVE (colored) nodes at time $t = 0, 1, 2, \ldots$, with $S_0 = S$. The probability with which an uncolored node v will be colored at time $t+1$ is the likelihood that a neighbor u that got colored at time t pushes the total influence on v over the threshold θ_v. We denote this probability of v getting colored at time $t+1$ as ψ_v^{t+1}. Therefore,

$$Pr[v \text{ got colored at time } t+1] = \psi_v^{t+1} = \frac{\sum_{u \in S_t \setminus S_{t-1}} p_{u,v}}{1 - \sum_{u \in S_{t-1}} p_{u,v}}$$

Once a node v is ACTIVE, its color depends solely on the parameters λ_v and μ_v and so we have:

$$Pr[v \text{ turns green at } t+1] = \lambda_v \cdot \psi_v^{t+1}$$
$$Pr[v \text{ turns blue at } t+1] = (1 - \lambda_v) \cdot \mu_v \cdot \psi_v^{t+1}$$
$$Pr[v \text{ turns red at } t+1] = (1 - \lambda_v) \cdot (1 - \mu_v) \cdot \psi_v^{t+1}$$

Next, we consider the timed version of the live edge selection process defined above and iterate through the corresponding process. Let $S'_0 = S$ be the seed set. Let S'_t, $t \geq 0$, denote the set of nodes that is colored at time t according to this process. Let $v \notin S$ be any node. The probability that it got colored at time $t+1$, denoted by ϕ_v^{t+1}, is the probability with which its chosen in-neighbor u got colored at time t, given that v was not colored before time $t + 1$. More precisely,

$$Pr[v \text{ got colored at time } t+1] = \phi_v^{t+1} = \frac{\sum_{u \in S'_t \setminus S'_{t-1}} p_{u,v}}{1 - \sum_{u \in S'_{t-1}} p_{u,v}}$$

Applying induction on time, it is easy to see that the distributions of ACTIVE (i.e., colored) sets of nodes obtained from running the LT-C model, $S_t, t \geq 0$, and the sets of nodes reachable from the seed nodes S by live paths, $S'_t, t \geq 0$, are identical. Given the equivalence between the distributions over S_t and S'_t, it follows that $\phi_v^{t+1} = \psi_v^{t+1}$. Since the color acquired by an ACTIVE node solely depends on the parameters λ_v, μ_v, the distributions of nodes colored green, blue, and red under the LT-C model and under the timed live edge selection process are identical. This was to be shown. $\square$

3.3 Choosing Optimal Seed Set

While product adption maximization is NP-hard, as shown by Theorem 2, the spread function $\sigma(S)$ under the LT-C model is monotone and submodular. Thus, we can employ a simple greedy algorithm which repeatedly picks a node with the maximum marginal gain and adds it to the seed set, until the budget k is reached. Furthermore, since the LT model is a special case of the LT-C model (corresponding to $\lambda_v = 1, \forall v$ and $r_{v,i} = r_{\max}, \forall v$), the #P-hardness [4] of computing the spread of a given seed set carries over to our setting. To mitigate this, we employ Monte Carlo simulation for estimating the spread. Finally, we adapt the CELF algorithm of Leskovec et al. [16]. The idea behind CELF is that the marginal gain of a node cannot increase in subsequent iterations (due to submodularity) and thus the spread of seed sets is computed in a lazy forward manner, speeding up the greedy algorithm considerably. Our implementation is based on these ideas. The adaptation of CELF to LT-C model is trivial and we omit the details. It is well known that this approach yields a $(1 - 1/e - \epsilon)$-approximation to the optimum, for any $\epsilon > 0$.

3.4 Learning Model Parameters

Edge Weights. As in [7], we learn influence weights from the past behavior of users. For example, consider a log of ratings in which each tuple is of the form $\langle u, i, t, r\rangle$, saying user u rated an item i at time t with rating r. There are several cases where such information is readily available in the real world. For example, consider a movie recommender system (see Section 4 for the complete case study), then an action can be considered as "user rating a movie". Seeing friends' ratings, one may be influenced to watch the movie and in turn, rate the movie at a later timestamp. Hence, the influence weight on an edge (u, v) is learnt as the fraction of times user v rated an item after u had done so, and normalized over all neighbors x of v such that $\sum_{x \in N^{in}(v)} w_{x,v} = 1$.

Ratings Matrix. We compute the ratings matrix R using collaborative filtering. Several sophisticated collaborative filtering methods have been proposed by the recommender systems community for predicting the rating of a user for a given item or product. Matrix factorization [14] is one such popular method. The input to matrix factorization is a very large sparse matrix of user ratings with a large number of missing values corresponding to users who have not rated a product. The main task is to predict these missing ratings. An assumption made is that some small number of ratings are available for each product. In the context of our problem, we maintain the assumption that the new product which we want to push in the market has been adopted by some small number of early adopters, who have assigned ratings to the product. This is commonly seen when new products are launched, for instance, technology bloggers get a sneak-peak of new products at tech-media events and they blog about those products. Often, linux operating system and even Google services are first released as beta versions to selected users for product analysis and to get initial feedback before launching to the public. In this paper, we make use of product ratings by users (either provided by users or predicted by recommender algorithms) as a way of modeling the user opinion of a product.

Node Parameters λ and μ. Recall that an active user v enters the ADOPT state with probability λ_v and it enters the TATTLE state with probability $1 - \lambda_v$. Typically, companies such as Netflix and Amazon have user logs which can be used to determine if a user reviewed the product without adopting it. Next to each review on Amazon, where the user bought the product being reviewed, there is a label that states "Amazon Verified Purchase" representing an adoption. The reviews without this label can be attributed to tattle nodes (states promote or inhibit). Given the user rating log, we learn λ_v as maximum likelihood estimate (MLE) which is the fraction of times a user provided an explicit rating for a product over the times the user provided any opinion including a comment, review and numeric rating.

The parameter μ_v models the inherent bias of user v. As with λ_v, we can rely on the rating log to compute this parameter. As an example, in the movie rating social network Flixster, there are special non-numeric ratings "want to see it" and "not interested" that map closely to the PROMOTE and INHIBIT states in our model respectively. Again, we use maximum likelihood estimate (MLE) to compute μ_v, that is, the fraction of times a user gave the rating "want to see it" over the number of times any such special rating was given by that user. We show the effectiveness of this choice by the means of extensive experiments in Section 4.

4. MODEL EVALUATION

We perform empirical analysis to study the adoption of two types of "products" – movies and artists. We analyze influence spread and actual adoption (viewing) of movies on two datasets, one from a social network for movies, Flixster[1], and the other from a movie recommender system, Movielens[2], presented in Section 4.1. Further, we analyze a music social network, Last.fm[3], and study the adoption (listening) of songs and artists, presented in Section 4.2. The table in Figure 2(a) presents the basic statistics of all these datasets.

We compare the following models in our evaluation.

Classical LT. Linear Threshold model proposed in [13].

LT-C. Our proposed model.

LT Ratings. Our proposed model without TATTLE nodes. That is, all the nodes who are influenced adopt the product, and $\lambda_v = 1, \forall v \in V$. This is equivalent to modifying the activation function of the classical LT model to include ratings as defined in Eq. (1).

LT Tattle. Our proposed model without any ratings. That is, all the nodes in ADOPT state are assumed to rate the item as $r_{\max}$, as do those in PROMOTE state, while users in INHIBIT state rate $r_{\min}$.

We tested these variations of LT-C in order to understand the relative contribution of the different components to the overall accuracy of predicting the expected adoption spread. In all the experiments, we run 10K Monte Carlo simulations to estimate the coverage for $k = 50$ seeds. The data was divided into test and training sets randomly such that all the ratings of a movie fall in exactly one of training or test sets. All validation experiments are run on the test set.

4.1 Adoption of Movies

Datasets. Flixster is a social network for movies which enables users to share their opinion on movies with friends by rating and reviewing movies. This dataset, collected by Jamali et al. [11], in its raw form has 1M users, 14M (undirected) friendship relations among users, and 8.2M ratings that range from half a star (rating 0.5) to five stars (rating 5). Since running Monte Carlo (MC) simulations is very expensive (may take several hours even for 10K nodes), a graph of that size is difficult to handle, given the extensive set of experiments we perform in our study. We use the Graclus[4] software to extract a subgraph which contains 13K users and 192.4K (directed) edges among them. There are 4.14M ratings by these users of which 1.84M are numeric ratings and 2.3M are special ratings.

Our second dataset on movies is from the Movielens recommender system released by Grouplens[5] research group. The dataset consists of 6K users and 1M ratings (on scale of 1-5) on 3.7K movies. The dataset does not have an explicit social graph, so we construct an implicit one representing the flow of influence. In Movielens, influence flows indirectly via the recommendation engine that is based on collaborative filtering [14]. For instance, if a movie is recommended to Bob, it is likely that the "nearest neighbors" of Bob (as seen by the recommendation engine) must have given high ratings to the movie. We deduce these implicit relationships among users by computing similarity among users. Precisely, an edge is created between users u and v if the Jaccard index[6] w.r.t. the movies they rated is greater than a threshold (0.25 in experiments). The process resulted in a graph with 6K nodes and 209K edges.

The size of the training set is 19.9K for Flixster, and 2.9K for Movielens; and test set is 5.1K for Flixster, and 741 for Movielens. Our experiments were run on 650 movies randomly picked from the test set for Flixster, and the entire test set of 741 movies for Movielens.

Model Parameters from Data. For both the datasets, edge weights are learned as described in Section 3.4. Figures 2(b) and 2(c) show the distribution of edge weights for Flixster and Movielens respectively. Flixster allows ratings from 0.5 to 5 in steps of 0.5, while Movielens allows integer ratings from 1 to 5. Figures 2(d) and 2(e) show the distribution of ratings for Flixster and Movielens respectively. The complete ratings matrix R is computed using the matrix factorization method described in [14] for both the datasets.

Flixster allows two types of *special* ratings – "want to see it" and "not interested" – in addition to the numeric ratings. There are 2.3M special ratings in our dataset, of which 730K ratings are "want to see it" and 1.6M are "not interested". We map the "want to see it" and "not interested" ratings to the PROMOTE and INHIBIT states in our model and fix their numeric values to 0.5 and 5 respectively. For any user $v \in V$, the adoption probability λ_v is computed as the fraction of times v gave a numeric rating over the number of all (numeric and special) ratings given by v. Similarly, μ_v is computed as a fraction of times v gives a "want to see it" rating over the number of special ratings given by v. Figures 2(f) and 2(g) show the distribution of the λ and μ parameters learned using the special ratings. In both the distributions, a large majority of the users have λ and μ values close to the ends of the spectrum i.e., 0 and 1. A value of one (zero) for λ for a user indicates that she adopts (tattles about) each product which she learns about. Similarly, a value of one (zero) for μ indicates if the user decides to tattle, she promotes (inhibits) the product. The distributions in Figures 2(f) and 2(g) show these strong preferences of users. Over 1200 users gave only numeric ratings and have $\lambda = 1$. Considering the special ratings, 1400 users gave "want to watch" ratings only ($\mu = 1$) and 1500 users gave "not interested" ratings only ($\mu = 0$). The fact that over half of all ratings are special and represent opinions shared by users who have not watched the given movie emphasizes the need to study the effect of TATTLE nodes on adoption.

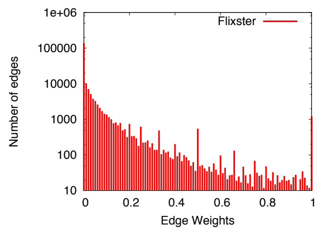
Movielens also features special ratings, such as, "want to see it" that corresponds to movies in a user's wishlist, and "hide this" that corresponds to "not interested". However, the actual dataset obtained from the Grouplens website did not include these ratings. In order to mitigate this, we use the distribution of λ and μ learned from Flixster for com-

[1]`www.flixster.com`
[2]`www.movielens.org`
[3]`www.last.fm`
[4]`www.cs.utexas.edu/users/dml/Software/graclus.html`
[5]`www.grouplens.org/node/73`

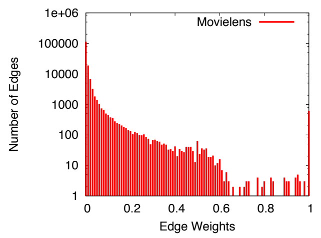
[6]Jaccard index for sets X, Y is $J(X,Y) = |X \cap Y|/|X \cup Y|$.

	Flixster	Movielens	Last.fm
#Nodes	13K	6040	1892
#Edges	192.4K	209K	25.4K
Avg. degree	14.8	34.6	13.4
#Movies or #Artists	25K	3706	17.6K
#Ratings	1.84M	1M	259K
#Edges with non-zero weight	75.7K	154K	157K

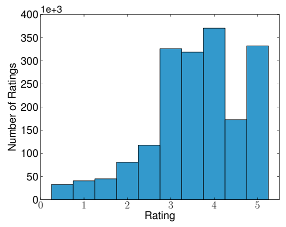
(a) Dataset statistics



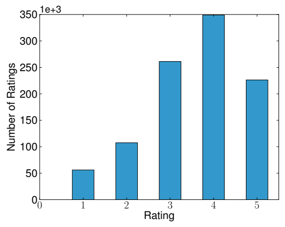
(b) Distribution of edge weights in Flixster



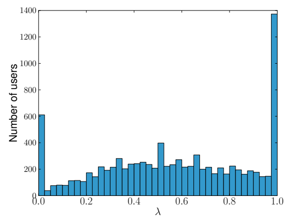
(c) Distribution of edge weights in Movielens



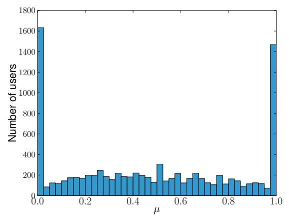
(d) Distribution of ratings in Flixster



(e) Distribution of ratings in Movielens



(f) Distribution of λ in Flixster



(g) Distribution of μ in Flixster

Figure 2: Summary of datasets and distributions of model parameters

puting the values and for users in the Movielens dataset. Though not ideal, we believe this is a reasonable approximation since both data sets represent users' opinions on movies. The shape of the distributions in Figures 2(f) and 2(g) suggests a Beta probability distribution as a natural choice for modeling both λ and μ. We learn the parameters for the Beta distribution by fitting a curve to the data from Flixster, and obtain the distributions $\lambda_u \sim \text{Beta}(0.3, 0.1)$ $\mu_u \sim \text{Beta}(0.001, 0.001)$, for any user $u \in V$.

Coverage with different models. We first analyze the effect of ratings on product adoption. Figures 3(a) and 3(b) show the coverage obtained for a seed set size $k = 50$ using the classical LT model, LT Ratings, LT Tattle and the LT-C model for maximizing product adoption. The figures show this comparison for a particular movie, and the results were similar for a set of 30 and 20 movies picked randomly from Flixster and Movielens, respectively. As can be seen, there is a huge gap between the prediction of classical LT and of LT-C model. For instance, on Flixster, LT model predicts a coverage of 4200 with a budget of 50, and LT-C predicts a coverage of only 701 for that budget. Which model is correct? Is LT-C too pessimistic or is the classical LT model too optimistic? Coverage is an important statistic used in estimating the revenue (and thus profit) and hence, it is important to have an accurate estimate of it. As shown next, gauging models based on the absolute coverage they predict can often lead to models that perform poorly.

Accuracy of estimated coverage. We evaluate the coverage obtained by different models against the actual adoption of a product. We define the *actual adoption coverage* (actual coverage, for short) as the number of users who adopted the product, in this case, gave a numeric rating for a given movie. Since the coverage estimated by a model depends on the size of the seed set, we fix the seed set for each movie and compute the coverage by each model given the seed set. The seed set for each movie is the set of *initiators* which includes all users who watched a given movie before any of their friends did. Figures 3(c) and 3(e) validate that the LT-C model estimates actual adoption coverage accurately on both datasets. Each point in the figures represents a movie and its coordinates correspond to the actual coverage vs. that estimated by a model, for the set of initiators as the seeds. The set of initiators and hence the size of the seed set may be different for each movie. Figure 3(d) shows that the actual coverage is positively correlated with the number of initiators in both the datasets (shown are all movies in the datasets). Moreover, the final coverage is much larger than the number of initiators, suggesting that the network structure plays a significant role in the final spread of adoption.

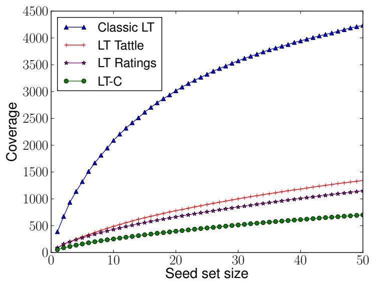
Table 1: Average RMSE for different models

Model	Flixster	Movielens
LT	1594.8	826.8
LT Ratings	269.3	301.8
LT Tattle	183	420.7
LT-C	144.6	301.7

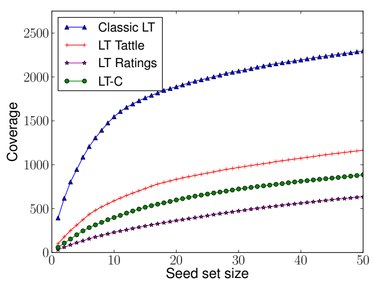
Figure 3(c) shows 650 movies and 3(e) shows 741 movies of the test set for Flixster and Movielens, respectively. The $x = y$ line is the ideal scenario where the model estimate is identical to the actual coverage. As seen from the figures, the classical LT model over-estimates the coverage by large amounts. Both including tattle nodes and including ratings are useful in providing better estimates, while estimates of adoption using LT-C model are closest to reality. To quantify this error in estimating the coverage, we compute the average root mean square value (RMSE[7]) over the test set, for the coverage predicted by the different models w.r.t. actual coverage. We observe that the RMSE obtained using the classical LT model is over 10 times that by our model (on Flixster), as shown in Table 1. On Movielens, the LT-C and LT Ratings models have similar RMSE. This might be because we do not have data to learn values of λ and μ for various users, instead we draw their values from the Beta distribution whose parameters are taken from Flixster dataset (see above).

Another interesting trend observed in Figure 3(d) is that the choice of seeds is critical. For instance, we picked the 109 movies from Flixster where the number of the initiators

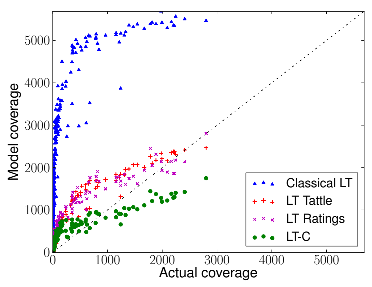
[7]For vectors x, y of size n, $\text{RMSE}(x, y) = \sqrt{\sum_1^n (x_i - y_i)^2 / n}$.



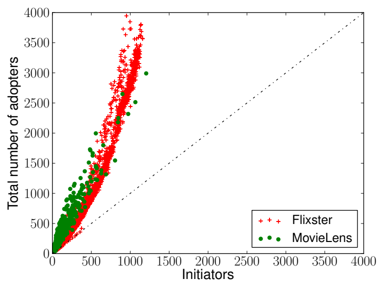
(a) Coverage obtained with different models on Flixster



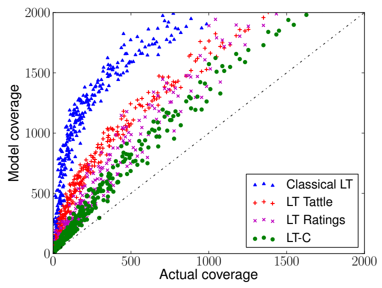
(b) Coverage obtained with different models on Movielens



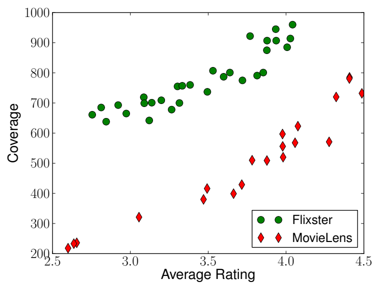
(c) Actual vs predicted coverage on Flixster



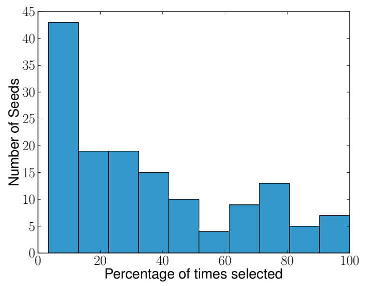
(d) Adoption Spread



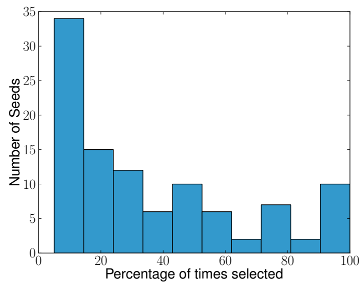
(e) Actual vs predicted coverage on Movielens



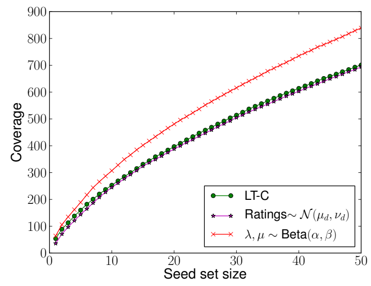
(f) Average rating of movie vs coverage for $k = 50$ on Flixster and Movielens



(g) Consistency of chosen seeds across actions on Flixster



(h) Consistency of chosen seeds across actions on Movielens



(i) Coverage with λ, μ and ratings from data and distributions on Flixster

Figure 3: Coverage and seed set analysis on Flixster and Movielens datasets

is 50. The average final coverage on these movies is 58.2, which is quite close to the number of initiators. On the other hand, if the initial seed set is picked carefully using the LT-C model, it is possible to achieve the coverage of even more than 500, as seen in Figure 3(a).

Product and seed set analysis. In Figure 3(f) we study the effect of ratings on the overall coverage given a budget of 50 seeds. Figure 3(f) shows the coverage for a set of 30 and 20 movies picked at random for the Flixster and Movielens datasets respectively. By taking ratings into account, the LT-C model predicts that good movies (rated high) are likely to have larger viewership compared with bad movies (rated low). By contrast, the coverage estimated using the LT model is the same across all movies, and is approximately 4200 nodes for movies in Flixster and 2300 for the Movielens dataset, significantly higher than the actual coverage in both cases.

A natural question to ask is whether there are some nodes that are influential across actions (movies) and are *consistently* selected as seeds. Figures 3(g) and 3(h) show a histogram of the percentage of times a node was selected as a seed for the same set of movies in Figure 3(f). For the Flixster dataset, there are 7 seeds among the 144 unique seeds that are consistently selected for 90% of the movies analyzed. The consistency decays exponentially where only a few seeds have high consistency and most seeds have low consistency. A similar trend was observed for the Movielens dataset, where 10 out of 104 unique seeds were consistently selected 90% of the times. Interestingly, these nodes are not the top high degree nodes in the graph. This observation is consistent with a recent study [17] on influence in Twitter.

Evaluating model parameters. During our study, we realized that sometimes we may not have access to the data required for computing λ and μ for each user in the network, instead, we may be able to infer the distribution of both these parameters. We evaluate the coverage obtained by drawing each model parameter from appropriate distributions and comparing it with that obtained when all parameters are computed from the data. The results are shown in Figure 3(i). We analyze the coverage when the ratings are drawn from a normal distribution $\mathcal{N}(\mu_d, \nu_d)$ where the distribution mean $\mu_d = 3.59$ and variance $\nu_d = 1.01$, as computed from the given numeric ratings. The obtained coverage is almost the same as that obtained when ratings are computed using matrix factorization from the available ratings. Next, we analyze the coverage when λ and μ are drawn from Beta distributions as $\lambda_u \sim \text{Beta}(0.3, 0.1)$ $\mu_u \sim \text{Beta}(0.001, 0.001)$, for each user $u \in V$. These distributions are learned by fitting a Beta probability distribution to the λ, μ values computed from the Flixster data. The obtained coverage comes close to that obtained from data. In conclusion, it is encouraging to observe that drawing the model parameters from a distribution that approximates the data may well give a good estimate of the coverage if access to user logs is limited.

4.2 Adoption of Music

Datasets. We study the adoption of music on a dataset from the popular music service Last.fm. The dataset we used was released for the Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011)[8]. The details of the dataset are presented in the table in Figure 2(a). It has 1892 users with 25.4K friendship relationships among them. The dataset includes the users' listening history in the form of number of times a user listened to songs by an artist, and users' tagging history for tagging artists.

Model Parameters from Data. We use both tagging and listening history as actions for computing edge weights. Since the listening history is only available as playcounts, it does not come with timestamps. We assume these are most recent actions and assign the current timestamp to all listening actions. Now, edge weights can be learned as before. The distribution of edge weights is shown in Figure 4(a). The data was divided into test and training sets, with the test set consisting of 3067 randomly picked artists.

A user's opinion of an artist is implicitly present in the form of playcounts, i.e., the number of times the user has listened to songs by that artist. To infer ratings on a scale of 0 to 5 for each user, we partition the artists into 5 buckets, where the top bucket represents artists in the top 20-percentile playcounts, the second bucket in the next 20-percentile and so on. Each artist in the top bucket is assigned a rating of 5, next bucket corresponds to rating 4 and so on. It is well known that listening history follows a power-law distribution where a few artists (or songs) are heard many times and many artists (songs) are heard very few times. This pattern is observed in the obtained ratings as shown in Figure 4(b).

Last.fm allows users to indicate their preference on songs by marking them as "loved" or "banned". For a given song, we regard the users who loved that song as being in the ADOPT state and those who banned that song as being in the INHIBIT state. Given this mapping, we can use the "loved" and "banned" songs for inferring a distribution of λ. However, the dataset for the HetRec workshop did not contain the information on loved and banned songs. We used the API from Last.fm to crawl this information for an arbitrary set of 10K users, and computed λ_u for a crawled user u as the fraction of loved songs over the number of loved and banned songs. Figure 4(c) shows the distribution of λ for the crawled users. A trend similar to that in Flixster is observed for values of λ in Last.fm, although with a larger skew towards adopting the product (in this case, an artist). We fit a Beta probability distribution to the values of λ and the best fit was obtained with $\text{Beta}(0.5, 0.001)$. Since the consumption of songs from an online music service does not cost much, it is not natural for users to want to listen to a song but not actually listen to it. Therefore, for this domain, $\mu_u = 0$, for every user u.

Accuracy of estimated coverage. Figure 5(a) compares the coverage obtained using classical LT, LT Tattle, LT Ratings and LT-C models with the actual coverage obtained by the initiators. Since the distribution of λ as obtained from the data is highly skewed towards 1, including the TATTLE state does not make the coverage estimate much better than the classical LT model. On the other hand, most ratings are low and hence their inclusion into the LT model has a greater impact for this dataset. The estimated coverage of LT Ratings and LT-C are similar, and provide a far better estimate compared with the classical LT model. The RMSE values averaged over 3067 artists for the four models were as follows, classical LT:97.5, LT Tattle:93.6, LT Ratings:31.7 and LT-C:31.7. These errors in coverage estimation show that for the music domain as well (where adoption may not cost money), incorporating users' ratings in the propagation model provides better estimates of adoption, compared with accounting for tattling but not ratings. In Figure 4(d), we present the actual coverage against the number of initiators, for all the artists. Again, it can be seen that the two are positively correlated.

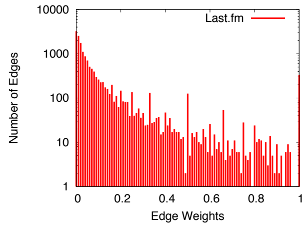
Product and seed set analysis. Figure 5(b) shows that the coverage varies for artists that have different popularity (or rating). The coverage estimated by the classical LT model is the 1230, regardless of the artist, again way off from reality. The trend in consistency of seeds observed for music domain is similar to that seen for movies. Figure 5(c) shows that few (i.e., 12) seeds are picked over 90% of the times and 62 out of 149 unique seeds are picked in less than 10% of the 20 artists analyzed. These two experiments show that using LT-C, it is not only the coverage that is different for different products, even the target seeds are different. In contrast, the classical model disregards these nuances. This suggests taking such nuances into account may lead to a more accurate and effective model for viral marketing.

5. CONCLUSIONS AND FUTURE WORK

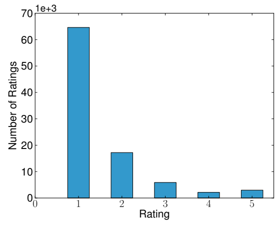
Classical diffusion models such as IC and LT do not distinguish between influence and product adoption. Thus, they implicitly assume that once influenced, a node necessarily adopts a product and that adopters always influence other users to adopt the product. Our observations on real data show that sometimes influenced users, once they become active, may choose to not adopt but instead tattle about the product; by doing so, they may either promote or inhibit adoption by other users. Furthermore, adopters may endorse a product to varying degrees based on their experience with the product, as often reflected by the ratings they provide in several real data sets.

In this paper, we proposed a propagation model called LT-C model that accounts for these observations. We formalized the problem of adoption maximization as distinguished from influence maximization and showed that it is NP-hard. We also showed the expected adoption spread function under the LT-C model is monotone and submodular and thus, the classic greedy algorithm can be used to get an approximate

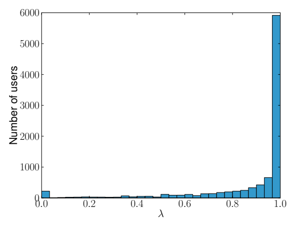
[8] `http://ir.ii.uam.es/hetrec2011`



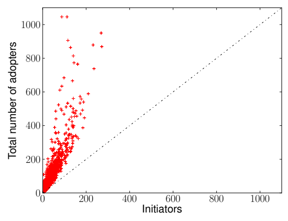
(a) Edge weight distribution



(b) Distribution of ratings

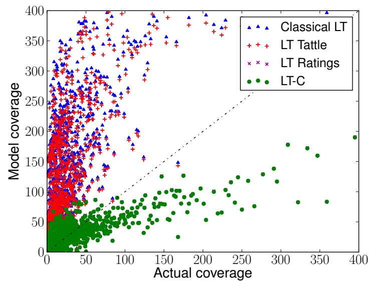


(c) Distribution of λ

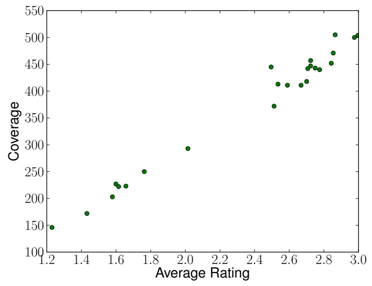


(d) Adoption Spread

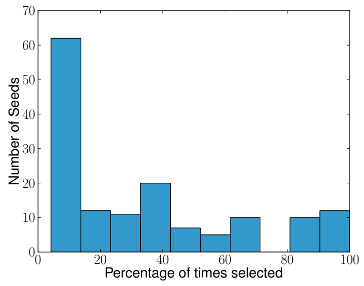
Figure 4: Summary of dataset and distributions of model parameters for Last.fm



(a) Actual vs predicted coverage



(b) Average rating of movie vs coverage for $k = 50$



(c) Consistency of chosen seeds across actions

Figure 5: Coverage and seed set analysis on Last.fm dataset

solution. We conducted extensive experiments on two domains – two data sets from movies and one from the music domain. Our results show that w.r.t. the accuracy of spread prediction, the LT-C model is consistently the best and the classical LT model is consistently the worst.

There is still a long way to go to develop a truly realistic product adoption model. It may be possible that the users are passive and may not express any opinion even after adopting a product [17]. One way to incorporate this is to split the ADOPT state into two sub-states, viz., "adopt and rate" and "adopt and not rate". However, it is not clear how we can identify the latter state (for learning the model parameters) in the datasets we can access. Similarly, users may not find tattlers' ratings as trustworthy as adopters' ratings. Marginalizing tattlers' ratings may improve the predictions. Extending the model by incorporating the five stages of product adoption [2] can be useful. Inclusion of negative opinions in the spirit of [3] within the framework of our model is interesting and may lead to an even more expressive model for product adoption. Next, it is important to validate the LT-C model against many more real data sets from diverse domains. Last but not the least, scalable heuristic algorithms need to be developed for the LT-C model in order to handle very large networks and seed sets. Our ongoing work addresses some of these questions.

Acknowledgments. This research was partially supported by a strategic network grant from NSERC Canada on Business Intelligence Network. We thank Mohsen Jamali and Martin Ester for sharing the Flixster dataset, and Mohammad Khabbaz for sharing the matrix factorization code.

6. REFERENCES

[1] S. Aral, L. Muchnik, and A. Sundararajan. Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks. In *PNAS*, 2009.

[2] J. M. Bohlen and G. M. Beal. The diffusion process. *Spl. Report No. 18, Agri. Extn. Serv., Iowa State College*, 1957.

[3] W. Chen, A. Collins, R. Cummings, T. Ke, Z. Liu, D. Rincon, X. Sun, Y. Wang, W. Wei, and Y. Yuan. Influence maximization in social networks when negative opinions may emerge and propagate. In *SDM*, 2011.

[4] W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *ICDM*, 2010.

[5] D. Crandall, D. Cosley, D. Huttenlocher, J. Kleinberg, and S. Suri. Feedback effects between similarity and social influence in online communities. In *KDD*, 2008.

[6] P. Domingos and M. Richardson. Mining the network value of customers. In *KDD*, 2001.

[7] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. Learning influence probabilities in social networks. In *WSDM*, 2010.

[8] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. A data-based approach to social influence maximization. *PVLDB*, 5(1), 2011.

[9] A. Goyal, W. Lu, and L. V. S. Lakshmanan. Simpath: An efficient algorithm for influence maximization under the linear threshold model. In *ICDM*, 2011.

[10] J. Hartline, V. Mirrokni, and M. Sundararajan. Optimal marketing strategies over social networks. In *WWW*, 2008.

[11] M. Jamali and M. Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *RecSys*, 2010.

[12] S. Kalish. A new product adoption model with price, advertising, and uncertainty. *Management Science*, 31(12), 1985.

[13] D. Kempe, J. M. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *KDD*, 2003.

[14] Y. Koren and R. Bell. Advances in collaborative filtering. In *Recommender Systems Handbook*. 2011.

[15] T. La Fond and J. Neville. Randomization tests for distinguishing social influence and homophily effects. In *WWW*, 2010.

[16] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *KDD*, 2007.

[17] D. M. Romero, W. Galuba, S. Asur, and B. A. Huberman. Influence and passivity in social media. In *WWW*, 2011.

[18] J. Tang, J. Sun, C. Wang, and Z. Yang. Social influence analysis in large-scale networks. In *KDD*, 2009.