Chapter 3

Link Prediction based Influence Maximization in Dynamic Social Networks

This chapter focus on the first objective of this thesis, i.e. link prediction based influence maximization in dynamic social networks. We give an introduction, motivation, and contributions for the considered problem in section 3.1. Section 3.2 gives the formal problem definition. Section 3.3 explains the proposed framework as the solution to the defined problem. Experimental details are given in section 3.4, and their outcomes are discussed in section 3.5. Section 3.6 concludes the overall outcome of the chapter.

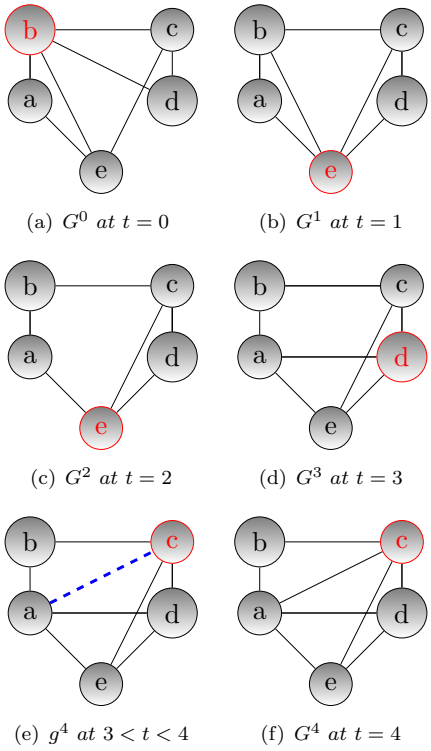
3.1 Introduction

Influence Maximization (IM) is the problem of finding a small set of highly influential users in the social networks. Most of the present IM solutions neglect the highly dynamic behaviour of social networks. It can result in either deprived seed qualities or a prolonged processing time. It is natural and significant to understand that social networks have a continuous change in structure [166, 167, 168]. These changes in structure often occur in real applications; for instance, connections appear and disappear when users friend/unfriend

others on Facebook [169] alternatively, follow/unfollow others on Twitter [170]. With the change in the structure of social networks, the amount of influence a person can have on others also keeps changing. People also get more affected by neighbours' she/he communicates more often. A neighbours' impact decreases if he/she does not communicate with the person for a long time. Thus using one static seed set need not give good performance for influence maximization in online social networks.

In dynamic networks, the number of snapshots is the graph instances taken after a fixed interval of time (timestamps) for an evolutionary graph in which the edges are being added/removed with the increase of time. Here, we need to select suitable seed nodes at different timestamps for achieving the maximum influence spread. Changes in structure affect the spreading capacity of seed nodes. To demonstrate the idea of link prediction-based IM in dynamic social networks, consider an example shown in figure 3.1. This example shows the links (edges) between users (nodes) at different timestamps. Each edge indicates that a user can influence another user. Figures 3.1-a, 3.1-b, 3.1-c, 3.1-d shows the snapshots G^0, G^1, G^2, G^3 of an evolving graph at the time-stamps $t = 0, 1, 2, 3$, respectively. The most influential node at timestamp $t = 0$ in snapshot G^0 is b, as it seems to influence the maximum number of nodes. Similarly, at time $t = 1$, $t = 2$ and $t = 3$, nodes e, e and d are the most influential nodes, respectively. Even in this simple example, we can see that the most influential node can be different for different snapshots of the graph.

For dynamic social networks, the seed nodes selected for a particular snapshot may not be influential at other snapshots of the network. So, we need to compute the seed nodes for each snapshot of the network. However, the seed selection process itself takes a significant amount of time for large networks, and if the network is highly dynamic in nature, then the computed seed nodes may not be the best ones due to continuous changes in network structure. To deal with this problem, we propose a new influence maximization algorithm, LPINT. In our proposed algorithm, the crucial idea is to predict the next snapshot of the graph by considering the evolving graph's temporal and structural behaviour. For example, in figure 3.1 (e), g^4 is the predicted snapshot at $3 < t < 4$, and node c is chosen as the seed node for the predicted graph g^4. We use this predicted seed node for information



(a) G^0 *at* $t = 0$ (b) G^1 *at* $t = 1$

(c) G^2 *at* $t = 2$ (d) G^3 *at* $t = 3$

(e) g^4 *at* $3 < t < 4$ (f) G^4 *at* $t = 4$

FIGURE 3.1: (a), (b), (c), (d), (f) are the Snapshots of the Graph $G = \{G^0, G^1, G^2, G^3, G^4\}$ respectively, and (e) is the Predicted $g^{t+1} = g^4$, here in g^4, Link $a - c$ is Expected to Appear in Snapshot G^4.

diffusion in actual upcoming snapshot G^4 at time $t = 4$ for efficient and faster information diffusion. In our proposed method, we predict the next snapshot of the graph using an efficient link prediction method. We then find the seed set for the predicted snapshot using an efficient IM technique. This seed set is used for information diffusion in the next snapshot of the graph.

For link prediction, we use the ctRBM technique, which combines the temporal as well as structural behaviour of the nodes in evolving graphs to predict the upcoming links. Next, to select the efficient seed set, we tried various state-of-the-art algorithms of influence maximization and found that the Upper Bound based Lazy Forward (UBLF) [127] approach works better in our problem. We use an improved *UBLF* algorithm named *EXCHANGE* algorithm in our proposed LPINT framework. To make the *EXCHANGE* algorithm run faster, we find the most influential nodes in the current snapshot starting

from the set of seed nodes found from the previous snapshot rather than from an empty set.

In this chapter, we assume that the active nodes are the ones that have communicated in past N snapshots. We find seed nodes only among these active nodes. For example, let us consider the friendship/following relationship of the social network. Many nodes in the network do not communicate frequently, or some of them are less active or inactive. Considering these inactive nodes for the IM problem would be a waste of time and resources. These inactive nodes are rarely helpful in IM applications like viral marketing or fake news containment. In the proposed work, we consider only the active nodes in the seed selection process. Implementing this assumption is novel and makes our goal of influence maximization more efficient and effective.

Briefly, our contributions in this chapter can be stated as: first, we define a novel Influential Node Tracking problem to maximize the influence spread in an online social network. Then, we propose an LPINT framework for efficient and effective influence maximization in dynamic social networks. The proposed model uses the link prediction technique to predict the upcoming snapshot of the graph and then computes the seed set for influence maximization. Finally, through experiments, we show that the proposed framework performs better in terms of influence spread in comparison to the considered baseline techniques on considered datasets. See Appendix A for our research paper supporting this work.

3.2 Problem Description

The goal of the proposed model is to predict graph G^{t+1} from graph snapshots up to G^t and find seed set S^{t+1} based on predicted G^{t+1}, which maximizes the influence function $\sigma_{t+1}(\cdot)$ at every snapshot $\hat{G}^{t+1}$ at time $t+1$. So, there are two parts in the proposed model; first part predicts the upcoming snapshot and second part finds the suitable seed nodes.

In this chapter, we have used time-series based link prediction using the ctRBM method, which was introduced in Section 2.3. To find the most influential seed nodes in online social networks, we need to track the dynamic behaviour of the networks. Here, we consider the sequence of the snapshot graphs $G^0, G^1, \ldots, G^t$ at time-stamp $T = 0, 1, 2, \ldots, t$, respectively. In this thesis, we consider the dynamics of the graph in terms of edge change with time, i.e., an edge gets added in the graph if there is communication between a pair of nodes in the networks. Each snapshot graph shows an undirected graph termed as a ***growing graph*** and denoted as $G^t = (\mathbb{V}, E^t)$, where $\mathbb{V}$ is the set of nodes and E^t is the set of edges showing the nodes communication between time-stamp $t-1$ and t. A propagation probability $P^t_{u,v}$ is associated with each edge of every snapshot graph G^t.

As the selection of seed set S_t itself takes significant time for large graphs, it is possible that by the time S_t is computed, the graph might evolve from G^t to $G^{t+\delta}$, where we assume *computation time for* $S_t \leq \delta <$ *snapshot interval*. So S_t may become less effective for influence maximization in the actual snapshot $\hat{G}^{t+\delta}$. To handle this problem, we propose a novel approach in which we use a time series dependent link prediction method to predict G^{t+1} by considering the evolution pattern of the graph, then we find the probable seed set S_{t+1} based on predicted G^{t+1}. And we use the predicted seed set S_{t+1} for influence maximization in the actual snapshot $\hat{G}^{t+1}$ at time $t+1$. We now formally define this Link Prediction based Influential Node Tracking (LPINT) problem in online social networks.

Problem Definition: *Let* $\rho = \{G_i\}_0^t$ *be an online social network. The LPINT problem is to find a sequence of seed sets* $S_1, \ldots, S_t$ *whose size is maximum* k*, such that* $S_t = \arg\max_{S_t \in V, |S_t| \leq k} \sigma_t(S_t)$ *for all snapshot graphs* G^t.

We propose a straightforward method to solve the LPINT problem. This method uses an efficient link prediction technique on the sliding window (a set snapshots) of snapshots of the dynamic graph to predict the next snapshot of the graph and then applies an effective Influence Maximization algorithm to find the predicted set of seed nodes in the predicted

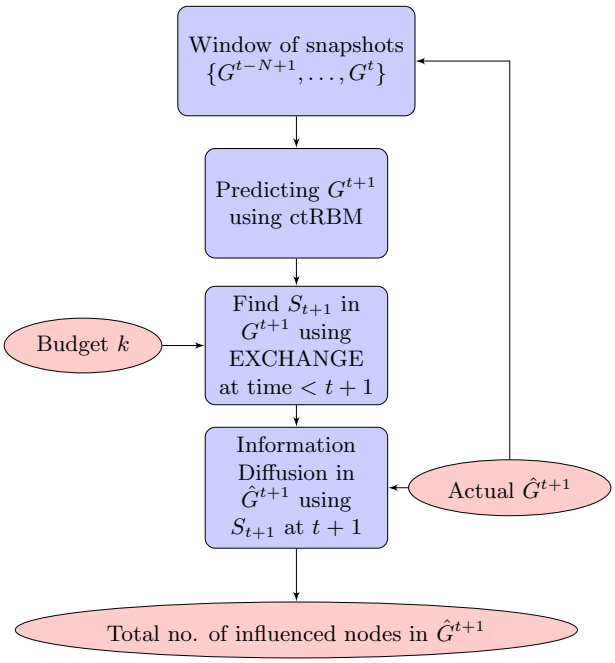


FIGURE 3.2: Block Diagram Showing LPINT Steps

snapshot of the graph. This predicted seed set is used for influence maximization in the actual upcoming snapshot of the graph.

3.3 Proposed Framework

The proposed LPINT framework is given by a block diagram in Figure 3.2. Here, we consider the dynamic behaviour of the graph in the past and depending on the time-series pattern of graph snapshots; we are predicting the next snapshot of a graph using the Link Prediction method. We then apply the proposed IM algorithm to find the probable seed set for the upcoming graph snapshot. More detailed descriptions about how our method works on the snapshot graphs and dynamic networks are presented in the next two subsections.

3.3.1 Predicting G^{t+1} using Link Prediction

We predict the appearance of new links by exploring the evolution pattern of the graph. We have used ctRBM [144], which adopts temporal variations (temporal connections) and neighbour opinions (neighbour connections) during the training phase and performs prediction dependent on the existing time window of snapshots and the local neighbour's predictions of each pair of nodes. The model trains a ctRBM denoted as L_m for each node m, and then collects a set of ctRBMs denoted by $\mathbb{L}$. The prediction is, therefore, G_{t+1} done by aggregating the results from each L_m.

3.3.1.1 Temporal Connections

Let N be the size of the window, which is a tunable parameter indicating the number of time steps (graph snapshots) we have to look back. In modelling a highly dynamic graph, N depends on the snapshots interval so as to better manage evolutionary networks. We assume that the graph snapshots at $t-N,\ldots,t-1$ is integrated into a vector $V^{<t}$ of dimension $N \cdot |V|$. The $N \cdot (|V| \times |H|)$ size weight matrix W_A provides the weights for temporal connections. Since the model has transitory information, the conditional probability at time t is represented as:

$$P(H^t|V^t, V^{<t}; \theta) = \beta \cdot \omega(y + W'_A V^{<t}) + (1-\beta) \cdot \omega(y + W' V^t), \tag{3.1}$$

$$P(\widetilde{V}^t|H^t) = \omega(x + WH^t), \tag{3.2}$$

here, visible layer variables are denoted as set V and Hidden layer variables are denoted as set H. β is a hyperparameter used to balance the static and dynamic characteristics of the graph. x and y are the biases for V and H respectively. W' is the transposes of W. $\widetilde{V}$ is the reconstructed data, representing the model's estimation. The goal of learning is to minimize the distance between $\widetilde{V}$ and V. θ is the parameter of ctRBM model R. ω is the logistic function defined as $\omega(Z) = (1 + exp(Z))^{-1}$

3.3.1.2 neighbour Connections

We can explain the common assumption that an individual's conduct is influenced by his/her neighbour's circle by considering this. To formulate this, we define the neighbour impact as the desirability of its prediction. If the total number of nodes in $\mathbb{V}$ is $\hat{p}$, the neighbour impact can be written as:

$$\eta_m^t = \frac{1}{U_m^t} \sum_{n=1}^{\hat{p}} l(x_m^t, x_n^t) \times P(\widetilde{V}_n^t | H_n^t) \times P(H_n^t | V_n^t, V_n^{<t}; \theta_n), \tag{3.3}$$

here, $U_m^t = \sum_{n=1}^{\hat{p}} l(x_m^t, x_n^t)$, and the indicator function l is 1 if node n is connecting to node m at time t, and 0 otherwise. And θ_n is the parameter of ctRBM model R_n for neighbour node n.

neighbours make their predictions for node m based on their past. It can be seen from Eq. 3.3 that the opinion η_m^t of a node m is an average of its neighbour's opinions. Since models are already trained using last $t-1$ snapshots, $P(\tilde{V}_n^t | H_n^t) \cdot P(H_n^t | V_n^t, V_n^{<t}; \theta_n)$ can be effectively computed by Eq. 3.1 & 3.2 by substituting V^t in Eq. 3.1 by V_m^t.

3.3.1.3 Training and Inferences on ctRBM

In the ctRBM [171], the state of the hidden units is controlled by the contribution from individual perception V^t and $V^{<t}$ and the input η^t from neighbours. Given V^t and $V^{<t}$, the hidden units at time t are restrictively independent. The impact of the neighbour influence can be seen as adaptive bias:

$$\bar{x}^t = \gamma \cdot x + (1 - \gamma) \cdot \eta^t, \tag{3.4}$$

which includes the static bias, x for the current observation, and the contribution from the neighbours. Here γ is a hyperparameter that says how much an individual complies with his/her neighbours. In our experiments, we set it to be 0.5. Hence in Eq. 3.2 x is replaced with $\bar{x}^t$ to obtain:

$$P(\widetilde{V}^t | H^t) = \omega(\bar{x}^t + W H^t). \tag{3.5}$$

The detailed algorithm for inference used by ctRBM is shown in Algorithm 1.

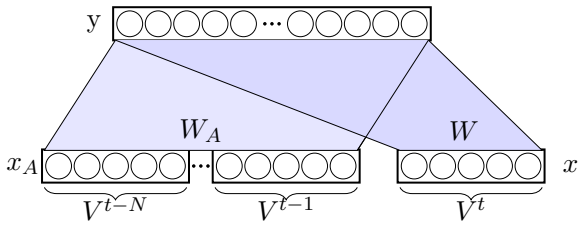


FIGURE 3.3: Restricted Boltzmann Machine with Temporal Information, here the Window Size is N.

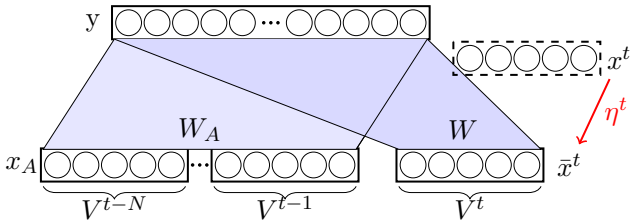


FIGURE 3.4: A Conditional Restricted Boltzmann Machine with Summarized neighbour Influence η^t Integrated into an Adaptive Bias into the Energy Function.

Algorithm 1 PREDICT($\{G^t\}_{t-N+1}^{t}, \mathbb{L}$) [144]

Require: : A trained $\mathbb{L}$ for all nodes, in which $L_m \in \mathbb{L}$ has parameters θ_m : $\{W_{Am}, W_m, x_{Am}, x_m, y_m\}$, Snapshots $\{G^{t-N+1}, \ldots, G^t\}$, here N is the size of window
Output: : Predicted graph G^{t+1}

Initialize: $m \leftarrow 1, G^{t+1} \leftarrow zero(size(\mathbb{V}), size(\mathbb{V}))$
for $m < size(\mathbb{V}) + 1$ **do**
  $V_m^{<t+1} \leftarrow \{G_m^{t-N+1}, \ldots, G_m^t\}$
  $V_m^{t+1} \leftarrow one(1, size(\mathbb{V})).5$
  `Take neighbour indicator:` $Jdx \leftarrow find(G_t^m == 1)$
  `Take neighbour models detail:` $\mathbb{L}_{nbr} \leftarrow \mathbb{L}(Jdx)$
  `Determine` η_m^{t+1} `by Eq.3.3 provided` $\mathbb{L}_{nbr}$
  $x_m^{t+1} \leftarrow x_m + \eta_m^{t+1}$
  `Determine` $\widetilde{V}_m^{t+1}$ `by Eq.3.1 & 3.5 replacing` V^t `and` $V^{<t}$ `by` V_m^{t+1} `and` $V_m^{<t+1}$
  $G_m^{t+1} \leftarrow \widetilde{V}_m^{t+1}$
end for
return G^{t+1}.

3.3.2 Finding Seed Nodes for Influence Maximization

The Interchange Heuristic proposed in [143] is utilized to replace the nodes in the seed set S_t. Beginning from a self-assertive set $S_t \subseteq \mathbb{V}$, Interchange Heuristic discovers a set $S'_t \subseteq \mathbb{V}$ that contains all nodes present in S_t except one node, and the number of nodes remains the same. According to Nemhauser et al., when we apply the Interchange Heuristic on any submodular monotone function till maximum improvement gives a solution having an approximation guarantee of $1/2$ [143].

Algorithm 2 *GREEDY*$(G = (\mathbb{V}, E), k)$

Require: : Graph $G = (\mathbb{V}, E)$, Number of seed nodes k
Output: : Seed set S
1: Initialize: $S = \emptyset$
2: **for** $i = 1$ `to` k **do**
3: $\quad e^* = \arg\max_{e \in \mathbb{V}-S} \{\sigma(S \cup \{e\}) - \sigma(S)\}$
4: $\quad S \leftarrow S + \{e^*\}$
5: **end for**
6: **return** S.

Algorithm 3 describes how to get the set S_{t+1} using the Interchange Heuristic. We select S_{t+1} so that the gain accomplished by means of substitution of any fixed $e_s \in S_t$ to $e \in \mathbb{V} - S_t$ is maximized. We evaluate $e^* = \arg\max_{e \in \mathbb{V}-S_t} \Delta_{e,e_{s_t}}(S_t)$, by choosing $S_{t+1} = S_t - e_s + e^*$, here $\Delta_{e,e_{s_t}}(S_t)$ is replacing gain from node e_{s_t} to e. The upper bound [131] on the replacement gain is denoted by $\bar{\Delta}_{e,e_{s_t}}(S_t)$. We stop to find another e_{S_t} for interchange if the largest replacing gain $\Delta_{e,e_{S_t}}$ is less than a given threshold $\chi \geq 0$. We are doing this to increase the speed of the process of interchange and reduce the computations for the case of minor improvements.

We determine the gain by substituting e_s with any node in $e \in \mathbb{V} - S_t$, which requires $|\mathbb{V} - S_t|$ influence estimations. Monte-Carlo simulation-based calculation for the above task is excessively costly, even for a network with a moderate size. To decrease the number of iterations for influence estimation, we use the upper bound on replacing gain as proposed in [131]. The subroutine in Algorithm 3 performs the Interchange Heuristic for any fixed $e_s \in S_t$.

Algorithm 3 EXCHANGE($G^{t+1}, S_t, e_s, \bar{\Delta}_{.,e_s}(S_t)$)

Require: : $G^{t+1}, S_t, e_s, \bar{\Delta}_{.,e_s}(S_t)$
Output: : S_{t+1}
Set $\Delta_{e,e_{s_t}} \leftarrow \bar{\Delta}_{e,e_{s_t}}(S_t)$, $e \in V - S_t$
Set $curr_e \leftarrow false$, $e \in V - S_t$
while $TRUE$ **do**
    $e^* \leftarrow \arg\max_{e \in \mathbb{V} \setminus S_t}\{\Delta_{e,e_{s_t}}\}$
    if $\Delta_{e^*,e_{s_t}} \leq \chi\sigma(S_t)$ **then**
    break
    end if
    if $curr_{e^*}$ **then**
    $S_t \leftarrow S_t - e_{s_t} + e^*$
    break
    else
    $\Delta_{e^*,e_{S_t}} \leftarrow \sigma(S_t - e_{s_t} + e^*) - \sigma(S_t)$
    $curr_{e^*} \leftarrow TRUE$
    end if
end while
$S_{t+1} = S_t$
return S_{t+1}.

Algorithm 4 LPINT($\{G^t\}_{t-N+1}^{t}, k$)

Require: : Snapshots of the graph $\{G^t\}_{t-N+1}^{t}$, size of seed set k
Output: : Seed set S_{t+1} with k nodes for snapshot G^{t+1}
$S_1 =$ `GREEDY`$(G^1(\mathbb{V}, E^1), k)$ `using Algorithm 2`
`Predicting` G^{t+1} `using Algorithm 1`
`For snapshot` G^{t+1} `compute` $\bar{\Delta}_{e,e_s}(S_t)$ `for` $e \in \mathbb{V} - S_t, e_s \in S_t$
for $i = 1$ `to` k **do**
    $e_s^* \leftarrow \arg\max_{e_s \in S_t}\{\bar{\Delta}_{.,e_s}(S_t)\}$
    $S_t \leftarrow$ `EXCHANGE`$(G^{t+1}, S_t, e_s^*, \bar{\Delta}_{.,e_s}(S_t))$
    `Update` $\bar{\Delta}_{e,e_s}(S_t)$ `for any` $e \in \mathbb{V} - S_t, e_s \in S_t$ `according to the EXCHANGE result`
end for
$S_{t+1} = S_t$
return S_{t+1}.

3.3.3 Link Prediction based Influential Node Tracking

By using the strategy of interchange from Algorithm 3, we describe our Link Prediction based Influential Node Tracking, in short, LPINT as Algorithm 4. In algorithm 4, when we start the process; firstly, we find the seed set in given snapshot G^1 by applying Algorithm 2, then we predict the upcoming graph snapshot using Algorithm 1, and then we apply Algorithm 3 to do at most k rounds of a replacement instead of doing it until no further improvement is possible, and finally, we got the fresh seed set S_{t+1} for snapshot G^{t+1}. Hence we ignore the insignificant performance improvement for reducing the computations and time of processing and hence increasing the efficiency of the overall process of seed set selection.

3.3.4 Theoretical Results

We present some theoretical results on influence maximization for dynamic networks.

Theorem 1. For growing graph $G(\mathbb{V}, E^t)$, snapshots G^t and G^{t+1} at timestamp t and $t+1 > t$, if $G^t \subseteq G^{t+1}$ and k size seed set $S_t, S_{t+1} \subseteq \mathbb{V}$ for graph G^t and G^{t+1} respectively, then the equation for the influence spread is related as:

$$\arg\max_{|S_{t+1}|=k} \sigma(G^{t+1}, S_{t+1}) - \arg\max_{|S_t|=k} \sigma(G^t, S_t) \geq 0. \tag{3.6}$$

Proof. To establish this result, we need to consider Claim 2.3 proposed in [59]. According to the claim, a node x ends up being active if and only if there is a path from a node in S to node x composed only of live edges. We also have the following relation

$$G^t \subseteq G^{t+1} \text{ implies } E^t \subseteq E^{t+1}.$$

The active node estimator function E[Active nodes] can be written as

$$E\left[\sum_{x \in \mathbb{V}} I_x\right],$$

where

$$I_x = \begin{cases} 1 & \text{if } node\ x \text{ is active,} \\ 0 & \text{otherwise.} \end{cases}$$

Thus,

$$E\left[\sum_{x\in\mathbb{V}} I_x\right] = \sum_{x\in\mathbb{V}} P_x^{G^t},$$

where $P_x^{G^t}$ is the probability that the node x is active or there is a path containing only live edges from x to S in G^t. We have,

$$P_x^{G^t} \leq P_x^{G^{t+1}}.$$

In words, the probability of x being active in $G^t \leq$ probability of x being active in G^{t+1}.

Thus we see for a given seed set S,

$$\sum_{x\in\mathbb{V}} P_x^{G^t} \leq \sum_{x\in\mathbb{V}} P_x^{G^{t+1}}.$$

As there can only be $l \geq 0$ extra paths in G^{t+1}, we have:

$$\arg\max_{|S_t|=k} \sigma(G^t, S_t) \leq \arg\max_{|S_{t+1}|=k} \sigma(G^{t+1}, S_{t+1}),$$

which concludes the proof.

□

Theorem 2. If the accuracy of link prediction is high, then the influence spread is high.

Proof. Let the accuracy of the prediction of edges be ξ, then:

$$\frac{|E^{t+1} \cap \hat{E}^{t+1}|}{|E^{t+1} \cup \hat{E}^{t+1}|} \geq \xi. \tag{3.7}$$

Here $\hat{E}^{t+1}$ is the actual number of edges at time $t+1$ and E^{t+1} is predicted edges at time $t+1$. We have for any graph G^{t+1},

$$|\arg\max\sigma(G^{t+1}, S_{t+1})| \leq |\arg\max\sigma(G^{t+1}, \hat{S}_{t+1})| \leq \lambda, \tag{3.8}$$

for some $\lambda > 0$. Now assume that the propagation probabilities $p_{u,v}$ is same for all the edges. Hence if the number of edges $|E^{t+1} \cap \hat{E}^{t+1}|$ is atleast $\xi \cdot |E^{t+1} \cup \hat{E}^{t+1}|$, we have

$$\arg\max\sigma((\hat{G}^{t+1} \cap G^{t+1}), \hat{S}_{t+1}) \geq \arg\max\sigma((\hat{G}^{t+1} \cup G^{t+1}), S_{t+1}) \cdot \xi$$

Now we have:

$$\begin{aligned}
&|\arg\max\sigma(\hat{G}^{t+1}, \hat{S}_{t+1}) - \arg\max\sigma(G^{t+1}, S_{t+1})| \leq \\
&|\arg\max\sigma((\hat{G}^{t+1} \cup G^{t+1}), \hat{S}_{t+1}) - \arg\max\sigma((\hat{G}^{t+1} \cap G^{t+1}), S_{t+1})| \leq \\
&|\arg\max\sigma((\hat{G}^{t+1} \cup G^{t+1}), \hat{S}_{t+1}) - \arg\max\sigma((\hat{G}^{t+1} \cup G^{t+1}), S_{t+1}) \cdot \xi|.
\end{aligned} \tag{3.9}$$

Combining equations (3.8) and (3.9), we get

$$|\arg\max\sigma(\hat{G}^{t+1}, \hat{S}_{t+1})| - |\arg\max\sigma(G^{t+1}, S_{t+1})| \leq \lambda(1-\xi).$$

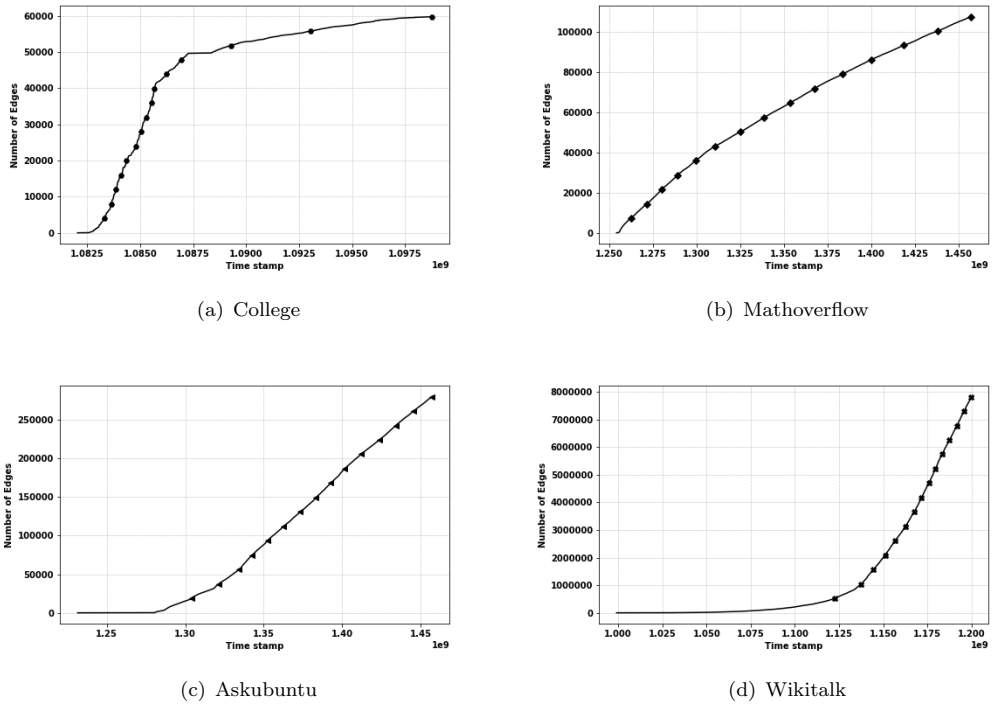
Thus we can conclude if the prediction accuracy ξ is high the difference between influence spread in the predicted and actual graph is small. □

3.4 Experiments

3.4.1 Dataset Used

We performed our experiments on four real-world dynamic networks: College, Mathoverflow, Askubuntu, and Wikitalk datasets. The datasets are introduced in section 2.5.1.

We generate the snapshot $G^t = (\mathbb{V}, E^t), V = \bigcup \mathbb{V}^t$ at timestamp t by considering all the edges appearing during the time $[t - \delta t, t]$, where δt is the time duration between two snapshots. The dynamic behaviour of the dataset is represented in the graphs shown in



(a) College (b) Mathoverflow

(c) Askubuntu (d) Wikitalk

FIGURE 3.5: Number of Edges versus Time-stamp Graph for Datasets

Figure 3.5. It shows the graph plot of timestamp versus the number of communications for different datasets.

3.4.2 Baseline Methods

Baseline approaches used for comparison of influence maximization in evolving networks with and without link prediction and approaches used for comparison of Online Influence Maximization with the proposed method are introduced in section 2.6.1.

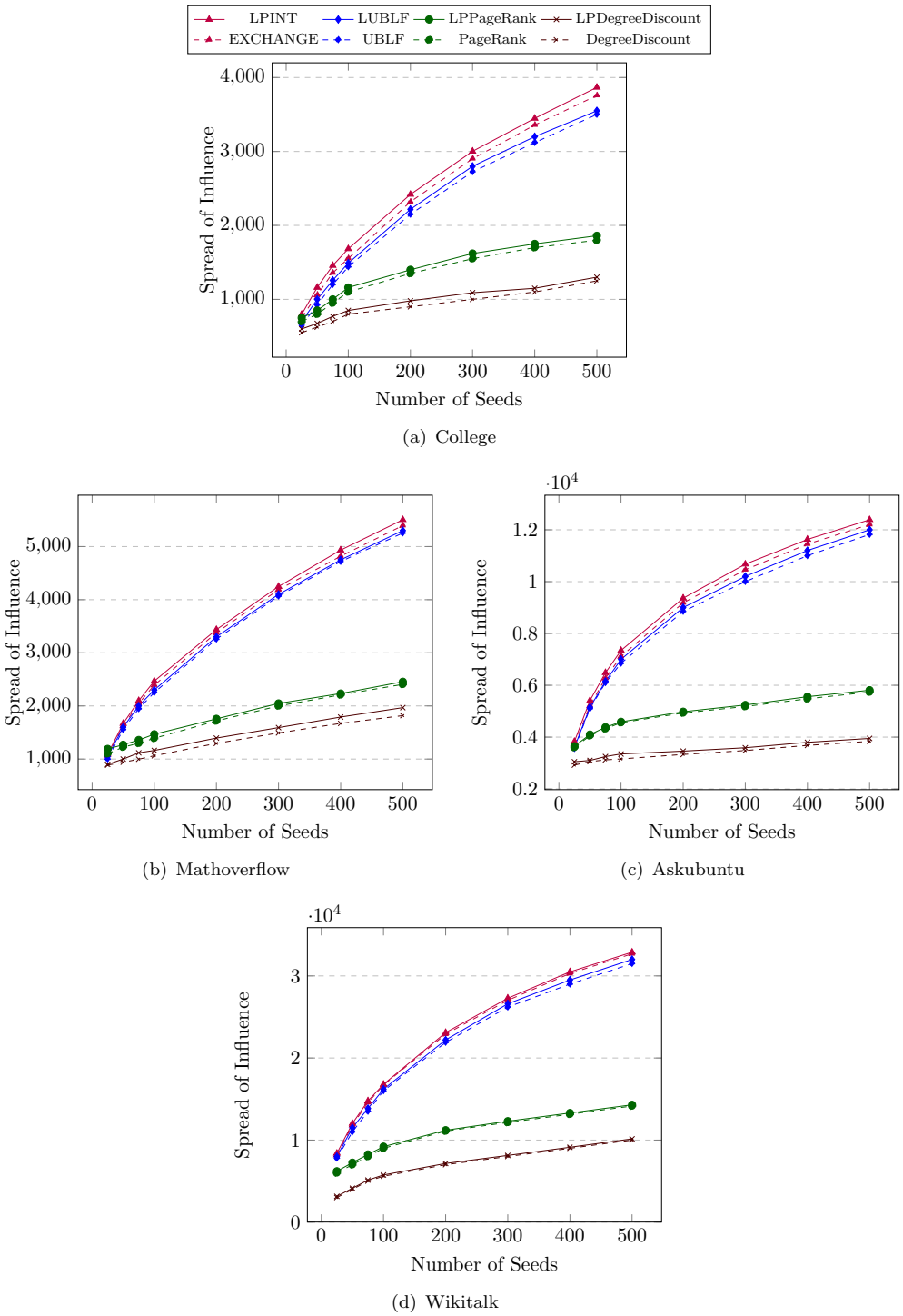
3.4.3 Quality Metric for Influence spread

To demonstrate the effectiveness of our proposed solution, we find the seed sets produced by all the strategies described above for every window shift. When a seed set S_{t+1} is returned by an algorithm, we evaluate the influence spread under the IC model by 10,000

rounds of Monte-Carlo simulation on the current snapshot of the graph, i.e., $\hat{G}^{t+1}$. Note that we assume that the graph is highly dynamic, so it evolves quickly after the selection of seed sets. Finally, we take the *average influence spread* over all windows as the quality metric to compare different approaches.

3.4.4 Experimental Settings

For the information diffusion process, we use the Independent Cascade model. In this model, most of the literature [67] have done experiments using a small propagation probability of $p = 0.01$. Larger p values such as $p = 0.1$ are not considered due to insensitivity to different algorithms. In our experiments, we use $p = 0.01$ as the value of propagation probability. In the first step, the dataset is divided into different snapshots. We have divided the dataset according to a fixed timeframe, which is different for the different datasets. For all datasets, we have divided them into $T = 25$ snapshots. Now, we have $[G^1, G^2, \ldots, G^{25}]$ snapshots of the graph. We use $N = 10$ as the size of the window; it means there are 10 snapshots in each window. Here, we use the first 10 snapshots for training the ctRBM using Algorithm 1 for link prediction task and then predict the newly arrived edges in newly added snapshot after window shift of 1 snapshot and compare it with original edges at latest snapshot for testing and repeat the same process for each window shift. For influence maximization, we find the seed node-set by applying Algorithm 3 on the predicted snapshot graph and use this seed set for evaluating the influence spread on the newly added snapshot graph. In our experiments, we are varying the size of seed set k represented on the x-axis, and the y-axis represents the information spread in the target snapshot of the graph. In our experiments, we predict 5 percent links, which may appear in the upcoming snapshot of the graph. We use the average of 10,000 rounds of Monte-Carlo simulations to estimate the actual influence spread and thereby assess the seed set found by the algorithms.



(a) College

(b) Mathoverflow

(c) Askubuntu

(d) Wikitalk

FIGURE 3.6: Number of Seeds versus Spread of Influence for Different Static IM Technique with and without Link Prediction on the Snapshot of Different Dynamic Networks

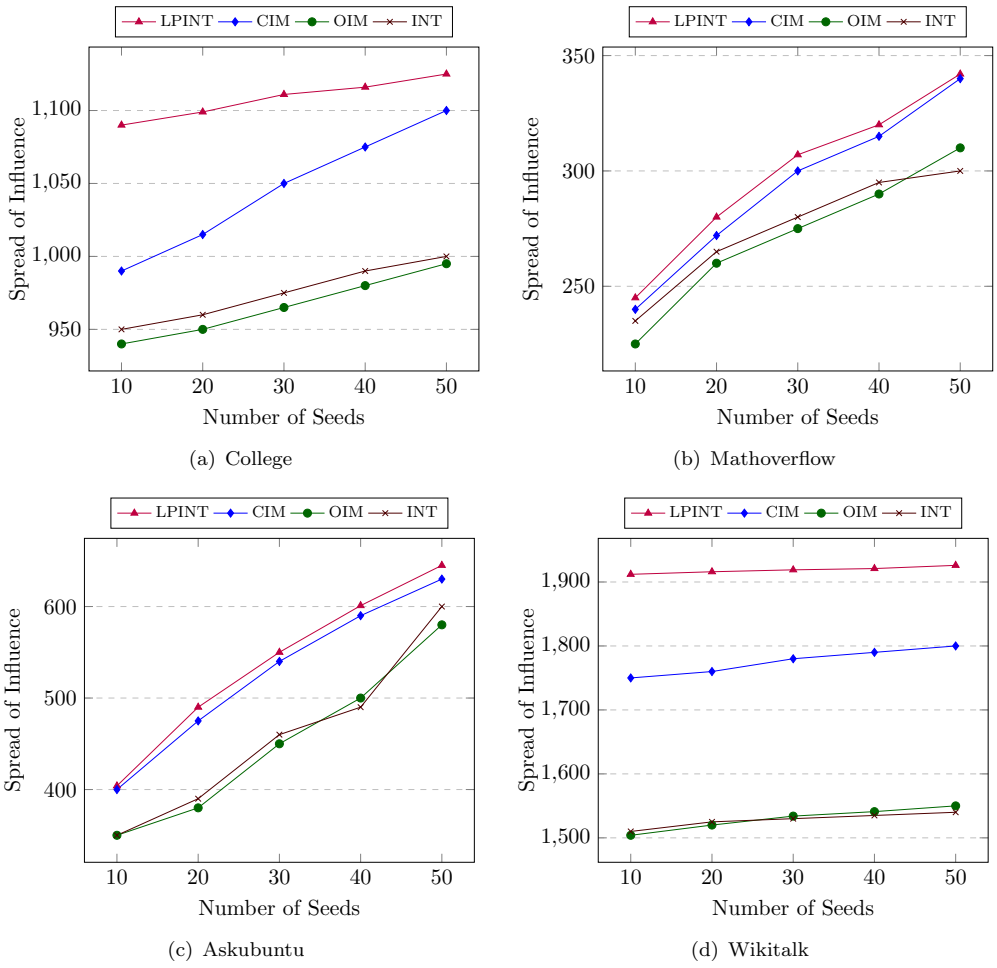


FIGURE 3.7: Comparison of Different Online IM Techniques with LPINT on the Snapshot of Different Dynamic Networks by Showing Number of Seeds versus Spread of Influence.

3.5 Results and Discussions

3.5.1 Comparing Different Approaches with and without using Link Prediction

The results in Figure 3.6 shows the influence spread for different datasets against varying seed nodes with and without link prediction technique for the dynamic network. The result shown here is valuated for a single snapshot of the network. The final number of influenced nodes are the average of repetition of the influence maximization process. The seed set size value varies as $\{25, 50, 75, 100, 200, 300, 400, 500\}$. Here, we compared

different static IM techniques applied with and without link prediction techniques for IM in dynamic social networks. Here we use the link prediction technique for predicting the next snapshot. Then, we find the seed nodes in the predicted snapshot using different static IM techniques. Further, we find the influence spread using those seed nodes in the actual snapshot. In this way, we get better results in terms of influence spread for all the considered algorithms when link prediction technique used. We see that the EXCHANGE algorithm gives an improvement in influence spread as compared with other baseline IM algorithms, and it becomes even better when used with link prediction as LPINT.

3.5.2 Comparing Dynamic Approaches with our Proposed LPINT Algorithm

We compare our proposed LPINT model for influence maximization with existing techniques for influence maximization in dynamic networks. The results are presented in graphs shown in Figure 3.7, here we have shown the influence spread by varying the seed set size k on the target snapshot of the graph. We can see that the proposed LPINT algorithm outperforms other considered algorithms in terms of influence spread.

3.5.3 Comparison of Average Running Time for Influence Spread

In table 3.1, we can see the average running time of influence spread for a snapshot graph using benchmark algorithms and the LPINT method. Notice that LPINT performs significantly better in terms of the time required for influence spread. The reason for this faster influence spread is the selection of better seed nodes. If we choose more effective seed nodes, it takes less time for influence spread as it requires fewer iterations to complete the influence spread process using the Independent Cascade model. This lower time for influence spread in LPINT again confirms its efficiency over benchmark methods.

TABLE 3.1: Average Running Time for Influence Spread

Datasets	Methods			
	INT	OIM	CIM	LPINT
college	46 ms	40 ms	32 ms	21 ms
mathoverflow	38 ms	33 ms	28 ms	24 ms
ask-ubuntu	1.1 s	59 ms	53 ms	40 ms
wiki-talk	1.6 s	1.2 s	1.0 s	59 ms

3.5.4 Insightful Discussion

In our proposed influence maximization algorithm for the dynamic social network, we show the improvement in results in terms of influence spread experimentally and theoretically. In our proposed work, once the behaviour of nodes for making the new links are learned, the prediction of the upcoming snapshot becomes efficient and effective. At each snapshot, there is no need to explore all the nodes to find suitable seed nodes. Efficient seed nodes reduce the number of iteration in the IC model for influence spread and hence take less time for information spread as compared to other considered baseline algorithms.

The limitation of our proposed method includes the overhead of prediction of the upcoming snapshot; however, with the increase of time system learns for efficient prediction. Here, we have not considered the situation where any node behaves randomly, although it is also not considered by the baseline algorithms. Our proposed model can also be implemented with other diffusion models, which is not explored here.

3.6 Conclusions

A link prediction based influential node tracking method is presented in this chapter to find seed nodes for information spread in the dynamic social network. We use the ctRBM based deep learning technique for link prediction to predict the next snapshot of the graph. We

then find the seed set in the predicted snapshot using the EXCHANGE algorithm. This seed set is used for actual influence spread in the real snapshot of the graph. This method improves the influence spread in terms of the number of influenced nodes in highly dynamic social networks. Extensive experiments on datasets obtained from four real social networks demonstrate that our method outperforms the baselines in terms of influence coverage and influence spread time.