Link prediction approach to recommender systems

TANGIRALA, Jaya Lakshmi <http://orcid.org/0000-0003-0183-4093> and BHAVANI, S. Durga

Link Prediction Approach to Recommender Systems

T. Jaya Lakshmi T[1*] and S. Durga Bhavani[2]

[1*]Department of Computer Science and Engineering, Data Science Research Lab, SRM University, Guntur, 522502, Andhra Pradesh, India.
[2]School of Computer and Information Sciences, University of Hyderabad, Hyderabad, 500002, Telangana, India.

*Corresponding author(s). E-mail(s): jaya.phd.hcu@gmail.com;
Contributing authors: sdbcs@uohyd.ac.in;

Abstract

The problem of recommender system is very popular with myriad available solutions. Recommender systems recommend items to users and help them in narrowing their search from huge amount of options available to the user. In this work, a novel approach for the recommendation problem is proposed by incorporating techniques from the link prediction problem in social networks. The proposed approach models the typical user-item information as a bipartite network, and predicts future links using link prediction measures, in which link prediction would actually mean recommending an item to a user. The standard recommender system methods suffer from the problems of sparsity and scalability. Since link prediction measures involve computations pertaining to local neighborhoods in the network, this approach would lead to a scalable solution to recommendation. In this work, we present top $\boldsymbol{k}$ links that are predicted by link prediction measures as recommendations to the users. Our work initially applies different existing link prediction measures to the recommendation problem by making suitable adaptations. The prime contribution of this work is to propose a recommendation framework routed from link prediction problem in social networks, that effectively utilizes probabilistic measures of link prediction and embed temporal data accessible on existing links. The proposed approach is evaluated on one movie-rating dataset of MovieLens, two product-rating datasets of Epinions & Amazon and one hotel-rating dataset of TripAdvisor. Results show that the link prediction measures based on temporal probabilistic information prove to be more effective in improving the quality of recommendation. Especially, Temporal cooccurrence probability measure improves the area under ROC curve (AUROC) by 10% for MovieLens, 23% for Epinions, 17% for TripAdvisor,

9% for Amazon over standard item-based collaborative filtering method. Similar improved performance is observed in terms of area under Precision-Recall curve (AUPR) as well as Normalized Rank-Score.

Keywords: Recommender Systems, Link prediction, Bipartite Graph, Temporal probabilistic measures.

1 Introduction

Many e-commerce websites provide a wide range of products to the users. The users commonly have different needs and tastes based on which they buy the products. Providing the most appropriate products to the users make the buying process efficient and improves the user satisfaction. The enhanced user satisfaction keeps the user loyal to the website and improves the sales and thus profits to the retailers. Therefore, more retailers start recommending the products to the users which needs efficient analysis of user interests in products. E-commerce leaders like Amazon and Netflix use recommender systems to recommend products to the users. The standard recommendation algorithms like content-based filtering and collaborative filtering, model the ratings given by users to items as a matrix and predict the ratings of the customers to unrated items based on user/item similarity.

Recommender systems recommend items to users. Items include products and services such as movies, music, books, web pages, news, jokes and restaurants. The recommendation process utilizes data including user demographics, product descriptions and the previous actions of users on items like buying, rating, and watching. The information can be acquired explicitly by collecting ratings given by users on items or implicitly by monitoring user's behavior such as songs listened in music websites, news/movies watched in news/movies websites, items bought in e-commerce websites or books read in book-listing websites in the past. Usage of intelligent recommendation systems improved the revenue of Amazon by 35%, caused a business growth of 24% for BestBuy, increased 75% of views on Netflix and 60% views on YouTube. [1]. Therefore, building a personalized recommendation system has a profound significance not only in commercial arena, but also in the fields like health care, news, food, academia and so many. Each domain needs to consider different features.

Recommender system is a typical application of link prediction in heterogeneous networks. Link Prediction problem infers future links in a network represented as graph where nodes are users and edges denote interactions between users. In the context of recommender systems, the nodes may be of two types: items and users. A transaction of a user buying an item can be shown as an edge from user node to item node. Recommendation problem can be viewed as a task of selecting unobserved links for each user node, and thus can be modeled as a link prediction problem.

In this work, we have applied various link prediction measures on the bipartite network in the context of recommender systems and verified the efficacy of those measures. The contributions made in this paper are as follows.

- Modelled the recommendation task as a link prediction problem in a bipartite network.
- Computed recommendations on rating data using the following link prediction measures:
  - Neighborhood based: Common Neighbors, Jaccard-coefficient, Adamic-Adar and Preferential Attachment
  - Path-based measure: Katz.
  - Random-walk-based measure: PropFLow.
- Extended temporal link prediction measures of Time-score, Link-score, and T_Flow for recommendation problem.
- Extended existing probabilistic measures called co-occurrence probability measure and temporal co-occurrence probability measure to make them suitable for recommendation problem.
- Evaluated the efficacy of all measures on four benchmark datasets of MovieLens, Epinions, TripAdvisor and Amazon datasets using three metrics: Area under ROC curve (AUROC), Area under Precision-Recall curve (AUPR) and Normalized rank score (NRS).

2 Problem Statement

Schafer et al [2] define the problem of recommender systems as follows: Given a set of m users $U=\{u_1, u_2, \ldots u_m\}$, a set of n items/products $I=\{I_1, I_2, \ldots, I_n\}$ and the ratings R, where R_{ij} represents the rating given by user u_i to item I_j, the task of recommender systems is to recommend a new item I_j to a user u_i which the user u_i did not consume yet.

For example, consider the matrix given in Fig.1, where rows correspond to the users and the columns denote items(movies/TV shows). The matrix entry R_{ij} represents the ratings given by user u_i on item I_j. The main task of the recommender systems is to predict the unrated entries in the rating matrix.

		Movies					
		Movie1	**Movie2**	**Movie3**	**Movie4**	**Movie5**	**Movie6**
Users	**User1**		**5**				
	User2	5		2		1	
	User3		5	5	5	5	5
	User4	2	5		3		
	User5			5	3		
	User6	1	5				1
	User7	2		5		5	

Fig. 1 Sample Rating Matrix

Recommender systems are natural examples of weighted bipartite networks. A **Bipartite network** contains exactly two types of nodes and single type of edges existing between different types of nodes. A bipartite network is defined as $G=(V_1 \cup V_2,\ E)$, where V_1 and V_2 are sets of two types of nodes, E represents the set of edges between nodes of type V_1 and V_2. Fig.2 depicts a sample scenario of users buying items in e-commerce sites such as Amazon, modeled as a bipartite network. The next section discusses a few existing works on recommendation problem.

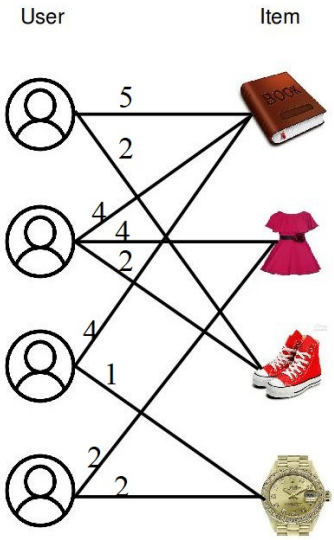

Fig. 2 An example of user-item Bipartite Network with edges denoting rating given by user on item.

3 Related Literature

The studies on recommender system in the literature are discussed from algorithmic as well as domain perspective in the next two sections. Fig. 3 gives a taxonomy of the literature.

3.1 Domain Perspective

Product recommendation is the mostly explored domain. E-commerce sites like Amazon, Flipkart, Ebay etc use various collaborative filtering techniques for recommending products to their customers. Video recommendations like movies, TV shows and web series are increasing exponentially by companies like Netflix, YouTube. Netflix has launched a competition with 1Million dollars prize money for improving 10% of RMSE in the movie recommendation, by releasing a 100 million customer ratings [3]. A news recommendation system has been implemented in [4] using google news, by constructing a Bayesian network for identifying the user interests and then applying content

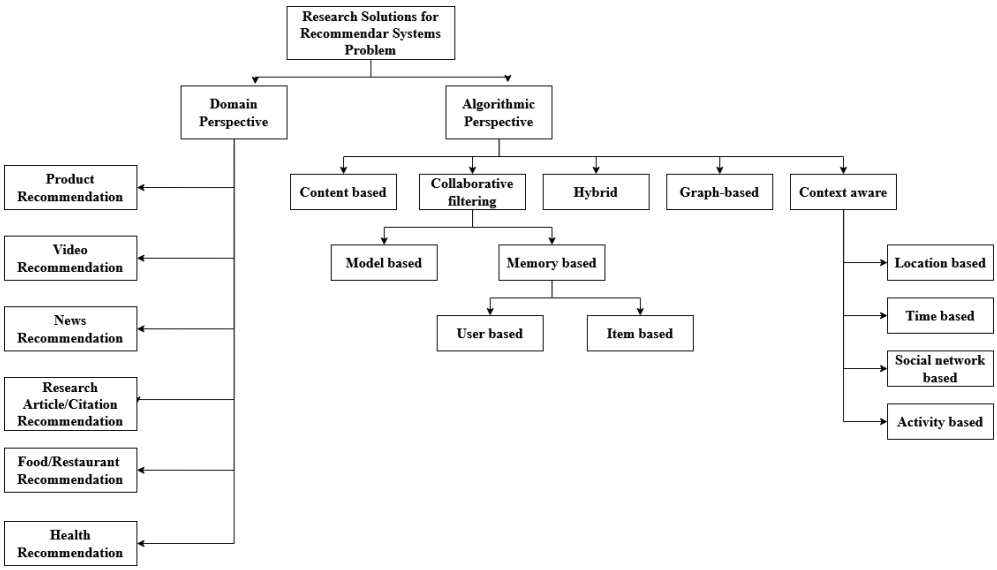

Fig. 3 Taxonomy of recommender systems

based as well as collaborative filtering and hybrid techniques. Research article recommendation, food recommendation and health recommendation are other domains among many.

Valdez et al. integrate recommender systems into individual medical records to help patients improve their autonomy [5]. The authors have collected health related information from the patients' records and have performed text processing for extracting features and evaluated using metrics available in Information Extraction domain [5].

All these domains need different pre-processing techniques, algorithms, and evaluation metrics.

3.2 Algorithmic perspective

Recommender systems is seeing an explosive growth in the literature with many novel algorithms being proposed almost every day. Content-based recommendation, Collaborative Filtering (CF) and Hybrid approach are some of the popular approaches that solve the problem of recommender systems. Context aware recommender systems predict recommendations based on user context. Graph based recommender systems is main focus of this work. In graph based recommender system, the purchases are modeled as bipartite graph and various graph traversal techniques are applied to generate a set of co-purchased items/users.

3.2.1 Content-based recommendation

Content-based recommendation uses item descriptions and constructs user profiles which contain the information about user preferences [6]. These user preferences may

include a genre, director and actor for a movie, an author for book etc. The recommendation of an item to the user is then based on the similarity between the item description and the user profile. This method has the benefit of recommending items to users that have never been rated by them [7]. Content-based recommendation systems require complete descriptions of items and detailed user profiles. This is one of the main limitations of such systems [7].

Content-based recommender systems are based on three kinds of content: Explicit, Implicit and Model-based. Explicit content is collected in the form of profile information of the users through questionnaires that ask questions about their preferences for the items. Implicit content builds the user profiles implicitly by searching for similarities in liked and disliked item descriptions by the user. Model-based content constructs user profiles based on a learning method, using item descriptions as input features to be given to a supervised learning algorithm, and producing user ratings of items as output.

The item descriptions are often textual information contained in documents, web sites and news articles. User profiles are modeled as weighted vectors of these item descriptions. The advantage of this method is the ability to recommend newly introduced items to users [7]. Content-based recommendation systems require the complete descriptions of items and detailed user profiles. This is the main limitation of such systems. Privacy issues such as users dislike to share their preferences with others is another limitation of content-based systems.

3.2.2 Collaborative filtering

Collaborative filtering systems, however, do not require such difficult to come by, well-structured item descriptions. Instead, they are based on users' preferences for items, which can carry a more general meaning than is contained in an item description. Indeed, viewers generally select a film to watch based upon more criteria other than only its genre, director or actors.

Collaborative filtering (CF) techniques depend only on the user's past behavior and provide personalized recommendation of items to users [8]. CF techniques take a rating matrix as input where the rows of the matrix correspond to users, the columns correspond to items and the cells correspond to the rating given by the user to the item [9]. The rating by a user to an item represents the degree of preference of the user towards the item. The major advantage of the CF methods is they are domain independent. CF methods are classified into Model-based CF [10] and Memory-based CF. Model based techniques model ratings by describing both items and users on some factors induced by matrix factorization strategies. Data sparsity is a serious problem of CF methods.

Model based approaches such as dimensionality reduction techniques, Principal Component Analysis (PCA), Singular Value Decomposition (SVD) are some of the popular techniques that address the problem of sparsity. However, when certain users or items are ignored, useful information for recommendations may be lost causing reduction in the quality of recommendation [11]. The limitation of collaborative filtering systems is the cold start problem. i.e., these methods can not recommend new items to the existing users as there is no past buying history for the item. Similarly,

it is difficult to recommend items to a new user without knowing the user's interests in the form of ratings.

Memory based methods compute a set of nearest neighbors for users as well as items using similarity measures like Pearson coefficient, cosine distance and Manhattan distance. Memory based CF techniques are further classified into user based and item based [12].

User-based CF[13]

User based methods compute the similarity between the users based on their ratings on items [14]. This creates a notion of user neighborhoods. These methods associate a set of nearest neighbors with each user and then predict the user's rating for unrated items utilizing the information of the neighbor's rating for that item.

Item-based CF [15]

Similarly, item neighborhood depicts the number of users who rate the same items [15]. The item rating for a given user can then be predicted based upon the ratings given in their user neighborhood and the item neighborhood. These methods associate an item with a set of similar neighbors, and then predict the user's rating for an item using the ratings given by the user for the nearest neighbors of the target item.

3.2.3 Hybrid approach

Hybrid approach uses both types of information, collaborative and content based. Content boosted CF algorithm [16], uses the item profile information to recommend the items to new users.

3.2.4 Context aware recommendation system

Context aware recommendation system is able to label each user action with an appropriate context and effectively tailor the system output to the user in that given context [17]. In [18], the authors proposed a recommendation of an activity to a social network user based on demographic information. Kefalas et al. have represented their data as a k-partite network modeled as a tensor and proposed novel algorithms. In the same way, in [19], a citation recommendation is made to researchers also considering tags and time. Standard recommender system algorithms cannot address these issues.

3.2.5 Graph based Recommender Systems

Li et al. map transactions to a bipartite user–item interaction graph and thus converting recommendation into the link prediction problem. They propose a kernel-based recommendation approach which generate random walk between user–item pair and define similarities between user–item pairs based on the random walks. Li et al. use a kernel-based approach that indirectly examines customers and items related to user-item pair to foresee the existence of an edge between them. The paper uses a set of nodes representing two types of nodes users(U) and items(I) and edges denoting transactions by the user regarding the items. The graph kernel is defined on the user-item pairs context [20].

Zhang et al. propose a model for music recommendation [21]. The authors represent the recommendation data as a bipartite graph with user and item nodes and the weight of the link is represented as a complex number depicting the dual preference of users in the form of like and dislike and improve the similarity of the users.

In graph-based recommender system, the task of recommendation is equivalent to predicting a link between user-item based on graphical analysis. Link prediction in graphs is commonly a classification task, that predicts existence of a link in future [22]. But the problem of recommender systems is modeled as a regression task, which predicts the rating of a link. The predicted links with high ratings are generally recommended to users. The standard techniques of recommendation systems take the ratings matrix as input and predicts the empty cells in the matrix. This approach severely suffer with the sparsity in the matrix. Ranking-oriented collaborative filtering approach is more meaningful compared to the rating based approach because, the recommendation is a ranking task recommending top-k ranked items to a user [23]. In some applications like recommending web pages, rating information may not be available. Graph-based recommendation can efficiently utilize the heterogeneous information available in the networks by expanding the neighborhoods and can compute the proximity between the users and items [12].

A probabilistic measure for predicting future links in bipartite networks is given in [24]. Several link prediction measures in various kinds of networks have been summarized in [25].

Our aim is to evaluate the efficacy of these measures in the context of recommender systems by modeling them as bipartite graphs. In this paper, we have applied various existing link prediction measures on bipartite graphs in order to build a recommender system. However, the link prediction approach does not give the actual rating, but gives the top k-item recommendations for a user. For experimentation, we have chosen the bench-mark 'Movielens' dataset. Next section gives an overview of application of link prediction measures on bipartite graphs.

4 Link Prediction in bipartite graph

Graph-based recommendation algorithms compute recommendations on a bipartite graph where nodes of the graph represent users and items and a link forms between a user node and an item node when the user buys/rates the item. Once the transactions are modeled as a graph, all the standard link prediction methods defined in [22],[26] can be applied directly on the graph to predict the future links [27]. All the measures used in [22] are based on neighbors and paths. Especially, common neighbors, which lie on paths of length two play an important role. All the measures are extended to heterogeneous environment in [25] with a mention of bipartite networks as special case.

We follow similar notation as in [25] in this paper, which is given below.

- (u, p) is a pair of user and product nodes without an edge between them.
- k represents hop-distance between two nodes and t denotes time.
- $\Gamma_k(x)$ is the set of *k-hop* neighbors of node x. By *k-hop* neighbors, we mean all the nodes that lie on the paths from x upto a distance of k. $\Gamma_1(x)$ refers to the set of all nodes connected by an edge to x generally written as $\Gamma(x)$.

- $\Gamma(u) \cap \Gamma(p)$ denotes the set of common neighbors between the nodes u and p. In bipartite graph, paths of even length do not exist between u and p. This leads to the set of common neighbors, $\Gamma(u) \cap \Gamma(p)$ to be empty. For instance, in Fig.4, $|\Gamma(u) \cap \Gamma(p)| = 0$ as the number of paths of length 2 between u and p is 0. On the other hand, $|\Gamma_2(u) \cap \Gamma_2(p)|$ = Number of common nodes on paths of length 3 between u and p which is 2. Therefore, it is needed to expand the neighborhood in bipartite environment. A hop distance 2 is the minimum to have some common neighbors between a pair of nodes in bipartite networks.

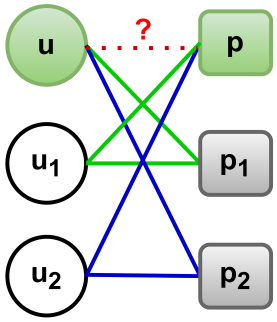

Fig. 4 Demonstration of need of expanding neighbourhood in bipartite graphs

- $\Gamma_k(u) \cap \Gamma_k(p)$ contains all the common neighbors within k-hop distance between nodes u and p.
- $P_l(u, p)$ denotes the set of paths of length l connecting u and p.

4.1 Baseline link prediction measures in bipartite graph

In recommendation systems modeled as bipartite graph, the task is to recommend products to users. The product nodes and user nodes are connected with odd length paths. Even length paths connect the nodes of same type, which is not meaningful in this scenario. Keeping this in mind, the baseline link prediction measures for bipartite graphs are defined as given below. We use suffix B with all link prediction measures to indicate they are used in context of bipartite network.

- **Common Neighbors (CN)** : The common neighbor measure in bipartite graph is given as follows:

$$CN_B(u, p) = |\Gamma_2(u) \cap \Gamma_2(p)| \tag{1}$$

- **Jaccard Coefficient (JC)** : Jaccard Coefficient is the normalized CN measure

$$JC_B(u, p) = \frac{|\Gamma_2(u) \cap \Gamma_2(p)|}{|\Gamma_2(u) \cup \Gamma_2(p)|} \tag{2}$$

- **Adamic Adar (AA)** : This measure gives importance to the common neighbors with low degree. The following definition for bipartite networks has been hinted at [28].

$$AA_B(u,p) = \sum_{z \in \Gamma_2(u) \cap \Gamma_2(p)} \frac{1}{log(|\Gamma_2(z)|)} \tag{3}$$

- **Preferential Attachment (PA)** : This measure does not change in bipartite environment because the measure is concerned only about the degree of the node whatever the type of the node may be.

$$PA_B(u,p) = |\Gamma_1(u)| * |\Gamma_1(p)| \tag{4}$$

- **Katz (KZ)** : This measure is based on the total number of paths between u and p bounded by a limit penalized by path length l.

$$KZ_B(u,p) = \sum_{l=2}^{L} \beta^l |P_l(u,p)^l| \tag{5}$$

where L is the threshold of path length between nodes u and p, β is damping factor. It is common practice to consider the value of β between 0.005 to 0.5.

- **PropFlow (PF)** : This is a random-walk beginning at node u and ending at p within l steps. This random-walk from node u to p terminates either when it reaches p or revisits any node. $PF_B(u,p)$ is the probability of information flow from u to p based on random transmission along all paths defined recursively as follows.

$$PF_B(u,p) = \sum_{l=2}^{L} \sum_{path \in P_l(u,p)} \sum_{\forall (z_1,z_2) \in path} PF_B(z_1,z_2) \tag{6}$$

where L is the threshold of path length between nodes u and p.

$$PF_B(z_1,z_2) = PF_B(a,z_1) * \frac{w(z_1,z_2)}{\sum_{z \in \Gamma(z_1)} w(z_1,z)} \tag{7}$$

with a as previous node of z_1 in the random-walk, $PF(a,z_1)$=1 if a is the starting node. In bipartite network, l is odd.

$$PF_B(u,p) = \sum_{l=3}^{L} \sum_{p \in P_l(u,p)} \sum_{\forall (z_1,z_2) \in p} PF_B(z_1,z_2) \tag{8}$$

where $PF_B(z_1,z_2)$ is as defined in Eq.7.

4.2 Temporal measures for link prediction in bipartite graph

In temporal measures, for each edge e, time of formation of e is available. We denote this as $t(e)$. We extend a few temporal link prediction measures to bipartite networks as follows.

4.2.1 Time-Score(TS_B)

We extend Time-Score measure [29] to bipartite environment as follows:

$$TS_B(u,p) = \sum_{path \in P_3(u,p)} \frac{w(path) * \beta^{r(path)}}{|latest(path) - oldest(path)| + 1} \tag{9}$$

where $w(path)$ is equal to the harmonic mean of edge weights of edges in $path$, β is a damping factor (0¡β¡1), $latest(\text{path}) = \max_{e\ on\ P_3}(t(e))$, $oldest(\text{path}) = \min_{e\ on\ P_3}(t(e))$ and r is a recency factor, defined as $r(\text{path}) = current\_time - latest(\text{path})$. It is common practice to consider the value of β between 0.005 to 0.5.

4.2.2 Link-Score(LS_B)

Choudhary et al. extend the Time-score measure to obtain a path based measure called *Link-score* [30]. To obtain the *Link-score* between a pair of nodes u and p which are not directly connected, the authors define a Time Path Index (TPI) on each path p between the nodes u and p. TPI evaluates path weight based on time stamps of links involved in a path. Link-score is the sum of TPI of each path between the nodes u and p.

We extend *Link-score* to bipartite network by considering *paths* of odd length between two nodes instead of paths containing any length. The modified definitions of TPI and *Link-Score* are given in equation 10.

$$TPI_{path} = \frac{w(path) * \beta^{current\_time - avg(path)}}{|current\_time - latest(path)| + 1} \tag{10}$$

where $avg(path)$ is the average active year, which is the average of years of recent interaction of edges on odd length path and all others are as defined in equation 9.

$$LS_B(u,p) = \sum_{l=3}^{L} \frac{Avg(TPI_{P_l(u,p)})}{l-1} \tag{11}$$

where L is the maximum length of paths between nodes i and j.

4.2.3 T_Flow(TF_B)

T_Flow [31] is a random-walk based measure, which is an extension of PropFlow measure defined in [26]. Munasinghe et.al [31] define T_Flow that computes the information flow between a pair of nodes u and p through all random-walks starting from node u to node p including link weights as well as activeness of links by giving more weight to recently formed links recursively and take the summation.

We extend T_Flow measure to bipartite networks as follows:

$$TF_B(u,p) = \sum_{l=3}^{L} \sum_{path \in P_l(u,p)} \sum_{\forall (z_1,z_2) \in path} TF_B(z_1,z_2) * (\beta)^{r(path)} \tag{12}$$

Note that, $TF_B(u,p)$ is temporal extension of equation 6. L is the threshold of path length between nodes u and p. β is damping factor generally chosen between 0.005 and 0.5. r is a recency factor, defined as $r(\text{path}) = current\_time - \max_{e\ on\ path}(t(e))$.

If $(u,p) \in E$, then $TF_B(u,p)$ is given by

$$TF_B(u,p) = TF_B(a,u) * \frac{w(u,p)}{\sum_{z \in \Gamma(u)} w(u,z)} * (\beta)^{r(path)} \tag{13}$$

where $r(path) = t_u\text{-}t_p$, t_u being the time stamp of the link when the random walk visits the node u and t_p is the time stamp of the link when the random walk visits node p and β is damping factor.

4.3 Probabilistic measure for link prediction in bipartite graph

The Probabilistic Graphical Model (PGM) represents the structure of the graph in a natural way by considering the nodes of graph as random variables and edges as dependencies between them. By representing a graph as a PGM, the problem of link prediction, which is calculation of the probability of link formation between two nodes $u,\ p$ is translated to computing the joint probability of the corresponding random variables $U,\ P$.

Recommender systems represented as bipartite networks are large in size. Therefore, finding the joint probabilities of link formation is intractable. But the links in the graphs are also sparse, with nodes generally directly connected to only a few other nodes. For example, in the e-commerce sites, out of lakhs of items, a user buys only a few items. This property allows the PGM distribution to be represented tractably. The model of this framework is simple to understand. Inference between two random variables is same as finding the joint probability between those two random variables in PGM. Many algorithms are available for computation of joint probability between variables, given evidence on others. These inference algorithms work directly on the graph structure and are generally faster than computing the joint distribution explicitly. With all these advantages, link prediction in PGM is more effective.

In most of the cases, the pair-wise interactions of entities are available in the event logs. For example, a user buying/rating three items say p_1, p_2 and p_3 is available in the corresponding transaction database. In order to model the unknown distribution of co-occurrence probabilities, events available in the event logs can be used as constraints to build a model for the unknown distribution. Probabilistic Graphical Models efficiently utilize this higher order topological information and thus are efficient in link prediction task [32].

The probabilistic model helps in estimating the joint probability distribution over the variables of the model [33]. That means, a probabilistic model represents the probability distribution over all possible deterministic states of the model. This facilitates the inference of marginal probabilities of the individual variables of the model.

Wang et al. [32] are among first researchers who modeled the problem of link prediction using MRFs. Kashima et al. [34] propose a probabilistic model of network evolution and apply it for predicting future links. The authors show that by intelligently selecting the parameters in an evolution model, the problem of network evolution reduces to the problem of link prediction. Clauset et al. [35] propose a probabilistic model based on hierarchical structure of the network. In hierarchical structure, the vertices are divided into groups that further subdivide into groups of groups and so on. The model infers hierarchical structure from network data and can be used for prediction of missing links. The learning task uses the observed network data and infers the most likely hierarchical structure through statistical inference.

The works of Kashima [34] and Clauset [35] are global models and are not scalable for large networks. The method of Wang et. al. [32] uses local probabilistic information of graphs for link prediction and we adopt their method in our work. The following section explains the algorithm for link prediction using MRF proposed by Wang et al. [32]

5 Proposed approach: Link Prediction in bipartite networks induced as PGM

Wang et al. [32] propose a measure called Co-occurrence Probability (COP) to be computed between a pair of nodes without an edge between them. The measure has been extended to temporal networks in [36]. Heterogeneous version of the measure is defined in [24]. In this paper, we have extended COP measure to bipartite environment to make it suitable for recommender systems, named it as B-COP of a missing link (u, p) in the lines of COP. The procedure for computing B-COP(u, p) has three steps.

- Choose a few set of nodes which may play a significant role in future link formation. We call this set as Central Neighborhood Set in Bipartite graph (BCNS). Computation of BCNS is explained in section 5.1.
- Compute a Markov Random Field (MRF) with the chosen nodes in the first step. Construction of MRF requires extracting cliques. As the network is bipartite, we consider the complete bipartite subgraph containing the nodes in BCNS as clique and name it as B-clique. Computation of B-cliques is the main task of this procedure. The procedure of computation of B-cliques is given in section 5.2.
- Infer joint probability between the nodes u, p from the constructed MRF in step-2. This is explained in section.5.4.

The procedure of computing B-COP of a missing link is summarised in Algorithm 1.

Algorithm 1 B-COP measure for Link Prediction in bipartite graphs

Input: G=(V, E), where
V=$U \cup P$ is the set of user nodes as well as product nodes,
E is the set of links from nodes of U to P.
A pair of nodes:$(u, p) \in VXV - E$
B-*cliq*: A set of B-cliques extracted from even logs of user-item bipartite graph using the Algorithm. 3
Output: B-COP(u, p)
Step 1: Compute **BCNS**, the central neighborhood set of (u, p) using the Algorithm 2.
Step 2: Form Markov Random Field with the nodes in BCNF as follows:
    Step 2.1: Extract B-cliques formed with the nodes in BCNS from B-*cliq*.
    Step 2.2: Compute clique potentials.
Step 3: Return B-COP(u, p) which is the joint probability of link (u, p) using junction tree algorithm.

5.1 Computation of Central Neighborhood Set in Bipartite graph(BCNS)

BCNS(u, p) is computed using a breadth first search based algorithm on bipartite graph as follows: All paths between u and p are obtained using breadth first search (BFS) algorithm. Then, the frequency score of each path is computed by summing the occurrence count of nodes on the path. Occurrence count of a node is the number of times the node appears in all paths. The paths are now ordered in the increasing order of length with equal paths being ordered in decreasing order of frequency score. The size of the central neighborhood set is further restricted by considering only top k nodes. The procedure of computing $BCNS(u, p)$ is described in the Algorithm 2.

The procedure of computation of BCNS for a toy example given in Fig.5 is shown in Fig. 6.

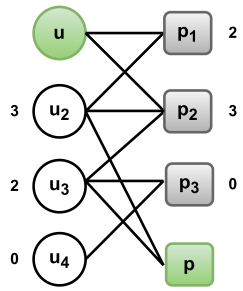

Fig. 5 A toy example for illustrating computation of BCNS between nodes u and p. Node weights are the occurrence counts.

path(p)	**frequency-score**(F_p)
$u - p_2 - u_2 - p$	$O_{p_2} + O_{u_2} = 6$
$u - p_1 - u_2 - p$	$O_{p_1} + O_{u_2} = 5$
$u - p_2 - u_3 - p$	$O_{p_2} + O_{u_3} = 5$
$u - p_1 - u_2 - p_2 - u_3 - p$	$O_{p_1} + O_{u_2} + O_{p_2} + O_{u_3} = 10$

Fig. 6 All paths between nodes u and p in Fig.5 and their frequency scores.

Algorithm 2 Bipartite Central Neighborhood Set$(G, u, p, l, maxSize)$

Input: G: a graph;
u: starting node;
p: ending node;
l:maximum path length;
$maxSize$:Central Neighborhood Set size threshold
Output: $BCNS$, Bipartite Central Neighborhood Set between u and p;
Step 1: Compute paths of length $\leq$ l between u and p.
Step 2: Find occurrence count O_k of each node k in paths between u and p.
Step 3: Compute $frequency\text{-}score$ F_p, of each path p as follows:

$$F_p = \sum_{k \in p} O_k$$

$frequency\text{-}score$ of a path is the sum of the occurrence counts of all nodes along the paths.
Step 4: Sort the paths in increasing order of path length and then in decreasing order of $frequency\text{-}score$. Let the ranked list of paths be P.
Step 5:
while size(BCNS) $\leq maxSize$ **do**
    Add nodes on path $p \in P$ to BCNS
end while
return BCNS

Table 6 shows all paths sorted in the increasing order of path length and frequency score, between the nodes u and p of the graph shown in Fig.5 along with their $frequency\text{-}scores$.

One can observe that all these paths are of odd length. For the bipartite graph in Fig.5, BCNS(u, p)=$\{u, p_2, u_2, p_1, p\}$, if $maxSize$ is taken as 5.

We adopt breadth first search starting from user node u to item node p to compute BCNS(u, p). Therefore, the complexity to compute BCNS(u, p) is $O(|V| + |E|)$.

5.2 Computation of B-cliques

Extraction of cliques from the probabilistic graph and the computation of clique potentials is the main part of this algorithm. A clique in a graph is a set of vertices with a link existing between every pair of vertices. But in recommender systems modelled as bipartite graph, links exist between a user node and product node, but not connecting two user nodes/product nodes. Therefore, we consider a complete bipartite sub-graph as clique. A sample B-clique is shown in Fig.7.

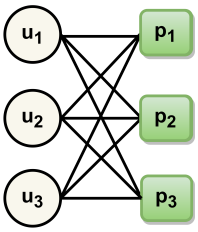

Fig. 7 An example B-clique.

In most of the cases, the information of homogeneous cliques containing all nodes of same type are available in the event logs. For example, in coauthorship networks, the group of authors who publish a paper together forms a homogeneous clique of author nodes and an author who publishes a paper in a conference forms a heterogeneous edge between the author node and conference node. But in the case of recommender systems, user cliques and item cliques are not readily available in the event logs. However, they can be extracted from the network with little effort. We propose an algorithm for extracting B-cliques shown in Algorithm 3. Since the number of users is huge in comparison to the set of items which is a much smaller set, the extraction of B-cliques can start from the item cliques.

The B-clique extraction algorithm first extracts all homogeneous cliques of items $Icliq$ and users $Ucliq$. A homogeneous user clique is formed with all the users who rate/buy the same item. Similarly, a homogeneous clique of items is formed with all the items a user rate/buy. To extract a B-clique, first consider an item i. For each user v who have rated i, compute the common items rated by user v. The union of U_p along with all the common items rated by U_p forms a B-clique. The process of extracting B-cliques for toy example in Fig.8 is illustrated below:

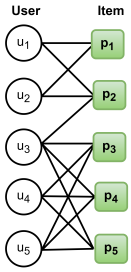

Fig. 8 User-Item bipartite graph

U=$\{u_1, u_2, u_3, u_4, u_5\}$ P=$\{p_1, p_2, p_3, p_4, p_5\}$
Item cliques :
Item clique corresponding to user $u_1 = \{p_1, p_2\}$
Item clique corresponding to user $u_2 = \{p_1, p_2\}$
Item clique corresponding to user $u_3 = \{p_2, p_3, p_4, p_5\}$
Item clique corresponding to user $u_4 = \{p_3, p_4, p_5\}$
Item clique corresponding to user $u_5 = \{p_3, p_4, p_5\}$
User cliques :
User clique corresponding to item $p_1 = \{u_1, u_2\}$
User clique corresponding to item $p_2 = \{u_1, u_2, u_3\}$
User clique corresponding to item $p_3 = \{u_3, u_4, u_5\}$
User clique corresponding to item $p_4 = \{u_3, u_4, u_5\}$
User clique corresponding to item $p_5 = \{u_3, u_4, u_5\}$

Given the number of users and items, the algorithm appears computationally expensive. Due to the common practise of keeping track of ratings and formation times of edges in the form of edge-list, the real computation time is proportional to the number of ratings. As the real world user-item networks are highly sparse, the number of ratings is significantly smaller in number.

Algorithm 3 Extraction of **B-Cliques** from user-item bipartite graph

Input: G=(V, E) where V=$U \cup P$, U is set of user nodes and P is the set of item/product nodes and $E \subseteq UXP$ represents weighted edges.
Output: $Bcliq$, set of maximal B-cliques of G.
Step 1 : Extract the set of all item cliques, $Item\_Cliq$ as follows:
$Item\_Cliq = \phi$
for each user $u \in U$ **do**
    P_u=ϕ
    for each $p \in P$ and $(u, p) \in E$ **do**
        P_u=$P_u \cup \{p\}$
    end for ▷ P_u is formed by taking all the items to which u gives a rating.
    Add P_u to $Item\_Cliq$
end for
Step 2 : Extract the set of all user cliques, $User\_Cliq$ as follows:
$User\_Cliq$=ϕ
for each item $p \in P$ **do**
    U_p=ϕ
    for each $u \in U$ and $(u, p) \in E$ **do**
        U_p=$U_p \cup \{u\}$
    end for ▷ U_p contains all users that have rated item p.
    Add U_p to $User\_Cliq$
end for
Step3 : Compute user-item bipartite cliques
for each item p in P **do**
    $J \leftarrow P$
    for each user v in U_p **do**
        $J \leftarrow J \cap P_v$
    end for
    J= $\bigcap_{v \in U_p} P_v$
    $Bcliq_p$=$J \cup P_i$
end for
$Bcliq$=$\bigcup_{p} Bcliq_p$
return $Bcliq$

5.3 Construction of local MRF

After computing the BCNS, the B-cliques containing only nodes of BCNS are extracted. This forms the clique graph of local MRF. MRF construction needs computation of clique potentials. The clique potential table of a B-clique is computed using the Non Derivable Itemsets [37]. A snapshot of aB-clique extracted from MovieLens 1M dataset used in our experimentation with its associated potential table is shown in Fig.10 and Table.11.

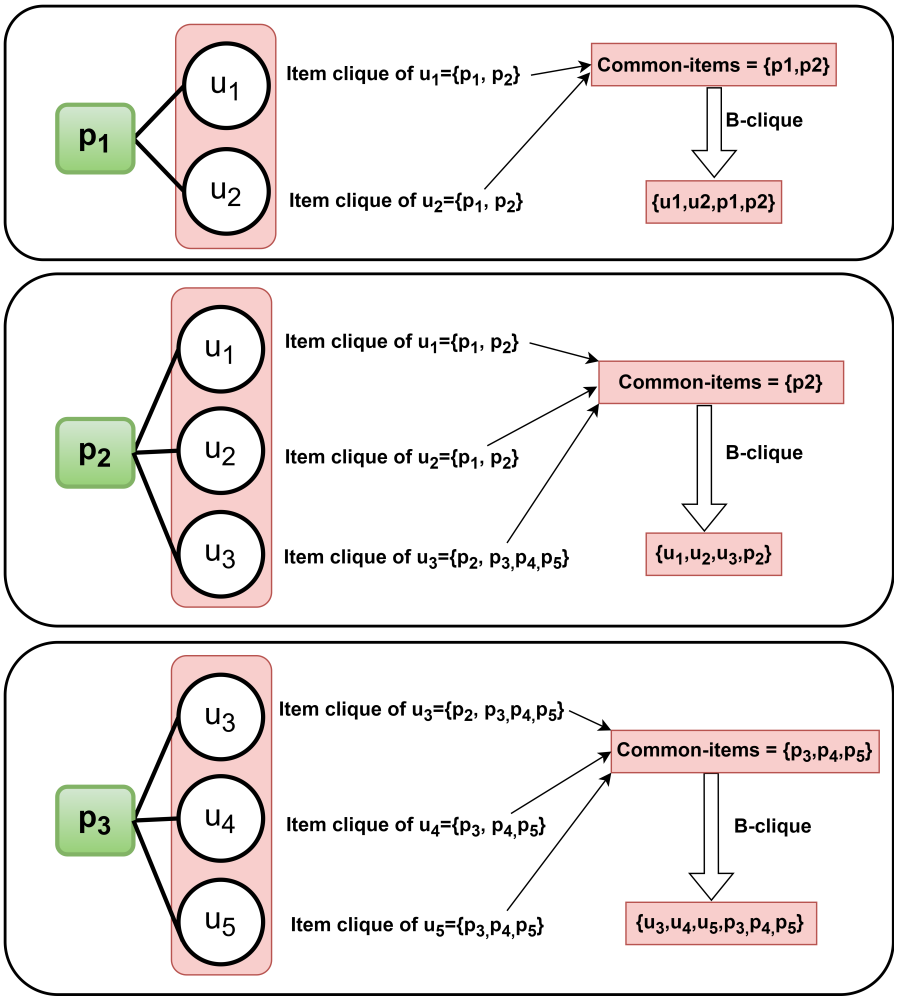

Fig. 9 Illustration of extracting *B*-cliques from User-Item event logs

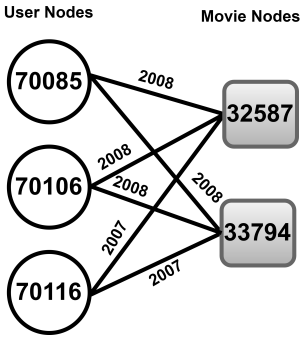

Fig. 10 A snapshot of B-cliq from MovieLense 1M dataset.

70085	70106	70116	32587	33794	Cliq Potential
0	0	0	0	0	0.0000
0	0	0	0	1	0.0001
...	...	...	...	...	...
0	1	1	0	0	0.2100
...	...	...	...	...	...
1	0	1	1	0	0.0406
1	1	1	1	1	0.0203

Fig. 11 Clique Potential table of B-Clique shown in Fig.10 computed following the procedure given in [36].

We construct MRF in the form of factor graph. In factor graph representation, the computation of joint probability is efficient.

5.4 Computation of *B*-COP score

Once the local MRF of a pair of nodes u and p is constructed, the B-COP score between the nodes is obtained using junction tree inference algorithm [38]. Note that B-COP score for a link u - p cannot be computed if u and p are in disjoint cliques as there exists no path connecting these cliques.

The experimental evaluation of proposed approach is given in next section.

6 Experimental Evaluation

The experimentation is carried out on four benchmark datasets whose details are given below.

6.1 Dataset

1. MovieLens [39]: This data set contains more than ten million ratings given by 71,567 users on 10,681 movies of the online movie recommender service MovieLens. The users who have rated at least 20 movies have been chosen randomly to be included in this dataset.
2. Epinions [40]: This is dataset in e-commerce domain. It contains 1,366,830 ratings given by 120,492 users on 755,760 products such as games, apps, music, TV shows, hardware, home appliances, etc.
3. TripAdvisor [41]: This is the rating network of TripAdvisor containing 175,765 ratings given by 145,316 users on 1,759 hotels between the years 2001 and 2009.
4. Amazon [42]: This is the rating network from Amazon, where 2,146,057 users rated 1,230,915 individual items. There are a total of 5,838,041 ratings available in this dataset.

The last three datasets are available for download at *Konnect* website. The dataset details are summarized in Table.1.

Table 1 Datasets details

Dataset	$\|Users\|$	$\|Items\|$	$\|Ratings\|$
MovieLens	71,567	10,681	10,000,054
Epinions	120,492	755,760	1,366,830
TripAdvisor	145,316	1,759	175,765
Amazon	2,146,057	1,230,915	5,838,041

In MovieLens dataset [43] is given with 80% - 20% split with 80% given as training set and test set containing 20%. The training and test sets are formed by splitting the ratings data such that, for every user, 80% of his/her ratings is taken in training and the rest are taken in the test set. In this experimentation, 5 fold cross validation is used. All the 5 sets of training and test datasets are made available at [43]. For other

three datasets, training and test sets are not given. We split them such that last two years of data is taken in test set and remaining in training set.

6.2 Evaluation Metrics

The common evaluation metrics used in recommender systems are Accuracy measures such as Mean Absolute Error (MAE), Root of Mean Square Error (RMSE), Normalized Mean Average Error (NMAE), Set recommendation metrics such as Precision, Recall and Area Under Receiver Operating Characteristic (AUROC), Area Under Precision-recall curve (AUPR) and Rank aware metrics such as Normalized Discounted Cumulative Gain(NDCG) and Normalized Rank-score(NRS). Most of the measures listed above, use rating to calculate the error and hence are not applicable in our context. In this work, we use two decision-support metrics of AUROC and AUPR and one rank-aware metric of NRS to evaluate the performance of proposed measures.

AUROC

AUROC denotes Area Under Receiver Operating Characteristic (ROC) curve. ROC curve is drawn by taking False Positive Rate (FPR) on x-axis and True Positive Rate (TPR) on y-axis [44]. TPR gives the fraction of positive instances predicted correctly out of total instances predicted as positive. FPR is the fraction of negative instances predicted out of total positive instances. AUROC gives the expected proportion of positives ranked before a uniformly drawn random negative.

AUPR

AUPR refers to Area Under Precision-Recall curve. Ratings data is highly unbalanced. In the case of extremely unbalanced data, ROC curve may provide an overly optimistic view of a classifier's performance [45]. In that scenario, the Precision Recall curves(PR curves)[46] can provide more informative representation of assessing performance [47].

NRS

Normalized Rank-score metric measures the ability of a recommendation algorithm to produce a ranked list of recommended items. The recommender system method is efficient, if the ranking given by the method matches with the user's order of buying the items in the recommended list. Rank-score is defined as follows : For a user u, the list of items i recommended to u, that is predicted by the algorithm is captured by $RS_{predicted}$

$$RS_{ideal} = \sum_{j=1}^{|T|} 2^{-\frac{j-1}{h}}$$

$$RS_{predicted} = \sum_{j \in T} 2^{-\frac{rank(j)-1}{h}}$$

$$NRS = \frac{RS_{predicted}}{RS_{ideal}}$$

where $rank(j)$ is the rank given by the recommender algorithm to item j. T is the recommended set of items and h is ranking half-life, an exponential reduction factor

and can be any numeric. In this work, we use the value of 2 for h. In addition, we only calculate NRS for the top 5000 recommended products, rather than all products. Since both Epinions and Amazon are massive and highly sparse datasets, it would be meaningless to consider the score or rank of every item. Recommender systems recommend only top k items in general.

6.3 Results and Discussion

The prediction scores of all baseline link prediction measures are computed using the tool $LPMade$ [48] with default parameters given in the tool $LPMade$. In the computation of TS_B, LS_B and TPI, the damping factor β is taken as 0.5 to ensure consistency with the KZ measure, which is adopted from the $LPMade$. The decay factor β is considered as 0.9 in computation of TF_B following the parameter setting of the base paper [31]. In $TCOP$, maximum path length of 10 is taken for computing BCNS. The accuracy of all these link prediction measures is compared with the standard User-based and Item-based collaborative filtering (CF) methods. Pearson correlation coefficient is used to find the similar users in User-based CF and cosine similarity is used for finding item similarity in Item-based CF. The results obtained for AUROC, AUPR and NRS for all the four datasets are given in Table.2, 3 and 4.

Table 2 AUROC Results

LP measure	**MovieLens**	**Epinions**	**TripAdvisor**	**Amazon**
Non-temporal LP measures				
CN_B	0.5123	0.6531	0.6641	0.7432
JC_B	0.4956	0.6321	0.6293	0.6815
AA_B	0.5749	0.6782	0.7087	0.6526
PA_B	0.6832	0.6656	0.7514	0.7208
KZ_B	0.6635	0.7315	0.6523	0.7598
PF_B	0.6846	0.7532	0.7693	0.7848
COP_B	0.7231	0.8394	0.8690	0.8493
Temporal LP measures				
TS_B	0.7016	0.7831	0.7932	0.7943
LS_B	0.7340	0.8142	0.8432	0.8216
TF_B	0.7532	0.8430	0.8515	0.8314
$TCOP_B$	**0.8165**	**0.8849**	**0.9124**	**0.8943**
Collaborative Filtering methods				
User-based CF	0.6925	0.6342	0.6793	0.7914
Item-based CF	0.7136	0.6513	0.7412	0.8014

First observation in this experimentation is that some of the link prediction measures like Katz, PropFlow could produce better recommendation compared to Item-based CF. In MovieLens dataset, every user rated at least 20 items. It is easy to catch the pattern of the users in MovieLens. But in Epinions, almost 50% of the users rated less than 5 items. This makes difficult to catch user rating behaviour. In MovieLens, the rating distribution is somewhat balanced and in Epinions, most of the ratings are 4 or 5. This makes the CF results worse for Epinion. The LP measures being local measures, the performance is reasonable. At the same time, COP

Table 3 AUPR Results

LP measure	MovieLens	Epinions	TripAdvisor	Amazon
Non-temporal LP measures				
CN_B	0.0092	0.0213	0.0247	0.1413
JC_B	0.0085	0.0194	0.0230	0.0314
AA_B	0.0153	0.0250	0.0723	0.0241
PA_B	0.0251	0.0231	0.0934	0.0614
KZ_B	0.0193	0.0594	0.0314	0.1141
PF_B	0.0286	0.0810	0.1421	0.2143
COP_B	0.0613	0.2410	0.4913	0.3816
Temporal LP measures				
TS_B	0.0365	0.1813	0.2841	0.1816
LS_B	0.0861	0.2316	0.4314	0.2613
TF_B	0.0960	0.2531	0.4526	0.2941
$TCOP_B$	**0.2351**	**0.4104**	**0.6813**	**0.4262**
Collaborative Filtering methods				
User-based CF	0.0415	0.0916	0.0351	0.2143
Item-based CF	0.0491	0.1104	0.0543	0.2416

Table 4 Normalized Rank-score Results at top 5000 items

LP measure	MovieLens	Epinions	TripAdvisor	Amazon
Non-temporal LP measures				
CN_B	0.2051	0.2846	0.2979	0.3975
JC_B	0.1361	0.2067	0.2476	0.2986
AA_B	0.1781	0.2756	0.4287	0.2488
PA_B	0.2680	0.2856	0.4684	0.3815
KZ_B	0.3254	0.4124	0.2476	0.5187
PF_B	0.3299	0.3770	0.4723	0.5475
COP_B	0.3666	0.5124	0.5478	0.7965
Temporal LP measures				
TS_B	0.4001	0.6424	0.4996	0.6956
LS_B	0.5213	0.6837	0.5025	0.7450
TF_B	0.6741	0.6731	0.5752	0.7634
$TCOP_B$	**0.8630**	**0.7423**	**0.7485**	**0.8513**
Collaborative Filtering methods				
User-based CF	0.3994	0.3146	0.4127	0.4364
Item-based CF	0.3027	0.4016	0.4563	0.5472

and TCOP being based on probabilistic measure using the higher order information, achived better performance even in datasets where per user rating is less.

The usage of temporal measures seem to help in improving the quality of recommendations. The time of formation of link or the time of rating given by a user to item plays crucial role, as the user's preferences change over time. The temporal measures TS, LS, TF and TCOP assign more weight to the recent ratings. Therefore, temporal measures performed better than all the other measures including User-based CF and Item-based CF.

Fig.12 depicts a situation in MovieLens network, where non-temporal measures predict a link between the nodes 1 and 1080 and temporal measures predict correctly that the link won't be formed. This is clearly evident from the fact that $B - Cliq1$ is formed with very old links formed in the year 1996 compared to the prediction year

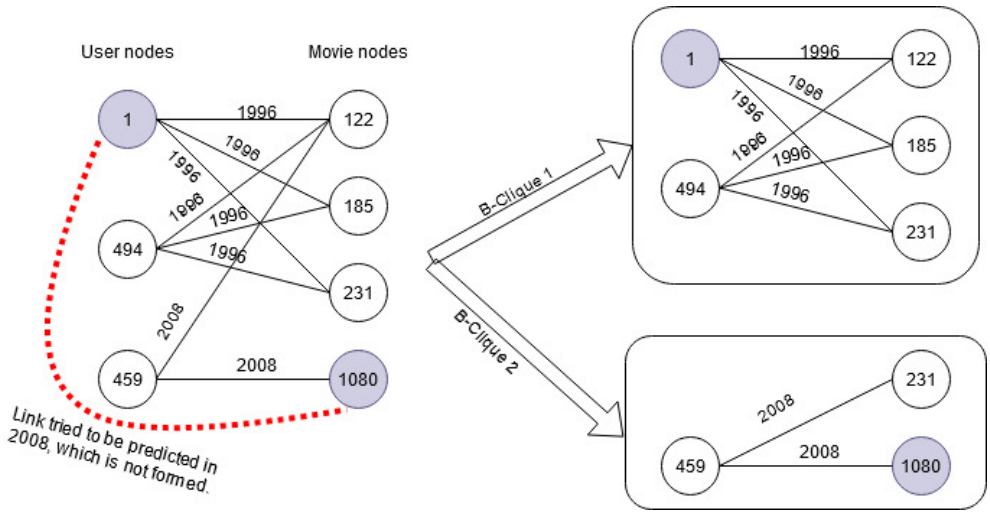

Fig. 12 A snapshot from movielens dataset where the link predicted is not formed.

2008. Fig.13 depicts the situation where the temporal links predict correctly. TCOP outperformed all the other link prediction measures and the user-based and item-based collaborative filtering methods. For MovieLens dataset, the recommendation performance is improved by 6% in terms of AUROC over TF and nearly 10% over CF methods. For Epinion, the increase is 4%, for TripAdvisor and Amazon, it is 6%. There is a clar improvement in the recommendation quality in terms of AUPR and NRS also when time information is used for prediction. All the temporal measures perform better than User-based CF and Item-based CF. TCOP is rated as highest by NRS measure for all the four datasets.

7 Conclusion

In this paper, the recommender systems problem is solved using link prediction approach. Link prediction approach is scalable as it is based on local neighborhood of the large sparse graph. The standard recommender systems approaches do not utilize the temporal information available on the link effectively. In this work, some extensions are proposed to existing temporal measures in bipartite graphs. One of the main contributions of this work is an algorithm for computing temporal cooccurrence probability measure on bipartite graphs and its application to movie recommendation system. Temporal measures for link prediction such as Time score, Link Score, T_Flow and Temporal cooccurrence measure achieve improvement in recommendation quality by utilizing this temporal information more efficiently. However, link prediction approach to solve recommender systems do not address the cold start problem. In future, we would like to work on predicting actual rating of the link.

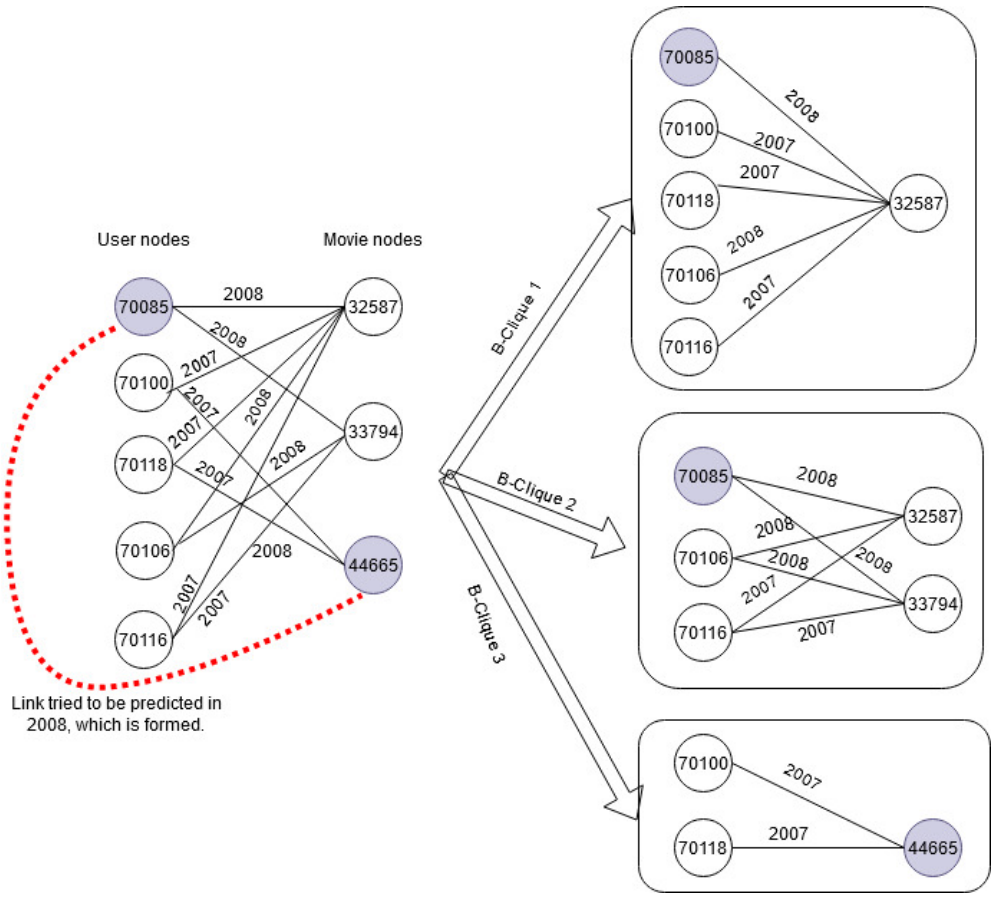

Fig. 13 A snapshot from movielens dataset where the link predicted is formed.

Conflicts of interest

The authors do not have any conflicts of interest.

References

[1] Chui, M. Artificial intelligence the next digital frontier? *McKinsey and Company Global Institute* **47**, 3–6 (2017).

[2] Schafer, J. B., Konstan, J. A. & Riedl, J. E-commerce recommendation applications. *Data Mining and Knowledge Discovery* **5**, 115–153 (2001).

[3] Bell, R. M. & Koren, Y. Lessons from the netflix prize challenge. *Acm Sigkdd Explorations Newsletter* **9**, 75–79 (2007).

[4] Liu, J., Dolan, P. & Pedersen, E. R. *Personalized news recommendation based on click behavior*, 31–40 (2010).

[5] Calero Valdez, A., Ziefle, M., Verbert, K., Felfernig, A. & Holzinger, A. Recommender systems for health informatics: state-of-the-art and future perspectives. *Machine Learning for Health Informatics: State-of-the-Art and Future Challenges* 391–414 (2016).

[6] Pazzani, M. J. & Billsus, D. in *The adaptive web* (eds Brusilovsky, P., Kobsa, A. & Nejdl, W.) Ch. Content-based Recommendation Systems, 325–341 (Springer-Verlag, 2007).

[7] Mooney, R. J. & Roy, L. *Content-based book recommending using learning for text categorization*, 195–204 (2000).

[8] Linden, G., Smith, B. & York, J. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* **7**, 76–80 (2003).

[9] Anand, R. & Jeffrey David, U. Mining of massive datasets (2011).

[10] Koren, Y., Bell, R. & Volinsky, C. Matrix factorization techniques for recommender systems. *Computer* **42**, 30–37 (2009).

[11] Sarwar, B., Karypis, G., Konstan, J. & Riedl, J. *Analysis of recommendation algorithms for e-commerce*, 158–167 (2000).

[12] Shams, B. & Haratizadeh, S. Graph-based collaborative ranking. *Expert Syst. Appl.* **67**, 59–70 (2017).

[13] Huang, Z., Chung, W. & Chen, H. A graph model for e-commerce recommender systems. *J. Am. Soc. Inf. Sci. Technol.* **55**, 259–274 (2004).

[14] Huang, Z., Chung, W. & Chen, H. A graph model for e-commerce recommender systems. *Journal of the American Society for information science and technology* **55**, 259–274 (2004).

[15] Sarwar, B., Karypis, G., Konstan, J. & Riedl, J. *Item-based collaborative filtering recommendation algorithms*, 285–295 (2001).

[16] Melville, P., Mooney, R. J. & Nagarajan, R. *Content-boosted collaborative filtering*, Vol. 9, 187–192 (2001).

[17] Adomavicius, G., Mobasher, B., Ricci, F. & Tuzhilin, A. Context-aware recommender systems. *AI Magazine* **32**, 67–80 (2011).

[18] Kefalas, P., Symeonidis, P. & Manolopoulos, Y. A graph-based taxonomy of recommendation algorithms and systems in lbsns. *IEEE Transactions on Knowledge and Data Engineering* **28**, 604–622 (2015).

[19] Ali, Z., Qi, G., Kefalas, P., Abro, W. A. & Ali, B. A graph-based taxonomy of citation recommendation models. *Artificial Intelligence Review* 1–44 (2020).

[20] Li, X. & Chen, H. Recommendation as link prediction in bipartite graphs: A graph kernel-based machine learning approach. *Decision Support Systems* **54**, 880–890 (2013).

[21] Zhang, L., Zhao, M. & Zhao, D. Bipartite graph link prediction method with homogeneous nodes similarity for music recommendation. *Multimedia Tools and Applications* 1–19 (2020).

[22] Liben-Nowell, D. & Kleinberg, J. The link-prediction problem for social networks. *Journal of The American Society For Information Science and Technology* **58**, 1019–1031 (2007).

[23] Cremonesi, P., Koren, Y. & Turrin, R. *Performance of recommender algorithms on top-n recommendation tasks*, 39–46 (2010).

[24] Jaya Lakshmi, T. & Durga Bhavani, S. *Link prediction in temporal heterogeneous networks*, 83–98 (Springer, 2017).

[25] Lakshmi, T. J. & Bhavani, S. D. *Link prediction measures in various types of information networks: a review*, 1160–1167 (IEEE, 2018).

[26] Lichtenwalter, R. N., Lussier, J. T. & Chawla, N. V. *New perspectives and methods in link prediction*, 243–252 (2010).

[27] Li, J., Zhang, L., Meng, F. & Li, F. Recommendation algorithm based on link prediction and domain knowledge in retail transactions. *Procedia Computer Science* **31**, 875 – 881 (2014).

[28] Davis, D. A., Lichtenwalter, R. & Chawla, N. V. Supervised methods for multi-relational link prediction. *Social Network Analysis and Mining* **3**, 127–141 (2013).

[29] Munasinghe, L. & Ichise, R. *Time aware index for link prediction in social networks.*, 342–353 (Springer, 2011).

[30] Choudhary, P., Mishra, N., Sharma, S. & Patel, R. Link score: A novel method for time aware link prediction in social network. *ICDMW* (2013).

[31] Munasinghe, L. Time-aware methods for link prediction in social networks. *PhD Thesis, The Graduate University for Advanced Studies* (2013).

[32] Wang, C., Satuluri, V. & Parthasarathy, S. *Local probabilistic models for link prediction*, 322–331 (IEEE, 2007).

[33] Druzdzel, M. J. *Some properties of joint probability distributions*, 187 (Elsevier, 2014).

[34] Kashima, H., Kato, T., Yamanishi, Y., Sugiyama, M. & Tsuda, K. *Link propagation: A fast semi-supervised learning algorithm for link prediction*, 1100–1111 (SIAM, 2009).

[35] Clauset, A., Moore, C. & Newman, M. E. Hierarchical structure and the prediction of missing links in networks. *Nature* **453**, 98 (2008).

[36] Jaya Lakshmi, T. & Durga Bhavani, S. Temporal probabilistic measure for link prediction in collaborative networks. *Applied Intelligence* **47**, 83–95 (2017).

[37] Calders, T. & Goethals, B. *Mining all non-derivable frequent itemsets*, Vol. 2, 74–85 (Springer, 2002).

[38] S. L. Lauritzen, D. J. S. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)* **50**, 157–224 (1988).

[39] https://grouplens.org/datasets/ (2009).

[40] Epinions product ratings network dataset – KONECT (2017). URL http://konect.cc/networks/epinions-rating.

[41] Tripadvisor network dataset – KONECT (2017). URL http://konect.cc/networks/wang-tripadvisor.

[42] Amazon (wang) network dataset – KONECT (2017). URL http://konect.cc/networks/wang-amazon.

[43] http://files.grouplens.org/datasets/movielens/ml-10m-README.html (2009).

[44] Han, J., Kamber, M. & Pei, J. Data mining concepts and techniques third edition. *University of Illinois at Urbana-Champaign Micheline Kamber Jian Pei Simon Fraser University* (2012).

[45] Davis, J. & Goadrich, M. *The relationship between precision-recall and roc curves*, 233–240 (2006).

[46] Boyd, K., Eng, K. H. & Page, C. D. *Area under the precision-recall curve: Point estimates and confidence intervals*, 451–466 (Springer, 2013).

[47] Chawla, N. Data mining for imbalanced datasets: An overview 853–867 (2005).

[48] Lichtenwalter, R. N. & Chawla, N. V. Lpmade: Link prediction made easy. *Journal of Machine Learning Research.* **12**, 2489–2492 (2011).