Who to Connect to? Joint Recommendations in Cross-layer Social Networks

Jiaqi Liu[1], Qi Lian[3], Luoyi Fu[2], Xinbing Wang[1,2]
[1,2,3]Dept. of {Electronic Engineering, Computer Science, Joint Institute}, Shanghai Jiao Tong University, China

***Abstract*—Social recommendation has been widely applied to offer users suggestions on who to connect to, where most existing strategies overlook the existence of multi-type connections among users. To overcome such limitation, we characterize each type of connections by a corresponding network layer and then propose a novel algorithm for joint recommendations in cross-layer social networks. Particularly, two types of results are presented in the paper. (i) Our proposed algorithm, named as Cross-layer 2-hop Path (C2P) algorithm, implements the joint recommendation by suggesting a user establish connections to his cross-layer two-hop neighbors, i.e., those who link to the user by two-hop paths with the two hops belonging to two different layers, respectively. In doing so, each produced recommendation item is a combination of user relationships in both two layers and thus can better meet user demands. (ii) By analytical derivations, along with further empirical validation on real datasets, we give the performance evaluation on our proposed algorithm. Firstly, we prove that the algorithm is efficiently implementable with a constant complexity in each recommendation. Then, we evaluate its recommendation performance by two metrics, i.e., acceptance and diversity, where the former metric measures recommendation accuracy and the latter one measures an algorithm's capability to provide diverse recommendation items. Our results show that C2P algorithm is optimal in terms of acceptance and for diversity, its performance is in the same order of the theoretical upperbound. And finally, the effectiveness of the proposed algorithm is validated by our simulations on three real datasets, where it outperforms baseline algorithms with an up to 38% acceptance gain and obtains an around 0.5 diversity ratio to the theoretical upperbound.**

I. Introduction

Social recommendation [1]–[3] has been proved to be an effective method to overcome information overload, i.e., difficulty in making decisions with too much related information. Specifically, recommendation algorithms regularly offer users some personalized suggestions on who to connect to, through providing a list that contains recommendation items for users to browse and select from according to their own wills. Many different recommendation algorithms have been proposed and two basic types are relationship-based [4], [5] and similarity-based [6], [7] recommendation, where the former one works based on user relationships and the latter one recommends a user who shares common features with him. In addition, some variants are developed, including caching-aware [8], context-aware [9], package-group recommendation [10], etc. However, all these algorithms are designed for single-layered networks that overlook the layered structure of realistic networks.

Many realistic networks can be regarded as cross-layer ones, where all the layers have the same node set but each of them contains a certain type of connections among nodes [11], [12]. A typical example comes to academic networks, where any

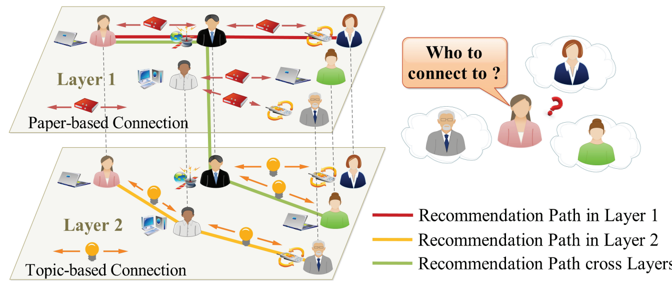


Fig. 1. An illustration on layered recommendations in academic networks.

two authors are related if they have some common published papers, or alternatively, do research on the same topics. In this way, two types of author connections are naturally formed as illustrated in Figure 1. In cross-layer networks, recommendation is also an essential issue, which, however, suffers from a negative performance since most of existing algorithms only consider a particular layer of the network. To illustrate, let us continue to the example of academic networks. When make recommendations for authors, most existing applications only extract paper-based connections among authors, i.e., connections between two authors who collaborate common papers. We note that this strategy results in information loss, where the users' demands are not fully satisfied since they may also want to connect to some topic-related friends. To summarize, the recommendation algorithms that only exploit single layer cannot work well in cross-layer networks. Then, the following questions arise naturally: *What are distinctive features of user demand in cross-layer networks? How to design an effective recommendation algorithm under this condition?*

The answer is that each recommendation item should be an instinct combination of the user's wills in both layers rather than a simple mixture. Specifically, let us consider a method that gives recommendations by mixing the items in two layers together. In this case, though it simultaneously contains user information in two layers, each item is produced from either of the two layers and thus the whole performance is a linear combination of that of the two basic ones, which therefore results in an unsatisfactory performance. Based on this fact, we put forward a novel idea that whether the algorithm's performance can be improved if every recommendation item contains cross-layer user information? Obviously, the answer is positive since such joint recommendation is more attractive for users, which is also validated in our experimental measurements.

To achieve this aim, we propose a novel algorithm named Cross-layer 2-hop Path (C2P) algorithm. The core idea of the algorithm is inspired by a prior commonly involved recommendation method [4], [5], which recommends a user his two-hop neighbors with the basic idea that a user is likely to be interested in the friends of his friends. While this method has been proved to be effective with better acceptace [1], it fails to work well in cross-layer networks due to its exclusive focus on singer-layered user relations. To overcome this limitation, we design an algorithm that makes joint recommendation by utilizing cross-layer two-hop paths, defined as two-hop paths whose two hops belong to the two different layers respectively. The algorithm recommends with a probability proportional to the number of cross-layer two-hop paths between the user and the candidate and thus, each item returned by it is an instinct combination of user relationships in both two layers.

Moreover, we conduct the performance evaluation on C2P algorithm in both theoretical analysis and experimental measurements. Firstly, we prove that the algorithm is an efficiently implementable one. The computational complexity of it is in constant order for each recommendation if the average node degree is a constant, which holds in most of realistic networks. Then, we evaluate the recommendation performance of C2P algorithm by two metrics, i.e., acceptance and diversity, where the former one measures the recommendations accuracy and the latter one measures the algorithm's capability to generate diverse recommendations among users. Our results show that the performance of our proposed algorithm is the optimal one in terms of acceptance and for diversity, its performance is in the same order of the theoretical upperbound. Besides theoretical analysis, we also conduct experimental measurements based on three datasets extracted from Microsoft Academic Graph [13]. Results show that C2P algorithm outperforms the single layer based ones and the mixture one.

Our main contributions are summarized as follows:

- **Design:** We propose a novel algorithm, i.e., C2P algorithm, that makes joint recommendation in cross-layer networks by fully utilizing user relationships among layers.
- **Analysis:** We theoretically analyze the performance of our proposed algorithm and results show that the algorithm is efficiently implementable, optimal in acceptance and in the same order of theoretical upperbound in diversity.
- **Validation:** We make experimental measurements on three real datasets to show the outperformances of C2P algorithm, with an up to 38% acceptance gain and an approximately 0.5 diversity ratio to the theoretical upperbound.

Different from prior art that adopt sophisticated data mining techniques, this paper provides analysis primarily from probabilistic manner. It ensures two-folded superiorities: theoretical tractability and data independence. On one hand, our results is theoretically tractable in the sense that the influential factors and their impacts on results are directly given in mathematical expressions. On the other hand, performance of techniques in data mining is often unstable and sensitive to data fluctuation, which can be solved in our work due to the data independence. To our best knowledge, this work is the first attempt towards cross-layer recommendations with systematic theoretical support, and we believe that it can stimulate more creative works.

II. Related Work

Recommendation systems have been adopted in many realistic applications. Various algorithms [14]–[16] are proposed and according to their intuitions, the algorithms can be classified into two main types: relationship-based and similarity-based recommendations, where the former one [4], [5] makes recommendations based on the idea that a user is more likely to be interested in the one sharing common friends with him, and the latter one [6], [7] extracts recommendations for a user from those owning the same content, interest, or other features as him. Though the algorithms have been exhaustively studied, most of them are designed for single-layered networks and thus cannot work well in cross-layer social networks.

Many realistic networks can be regraded as cross-layer ones due to the fact that more than one type of connections may exist among users. Due to the typical structures, applications in cross-layer networks [12], [17]–[20] often reveal distinctive properties and it is the same for recommendations, since that user demand is often reflected in multiple layers of the network simultaneously [11]. To study recommendations in cross-layer networks, we should first select an appropriate mathematical model. Bródka et al. [21] define a basic multi-layered network where each layer includes a fixed set of nodes with edges that may vary from layer to layer. Magnani et al. [22] propose a more flexible model that allows different nodes sets for each layer. Hao et al. [23] emphasize the relationships among layers by measuring the influence of one layer on the other layers. In additional, another model named as Affiliation Network Model [24] is established by utilizing user's attributes such as gender, interest and residence. Exploiting such information can help to make recommendations more precise [3], [25], [26] since that attributes represent the inherent relationships among users, and thus we select this model in our work.

We note that studying recommendation algorithms in cross-layer networks is an essential issue and to our best knowledge, this work is the first attempt towards this purpose.

III. System Model

In this section, we first give the underlying network model. Then, we introduce two metrics on recommendation systems, i.e., acceptance and diversity, that will be used in later performance evaluation. And finally, we discuss and model users' demands on making friends in cross-layer networks.

A. Network Modeling for Recommendation Systems

The network model we used in this work is an extension of Affiliation Network Model [24] that is widely adopted in modeling evolving networks with affiliation relationships. The reasons that we choose this model are two folded:

- The model is a good capture of realistic networks with properties such as power-law degree distribution, densification (the ratio of number of edges to that of nodes grows over time) and shrinking diameter (the network diameter reduces over time to a constant).
- The model is a mathematically tractable one that guarantees theoretical analysis.

In our modeling of network structure, we make some extensions on this model in order to make it fit in characterizing cross-layer networks. We note that all the network properties generated by the original model still hold in our extension. The extended model is illustrated as follows.

1) Network Structure: We use two basic structures, i.e., bipartite graph and generated graph, to characterize the relationships between users and attributes, and that among users, respectively. The two graphs are modeled as follows:

- **Bipartite Graph** $B(V_0, V_i)$**:** Let V_0 denote the set of users and V_i, $i \in \{1, 2\}$, denote the set of i-th types of attributes. $B(V_0, V_i)$ is a bipartite graph composed of the node sets V_0 and V_i, and a set of edges characterizing the relationships between them.
- **Generated Graph** $G(V_0|V_i)$**:** The graph $G(V_0|V_i)$ characterizes the social relationships among users in V_0, which is generated from $B(V_0, V_i)$ following the rule that *an edge exists between* $v_1, v_2 \in V_0$ *in* $G(V_0|V_i)$ *if and only if they share a common neighbor* $u \in V_i$ *in* $B(V_0, V_i)$.

In our model, we set parameter $i \in \{1, 2\}$ and thus the network is two-layered. Under this assumption the connections among nodes in V_0 can be classified into two types:

- *Type 1:* Connections in $G(V_0|V_1)$ generated from $B(V_0, V_1)$.
- *Type 2:* Connections in $G(V_0|V_2)$ generated from $B(V_0, V_2)$.

Therefore, the network of nodes in V_0 is two-layered, and we denote it as $\boldsymbol{G}(V_0) = [G(V_0|V_1), G(V_0|V_2)]$.

The model has its applicability to massive realistic scenarios. Take academic networks for example. Denoting authors, papers and topics by V_0, V_1 and V_2, respectively, the model can be intuitively interpreted as follows. For the bipartite graph $B(V_0, V_1)$, an edge exists between nodes $v \in V_0$ and $u \in V_1$ indicates that the author v published the paper u, and edges in $B(V_0, V_2)$ have a similar physical meaning. For the generated graph $G(V_0|V_1)$, any two authors have a common neighbor in $B(V_0, V_1)$ indicates that they collaborate a paper and thus are correlated in the generated network, which is also known as co-author network. Combing $G(V_0|V_1)$ and $G(V_0|V_2)$ together we obtain a two-layered network $\boldsymbol{G}(V_0)$ that can characterize two types of relationships among authors simultaneously.

2) Evolving Process: In our model, the two graphs jointly evolve following the preferential attachment manner [27] [28]. The evolving process of bipartite graph is given in Algorithm 1, according to which that of generated graph can be obtained and thus we omit it here for concision.

Figure 2 illustrates an example of the evolving process of $B(V_0, V_1)$ and $B(V_0, V_2)$ with a new node arrival in V_0. In this example, the new arrived node picks node v_1 as the prototype in $B(V_0, V_1)$ with probability $\frac{1}{2}$ and randomly copies $c_{01} = 1$ edge of the prototype. Similarly, node v_2 is picked as the prototype in $B(V_0, V_2)$ and its $c_{02} = 2$ edges are copied. And finally, the two graphs $B(V_0, V_1)$ and $B(V_0, V_2)$ evolve from the left one to the right one during this time slot.

B. User Demand for Social Recommendation

User demand is an essential issue in recommendation systems since it reflects the user's own wills on who to connect to.

Algorithm 1: Evolution of $B(V_0, V_1)$ and $B(V_0, V_2)$

Fix parameters $\alpha_0, \alpha_1, \alpha_2 \in (0, 1)$ and $c_{01}, c_{02}, c_{10}, c_{20} > 0$.

At time $t = 0$**:**
Give two initial bipartite graphs $B(V_0, V_1)$ and $B(V_0, V_2)$.

At time $t > 0$**:**
(Evolution of V_0***)***
Arrival: A node v arrives with probability α_0 and is added to the node set V_0.
Preferential Attachment: In bipartite graph $B(V_0, V_1)$, a node $u \in V_0$ is chosen as the prototype for the new node with a probability proportional to its degree. Then, c_{01} edges are copied from u, that is, c_{01} neighbors of u, denoted by $i_1, ..., i_{c_{01}}$, are chosen uniformly and randomly (without replacement), and the edges $(v, i_1), ..., (v, i_{c_{01}})$ are added to the graph. In $B(V_0, V_2)$, c_{02} edges are created following the same way.
(Evolution of V_1 ***and*** V_2***)***
A node v is added to V_1 with probability α_1 and c_{10} edges are created in $B(V_0, V_1)$ following a symmetrical process. And similarly, c_{20} edges are created in $B(V_0, V_2)$.

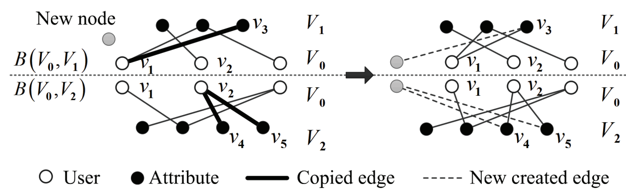


Fig. 2. An example of the evolving process of $B(V_0, V_1)$ and $B(V_0, V_2)$.

Particularly, it reveals some distinctive features in cross-layer networks that we will discuss in this subsection.

Many empirical studies [1], [2] have verified that in social networks, a user is more likely to connect to a popular user, i.e., the user with large degree. This is an intuitive result since that celebrities such as famous singers, actors and politicians are often attractive to ordinary users. Based on this fact, many existing works assume that a user's demand on making friends with user u is proportional to its degree D_u. In cross-layer networks, since information can exchange between layers, a user's demand on user u is determined by its degrees in both two layers. Denote the demand of user v by $E(v)$ and we have

- $\mathbb{P}(E(v) = u) \propto D_{u1}$ and $\mathbb{P}(E(v) = u) \propto D_{u2}$.

In addition to the degrees of node u, we note that a user's demand may also relate to the characteristics of itself. Though a user v with large degree can attract users to connect to it, its own demand for making new friends is negative in reverse. The reasons of this phenomenon are three folded:

1) The number of existing friends of user v is large and thus his demand on making friends has been greatly satisfied.
2) Since user v has been recommended to a variety of users, he can make friends by accepting requests from others instead of taking the initiative to add friends.
3) For user v, there are few popular users with larger degree than him that user v may be interested in.

Similarly, to illustrate it, let us consider an example in academic networks. It is an obvious fact that most of researchers are willing to build connections with experts who are authoritative in the field. However, for the experts, their demands to make new friends are negative since they have owned a large number of friends and few new ones can attract them. Based on this observation, the demand of user v should be in inverse proportion to its own degrees, that is,

- $\mathbb{P}(E(v) = u) \propto \frac{1}{D_{v1}}$ and $\mathbb{P}(E(v) = u) \propto \frac{1}{D_{v2}}$.

Furthermore, the above two conditions should be considered simultaneously. In other words, the probability that user v has the demand to connect to user u is proportional to the degree of node u and is inversely proportional to its own degree in this layer. Therefore, the probability can be formulated as

$$\mathbb{P}(E(v) = u) \propto \frac{\frac{1}{D_{v2}}}{\frac{1}{D_{v1}} + \frac{1}{D_{v2}}} D_{u2} + \frac{\frac{1}{D_{v1}}}{\frac{1}{D_{v1}} + \frac{1}{D_{v2}}} D_{u1},$$

where the effects of D_{v1} and D_{v2} are normalized to 1. Then, make the normalization among all user u and we have

$$\mathbb{P}(E(v) = u) = \frac{D_{v1} D_{u2} + D_{v2} D_{u1}}{2|E_2| D_{v1} + 2|E_1| D_{v2}},$$

where $|E_i|$ denote the total number of edges in the i-th layer.

C. Metrics on Recommendation Systems

We evaluate the performance of recommendation algorithms by two metrics: acceptance and diversity, where the former one measures recommendation accuracy and the latter one makes measurements on an algorithm's capability to provide diverse or idiosyncratic recommendation items.

1) Acceptance: A fundamental objective of recommendation algorithms is to provide accurate recommendations, which is evaluated by the metric acceptance defined as follows.

Definition 1 (Acceptance)**.** *With the recommendation list $R(v)$ of user v in network $\boldsymbol{G}(V_0)$, the acceptance is defined as*

$$A_{\boldsymbol{G}}(v) = \frac{1}{|R(v)|} \sum\nolimits_{u \in R(v)} \mathbb{P}(E(v) = u).$$

The acceptance of a recommendation algorithm is defined as the average value of acceptances of all users, that is,

$$A_{\boldsymbol{G}} = \frac{1}{n} \sum\nolimits_{v \in V_0} A_{\boldsymbol{G}}(v).$$

2) Diversity: The performance of recommendation algorithms can be evaluated from different aspects, and considering acceptance along may be biased. In particular, another important goal of recommendation algorithm is to provide user with highly idiosyncratic or personalized recommendations, which can help to enrich the user's friendships. We use the metric diversity to evaluate the algorithm's performance towards this goal. This metric is firstly proposed in [29] and we represent it in the following definition.

Definition 2 (Diversity)**.** *Assume that each node v owns a list containing all his recommended friends and denote the list as $R(v)$. The aggregate diversity is defined as*

$$D_{\boldsymbol{G}} = \left| \cup_{v \in V_0} R(v) \right|.$$

For convenience, we present Table I to list all notations that will be used in later analysis, proofs and discussions.

TABLE I
NOTATIONS AND DEFINITIONS

Notation	Definition
$L(H_1, H_2)$	Two-hop path with the first hop H_1 and the second hop H_2.
$B(V_0, V_i)$	Bipartite graph with node sets V_0 and V_i.
$G(V_0 \mid V_i)$	Generated graph of V_0 obtained from $B(V_0, V_i)$.
$\boldsymbol{G}(V_0)$	Two-layered network consisting of the graphs $G(V_0 \mid V_1)$ and $G(V_0 \mid V_2)$.
$D_{vi}(t)$	Degree of node $v \in V_0$ in $G(V_0 \mid V_i)$ at time t.
$d_{vi}(t)$	Degree of node $v \in V_0$ in $B(V_0, V_i)$ at time t.
α_i	Arrival probability of nodes in V_i.
γ_{ij}	Parameter of degree growth of node $v \in V_i$ in $B(V_i, V_j)$.
$R(v)$	Recommendations list for user v.
$E(v)$	Friendship demands of user v.
$A_{\boldsymbol{G}}$	Acceptance of the algorithm in network $\boldsymbol{G}$.
$D_{\boldsymbol{G}}$	Diversity of the algorithm in network $\boldsymbol{G}$.
P_{uv}	Probability that an edge exists between node u and node v.

IV. The Proposed Recommendation Algorithm: Cross-layer 2-hop Path (C2P) Algorithm

In order to better meet the user demand as illustrated before, we propose a novel algorithm named as Cross-layer 2-hop Path (C2P) algorithm. Before the detailed description on the algorithm, we first introduce the cross-layer two-hop path.

Definition 3 (Cross-layer two-hop path)**.** *For a two-hop path $L(H_1, H_2)$ starting from X_0 with the first hop $H_1 = (X_0, X_1)$ and the second hop $H_2 = (X_1, X_2)$, we call it a cross-layer one if either of the following two cases is satisfied*

$$\begin{aligned} &\textit{Case 1:} \quad H_1 \in G(V_0|V_1), H_2 \in G(V_0|V_2) \\ &\textit{Case 2:} \quad H_1 \in G(V_0|V_2), H_2 \in G(V_0|V_1). \end{aligned}$$

With the above definition, we now come to the implementation of C2P algorithm and then analyze its complexity.

A. Implementation of C2P Algorithm

Algorithm 2 describes the implementation of C2P algorithm, where for a given node v, the algorithm produces a list $R(v)$ that contains k recommendation items based on the topology of network $\boldsymbol{G}(V_0)$. Specifically, line 1 in the algorithm conducts the initialization. Then, operations in line 2 to line 10 use a flooding method to find out all cross-layer two-hop paths that start from node v and update each node's weight as the number of paths between it and node v. And finally, k nodes are added in the recommendation list with probabilities proportional to their weights, as shown in line 11.

An example of the implementation process of C2P algorithm is given in Figure 3. Note that when conduct recommendations for node v, the algorithm floods over all cross-layer two-hop paths that start from node v. And thus to facilitate understanding, we use a two-hop paths tree to characterize all these paths, where node v acts as the root of the tree and all two-hop neighbors of node v are placed as the leafs connected to the root by relay nodes w_i, $i = \{1, 2, 3\}$. From the tree we

Algorithm 2: Implementation of C2P Algorithm

Input: Network $\boldsymbol{G}(V_0) = [G(V_0|V_1), G(V_0|V_2)]$;
Parameter $k > 0$.

Output: Recommendation list of node v, i.e., $R(v)$.

```
1  Set weight(u) = 0 for all u ∈ V_0 that u ≠ v.
2  for w in neighbors of v do
3      if the hop H_1 = (v, w) ∈ G(V_0|V_1) then
4          for u in neighbors of w do
5              if the hop H_2 = (w, u) ∈ G(V_0|V_2) then
6                  weight(u) ← weight(u) + 1
7      else
8          for u in neighbors of w do
9              if the hop H_2 = (w, u) ∈ G(V_0|V_1) then
10                 weight(u) ← weight(u) + 1
11 Select k nodes with the probabilities proportional to their
   weights and add them to the recommendation list R(v).
```

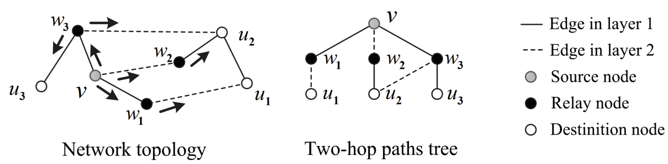


Fig. 3. An example of the implementation of C2P recommendation algorithm.

can find that there are total 3 cross-layer two-hop paths, i.e., $v - w_1 - u_1$, $v - w_2 - u_2$ and $v - w_3 - u_2$, where 2 of them connect to node u_2 and only 1 of them connects to node u_1. Then, node u_1 and node u_2 are selected as the candidates and are recommended to node v with probabilities $\frac{2}{3}$ and $\frac{1}{3}$, respectively. In addition, we note that though node u_3 is also connected to node v by a two-hop path $v - w_3 - u_3$, it is not selected as a candidate since both two hops belong to layer 1.

B. Complexity

We now come to analyze the complexity of C2P algorithm. The result is provided in Theorem 1 and we note that in most cases the complexity can achieve $\Theta(1)$, which indicates that the algorithm is an efficiently implementable one.

Theorem 1. *The complexity of conducting recommendations for a node v through C2P algorithm is $O\left(\sum_{i=1}^{d_v} d_{w_i}\right)$, where w_i denotes the i-th neighbor of node v.*

Proof. The implementation of C2P algorithm has been given in previous part, and in the following we prove the results by analyzing operations in Algorithm 2 line by line.

Line 1 makes the initialization and the corresponding complexity is $\Theta(1)$. Operations in line 2 to line 10 include two "for" loops, where the outer one traverses all the neighbors of node v, denoted by w_i, with loop times d_v and the inner one traverses part of neighbors of relay node w_i. Note that the inner loop does not need to search all the neighbors of node w_i since that some of them do not own cross-layer two-hop paths to node v. Then, note that the operation within two loops is an assignment with complexity $\Theta(1)$ and thus we have that the implementation complexity is upper bounded by $\Theta\left(\sum_{i=1}^{d_v} d_{w_i}\right)$. Line 11 selects recommendation items based on nodes' weights with complexity $\Theta(1)$. Combing the above three parts together, we complete the proof. $\square$

Remark 1. *In most cases, the implementation complexity of C2P algorithm is $\Theta(1)$. This result can be obtained by considering the average complexity, which can be calculated as $\mathbb{E}\left[\sum_{i=1}^{d_v} d_{w_i}\right] = \mathbb{E}\left[d_v\right]\mathbb{E}\left[d_{w_i}\right] = d^2$, where d denotes the average node degree. In most realistic networks d is a constant, and thus the average complexity is $\Theta(1)$.*

V. Performance Analysis

In this section, we first present some useful lemmas on network properties, based on which we then show the influence of node degrees on recommendations for a given node (Theorem 2). And finally, we give the performance evaluation on acceptance and diversity (Theorem 3 and Theorem 4).

A. Impact of User Degrees on Recommendations

Since the network is an evolving one, node degrees are time dependent. In the following lemma we discuss how the node degree in bipartite graph grows over time.

***Lemma* 1.** *For a node $v \in V_0$ in $B(V_0, V_i)$ where $i = \{1, 2\}$, given the condition that node v is added to the network at time t_v with degree $d_v(t_v)$, the degree of node v at time t is*

$$d_v(t) = d_v(t_v)\left(\frac{t}{t_v}\right)^{\gamma_{0i}},$$

where $\gamma_{0i} = \frac{\alpha_i c_{i0}}{\alpha_0 c_{0i} + \alpha_i c_{i0}}$ is a constant. And similarly, for a node $u \in V_i$ that is added to the network at time t_u, we have

$$d_u(t) = d_u(t_u)\left(\frac{t}{t_u}\right)^{\gamma_{i0}},$$

where $\gamma_{i0} = \frac{\alpha_0 c_{0i}}{\alpha_0 c_{0i} + \alpha_i c_{i0}}$.

Proof. We start the proof by calculating the degree of node $v \in V_0$ in $B(V_0, V_1)$. According to the evolving process, the degree of node v increases only in the case that a new node $u \in V_1$ is added to the network and one of its edges points to node v. Note that in this process, each endpoint of nodes in V_0 is selected with equal probability as the destination of the new created edge. Consequently, a new created edge connects to node v with probability $\frac{d_v(t)}{e_{01}(t)}$, where $e_{01}(t) = (\alpha_0 c_{01} + \alpha_1 c_{10})t$ is the number of edges in $B(V_0, V_1)$. Then, we have

$$d_v(t) = d_v(t-1) + \alpha_1 c_{10} \frac{d_v(t-1)}{e_{01}(t-1)}.$$

Using the Chernoff bound we have

$$\begin{aligned} d_v(t) =& d_v(t-1)\left(1 + \frac{\alpha_1 c_{10}}{(\alpha_0 c_{01} + \alpha_1 c_{10})(t-1) \pm o(t)}\right) \\ =& d_v(t_v) \prod_{k=t_v}^{t-1}\left(1 + \frac{\gamma_{01}}{k}\right) \\ =& d_v(t_v)\left(\frac{t}{t_v}\right)^{\gamma_{01}}, \end{aligned}$$

where $\gamma_{01} = \frac{\alpha_1 c_{10}}{\alpha_0 c_{01}+\alpha_1 c_{10}}$. Using the same method we can calculate the node degree in other cases and obtain a similar result, which completes the proof. □

Besides the node degree in bipartite graph $B(V_0, V_i)$, the degree in generated graph $G(V_0|V_i)$ is also an important one that will be used in the following analysis. Lemma 2 gives the relationship between them and consequently we can obtain the node degree in $G(V_0|V_i)$ from that in $B(V_0, V_i)$.

***Lemma* 2.** *For a node $v \in V_0$ in $G(V_0|V_i)$ where $i = \{1, 2\}$, its degree satisfies*

$$D_{vi}(t) = c_i d_{vi}(t),$$

where $c_i = \frac{\alpha_0 c_{0i}+\alpha_i c_{i0}}{\alpha_i} - 1$ is a constant.

Proof. According to the generation rule, every neighbor of node v in $B(V_0, V_i)$, denoted by u, provides $d_u(t) - 1$ edges connecting to node v in $G(V_0|V_i)$. Considering all the possible values of $d_u(t)$ and using the law of total probability, we have

$$\begin{aligned} D_{vi}(t) &= \sum\nolimits_{k=1}^{\alpha_0 t} d_{vi}(t)(k-1)\mathbb{P}(d_u(t) = k) \\ &= \left(\mathbb{E}(d_u(t)) - 1\right) d_{vi}(t), \end{aligned}$$

where $\mathbb{E}(d_u(t)) = \frac{(\alpha_0 c_{0i}+\alpha_i c_{i0})t}{\alpha_i t}$. □

Then, we calculate the existence probability of an arbitrary edge in the graph $G(V_0|V_i)$ as given in Lemma 3.

***Lemma* 3.** *In the graph $G(V_0|V_i)$ at time t, the probability that node u and node v are connected is*

$$P_{uv} = \frac{D_{ui}(t)D_{vi}(t)}{2|E_i(t)|},$$

where $D_{ui}(t)$ denotes the degree of node u in $G(V_0|V_i)$ at time t, $D_{vi}(t)$ denotes that of node v and $|E_i(t)|$ is the total number of edges in $G(V_0|V_i)$.

Proof. According to the evolving process, node u and node v are connected if and only if they share a common neighbor $w \in V_i$ in $B(V_0, V_i)$. Based on this fact, we make the proof by calculating the probability that at least one node $w \in V_i$ in $B(V_0, V_i)$ connects to node u and node v simultaneously.

Assume that node u and node v are added to the network at time t_u and t_v. Without loss of generality, we assume $t_u > t_v$. Thus, at the time $t > t_u$, every newly added node $w \in V_i$ may connect to them. Using the law of total probability, we have

$$P_{uv} = \sum_{k=t_u}^{t} \alpha_i \frac{d_u(k)}{e_{0i}(k)} \frac{d_v(k)}{e_{0i}(k)}.$$

This equality holds since each node $w \in V_i$ that is added to the network at time $k \in [t_u, t]$ connects to node u with probability $\frac{d_u(k)}{e_{0i}(k)}$. Then, according to Lemma 1, we have

$$\begin{aligned} P_{uv} &= \sum_{k=t_u}^{t} \frac{\alpha_i}{\left(e_{0i}(k)\right)^2} d_u(t_u)\left(\frac{k}{t_u}\right)^{\gamma_{0i}} d_v(t_v)\left(\frac{k}{t_v}\right)^{\gamma_{0i}} \\ &= \alpha_i \frac{d_u(t_u)d_v(t_v)}{\left(\alpha_0 c_{0i} + \alpha_i c_{i0}\right)^2}\left(\frac{1}{t_v}\right)^{\gamma_{0i}}\left(\frac{1}{t_v}\right)^{\gamma_{0i}} \sum_{k=t_u}^{t} k^{2\gamma_{0i}-2}. \end{aligned}$$

Then, according to the result in Lemma 1, we have

$$P_{uv} = \frac{\alpha_i}{\alpha_0 c_{0i} + \alpha_i c_{i0}} \frac{d_u(t)d_v(t)}{e_{0i}(t)}.$$

According to Lemma 2, we know $d_v(t) = \frac{1}{c_i} D_v(t)$, where c_i is a constant that is independent of node v. Consequently, the probability P_{uv} satisfies $P_{uv} \propto D_u(t)D_v(t)$. Then, since it is satisfied that

$$\sum_{u\in V_0}\sum_{v\in V_0} P_{uv} = 2|E_i(t)|,$$

we can normalize P_{uv} to the expression given in the Lemma. And consequently, we complete the proof. □

With the above lemmas in hand, we are now ready to prove the recommendation probabilities of our proposed algorithm.

Theorem 2. *For a fixed node v in the network $\boldsymbol{G}(V_0) = [G(V_0|V_1), G(V_0|V_2)]$, let $R(v)$ denote a recommendation for node v returned by C2P algorithm. Then, we have*

$$\mathbb{P}(R(v) = u) = \frac{D_{v1}(t)D_{u2}(t) + D_{v2}(t)D_{u1}(t)}{2|E_2(t)|D_{v1}(t) + 2|E_1(t)|D_{v2}(t)},$$

where the probability captures both randomness in the network structure and that in the recommendation algorithm.

Proof. For convenience of illustration, we use some intermediate random variables as listed below:

- X_0, X_1 and $X_2 \in V_0$: source node, relay node and destination node of a two-hop path.
- H_1: the first hop that starts from X_0 and ends at X_1.
- H_2: the second hop that starts from X_1 and ends at X_2.

Given the above variables, we come to calculate the recommendation probability. According to C2P algorithm, the probability that node u is recommended to node v is proportional to the number of cross-layer two-hop paths between them, i.e.,

$$\mathbb{P}(R(v) = u) = \frac{|\{L|X_0 = v, X_2 = u\}|}{|\{L|X_0 = v\}|}, \tag{1}$$

where $|\{L|X_0 = v, X_2 = u\}|$ denotes the number of cross-layer two-hop paths that start from node v and end at node u, and $|\{L|X_0 = v\}|$ denotes the number of the paths that start from node v. In the remaining part of the paper, we omit the parameter t for convenience of notations. For example, we use E_i and D_{ui} instead of $E_i(t)$ and $D_{ui}(t)$, respectively.

Then, according to the results in Lemma 3, we can calculate Equation (1) as

$$\begin{aligned} &|\{L|X_0 = v\}| \\ &= \sum_{u\in V_0}\sum_{w\in V_0} |\{L|X_0 = v, X_1 = w, X_2 = u\}| \\ &= \sum_{u\in V_0}\sum_{w\in V_0} \frac{D_{v1}D_{w1}}{2|E_1|}\frac{D_{w2}D_{u2}}{2|E_2|} + \frac{D_{v2}D_{w2}}{2|E_2|}\frac{D_{w1}D_{u1}}{2|E_1|} \\ &= \sum_{w\in V_0} D_{w1}D_{w2} \cdot \frac{D_{v1}\cdot\sum_{u\in V_0} D_{u2} + D_{v2}\cdot\sum_{u\in V_0} D_{u1}}{4|E_1||E_2|} \\ &= \sum_{w\in V_0} D_{w1}D_{w2} \cdot \frac{|E_2|D_{u1} + |E_1|D_{u2}}{2|E_1||E_2|}. \end{aligned}$$

And similarly, we also have

$$\begin{aligned}&|\{L|X_0=v,X_2=u\}|\\&=\sum_{w\in V_0}|\{L|X_0=v,X_1=w,X_2=u\}|\\&=\sum_{w\in V_0}D_{w1}D_{w2}\cdot\frac{D_{v1}D_{u1}+D_{v1}D_{u2}}{4|E_1||E_2|}.\end{aligned}$$

Plug above two expressions into Equation (1) and we have

$$\mathbb{P}(R(v)=u)=\frac{D_{v1}D_{u2}+D_{v2}D_{u1}}{2|E_2|D_{v1}+2|E_1|D_{v2}},$$

which completes the proof. □

Results in Theorem 2 indicate that C2P algorithm considers cross-layer user relationships. We note that this characteristic promotes good performances in both acceptance and diversity, which we will discuss in details in the next part.

B. Performance Analysis on Acceptance and Diversity

We now come to analyze the performance of our proposed algorithm on two metrics – acceptance and diversity. We start from the acceptance and results show that our algorithm is the optimal one, as given in Theorem 3.

Theorem 3. *When make recommendations in the two-layered network* $\boldsymbol{G}(V_0)=[G(V_0|V_1),G(V_0|V_2)]$, *C2P algorithm is the optimal one in terms of the metric acceptance.*

Proof. According to Definition 1, acceptance of the proposed algorithm can be calculated as

$$A_{\boldsymbol{G}}=\frac{1}{|V_0|}\sum_{v\in V_0}\sum_{u\in V_0}\mathbb{P}(R(v)=u)\mathbb{P}(E(v)=u).$$

The determination of the optimal algorithm in acceptance can be formulated as a optimization problem, that is,

$$\begin{aligned}&\max\ A_{\boldsymbol{G}}\\&s.t.\quad\begin{cases}\sum_{u\in V_0}\mathbb{P}(R(v)=u)=1,\\ \mathbb{P}(R(v)=u)\geq 0.\end{cases}\end{aligned}$$

As previously defined, we have

$$\mathbb{P}(E(v)=u)=\frac{D_{v1}D_{u2}+D_{v2}D_{u1}}{2|E_2|D_{v1}+2|E_1|D_{v2}}.$$

Then, we search for $\mathbb{P}(R(v)=u)$ that can maximize $A_{\boldsymbol{G}}$. According to Hölder's Inequality, it is satisfied that

$$\sum_{i=1}^{n}a_ib_i\leq\left(\sum_{i=1}^{n}{a_i}^p\right)^{\frac{1}{p}}\left(\sum_{i=1}^{n}{b_i}^q\right)^{\frac{1}{q}},$$

where $\frac{1}{p}+\frac{1}{q}=1$ and the above equality holds if and only if $\exists c_1,c_2>0$ that satisfy $c_1a_i=c_2b_i$. Moreover, this summation is maximized when $p=2$ and $q=2$.

Consequently, we have that $\sum_{u\in V_0}\mathbb{P}(R(v)=u)\mathbb{P}(E(v)=u)$ is maximized when $\mathbb{P}(R(v)=u)=\mathbb{P}(E(v)=u)$, and it is satisfied that

$$\sum_{u\in V_0}\mathbb{P}(R(v)=u)\mathbb{P}(E(v)=u)\leq\sum_{u\in V_0}\mathbb{P}^2(E(v)=u).$$

Since the above result holds for all node $v\in V_0$, the metric $A_{\boldsymbol{G}}$ is maximized when $\mathbb{P}(R(v)=u)=\mathbb{P}(E(v)=u)$. With the result in Theorem 2, we have that in C2P algorithm $\mathbb{P}(R(v)=u)=\mathbb{P}(E(v)=u)$ and thus we complete the proof. □

In addition to the optimality in acceptance, C2P algorithm can also achieve a good performance in diversity. The corresponding results and proofs are given in Theorem 4 as below.

Theorem 4. *When make recommendations in the two-layered network* $\boldsymbol{G}(V_0)=[G(V_0|V_1),G(V_0|V_2)]$, *the diversity of C2P algorithm satisfies*

$$D_{\boldsymbol{G}}=\Theta(n),$$

where n *is the number of nodes in network* $\boldsymbol{G}(V_0)$.

Proof. According to the results in Theorem 2 and Lemma 2, the recommendation probability in C2P algorithm can also be expressed as

$$\mathbb{P}(R(v)=u)=\frac{d_{v1}d_{u2}+d_{v2}d_{u1}}{e_2d_{v1}+e_1d_{v2}}.$$

In C2P algorithm, the recommendations for all nodes $v\in V_0$ are independent of each other. Based on this fact, the average number of recommendations of node u in all n recommendations can be calculated as

$$\begin{aligned}&\sum_{v\in V_0}\mathbb{P}(R(v)=u)=\sum_{v\in V_0}\frac{d_{v1}d_{u2}+d_{v2}d_{u1}}{e_2d_{v1}+e_1d_{v2}}\\&=\frac{n}{e_2+e_1\frac{1}{n}\sum_{v\in V_0}\frac{d_{v2}}{d_{v1}}}d_{u2}+\frac{n}{e_1+e_2\frac{1}{n}\sum_{v\in V_0}\frac{d_{v1}}{d_{v2}}}d_{u1}.\end{aligned}$$

Using the result in Lemma 1, we have

$$\sum_{v\in V_0}\frac{d_{v1}}{d_{v2}}=\sum_{v\in V_0}\frac{d_{v1}(t_v)\left(\frac{t}{t_v}\right)^{\gamma_{01}}}{d_{v2}(t_v)\left(\frac{t}{t_v}\right)^{\gamma_{02}}}=\Theta(n).$$

For convenience, we denote $c_3=\frac{1}{n}\sum_{v\in V_0}\frac{d_{v1}}{d_{v2}}$. From the above equation we know that c_3 is a constant. In the following part, we conduct the calculation in order sense and thus the specific value of c_3 has no influence on the final result. Then, the average number of recommendations generated for node u can be expressed as

$$\begin{aligned}\sum_{v\in V_0}\mathbb{P}(R(v)=u)=&\frac{n}{e_2+e_1\frac{1}{c_3}}d_{u2}+\frac{n}{e_1+e_2c_3}d_{u1}\\=&c_4d_{u1}+c_5d_{u2}.\end{aligned}$$

The second equality holds since $e_1=\Theta(n)$ and $e_2=\Theta(n)$. And similarly, the specific values of constants c_4 and c_5 have no influence on the final result.

Recall that $\sum_{v\in V_0}\mathbb{P}(R(v)=u)$ is the average number of recommendations of node u. If the average number is greater than one, node u is recommended at least once among all n recommendations, which indicates that $u\in\cup_{v\in V_0}R(v)$. Consequently, we have

$$\begin{aligned}\mathbb{P}\left(u\in\cup_{v\in U}R(v)\right)=&\mathbb{P}\left(\sum_{v\in V_0}\mathbb{P}(R(v)=u)>1\right)\\=&\mathbb{P}\left(c_4d_{u1}+c_5d_{u2}>1\right).\end{aligned}$$

If $\gamma_{10} > \gamma_{20}$, we have $c_4 d_{u1} > c_5 d_{u2}$ and the latter one can be omitted compared to the former one. In this case, we have

$$\begin{aligned}\mathbb{P}\left(u \in \cup_{v\in U} R(v)\right) =& \mathbb{P}\left(c_4 d_{u1}(t_u)\left(\frac{t}{t_u}\right)^{\gamma_{01}} > 1\right)\\ =& \mathbb{P}\left(t_u < \frac{t}{(c_4 d_{u1}(t_u))^{\frac{1}{\gamma_{01}}}}\right)\\ =& (c_4 d_{u1}(t_u))^{\frac{1}{\gamma_{01}}},\end{aligned} \quad (2)$$

where the last equality holds since t_u is randomly and uniformly distributed in $(0, t)$. Equation (2) gives the result that $\mathbb{P}\left(u \in \cup_{v\in U} R(v)\right) = \Theta(1)$ is a constant. We have obtained the probability that node u is included in the recommendation lists. Then, consider all nodes $u \in V_0$ and we have

$$|\cup_{v\in U} R(v)| = \sum_{u\in V_0} \mathbb{P}\left(u \in \cup_{v\in U} R(v)\right) = \Theta(n).$$

And thus we complete the proof. □

VI. Experimental Measurements

In this section, we conduct experimental measurements on three real datasets and empirically validate our results.

A. Dataset Description

We conduct our experimental measurements on three fields of academic networks: Data Mining (DM), Machine Learning (ML) and Computer Vision (CV). All the three datasets are collected from Microsoft Academic Graph [13] – a heterogeneous graph containing publication records, citation relationships between the publications, as well as authors, institutions, journals, conferences, and fields of study. In each dataset, there are three types of entities: authors, papers and topics, denoted by the node sets A, P and T, respectively. Moreover, every entity in the dataset has a timestamp recording the time it joins the network, ranging from the year 1801 to 1976.

B. Simulation Setting

1) Network Construction: In each dataset, publication relationships between authors and papers are recorded and thus the graph $B(A, P)$ is naturally constructed. And we can obtain the graph $B(A, T)$ in a similar way. Then, according to the generation rule described in Section III-A, we can construct a two-layered network of authors, where one layer is paper-based, denoted by L_p; the other one is topic-based and we denote it by L_t. The statistical properties of the three networks are summarized in Table II.

2) Calculation of the Metrics: In the calculation of acceptance, an essential issue is how to quantify a user's demand. In our simulation, we model it by the user's future behaviors. Specifically, we involve the network topology in a particular year as the input information and conduct recommendations according to it. Then, we quantify a user's demand by the new connections created in the following year. For example, if the algorithm works on the network of year 1975, we use the new connections generated in year 1976 to calculate the user's demands. We note that it is reasonable since the new connections are spontaneously created by the user according to his own will, and consequently can well represent his demand. Additionally, diversity can be directly calculated according to the users' recommendation lists.

TABLE II
Statistical Properties of Datasets.

	# of Nodes	# of Edges in L_p	# of Edges in L_t
DM	$47,634$	$41,321$	$873,664$
ML	$51,339$	$44,009$	$12,158,310$
CV	$67,972$	$76,460$	$12,105,641$

3) Baseline Algorithms Involved in Performance Comparison: To evaluate the performance of our proposed algorithm, we include three additional algorithms to make the comparison. We briefly introduce them as follows:

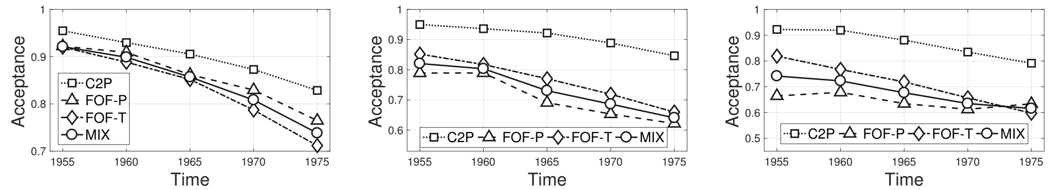
- *Friend of Friend algorithm in Paper-based layer (FOF-P):* An algorithm that works in L_p by making recommendations from all the two-hop neighbors of the user with probabilities proportional to the number of paths connecting to it.
- *Friend of Friend algorithm in Topic-based layer (FOF-T):* A recommendation algorithm that has a similar implementation principle as that in FOF-P but works in L_t.
- *Mixed algorithm (MIX):* An algorithm that provides recommendation by mixing the items returned by FOF-P and that returned by FOF-T together.

In the simulation, for our proposed algorithm and all the above ones, we set the length of recommendation list as $k = 1$.

C. Performance Analysis

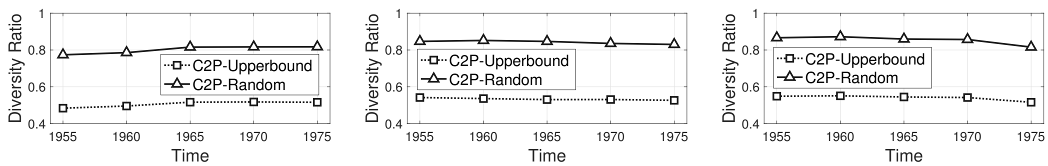
Performance on Acceptance: We conduct the simulation at 5 test points, ranging from the year 1955 to 1975 with internal 5. In each test point, we generate recommendations items by our proposed algorithm and three additional ones. Simulation results are given in Figure 4. Someone may wonder that why the value of acceptance decreases with test years. This is due to the reason that with the increase of test years, network size grows and network structure becomes more complicated, which makes recommendation more difficult. However, we can still observe from the results that, C2P algorithm outperforms the other three ones at all test points. Particularly, the acceptance gain achieves 38% in the ML dataset at year 1975.

Performance on Diversity: Similarly, the measurements on the metric diversity are also conducted at 5 test points, ranging from year 1955 to 1975. To evaluate the performance, we make the comparison between our proposed algorithm and the other two cases. The first case is theoretical upperbound that corresponds to the maximum value of diversity. This case can only be achieved if and only if all the recommendation lists are different, and the corresponding diversity equals to n, i.e., the number of users in the network. The second one is random case, where we randomly and uniformly recommend user an item. We include this case since that the theoretical upperbound is an ideal one that cannot be achieved in most cases. The results are given in Figure 5, where y-coordinates denote the diversity ratio of C2P algorithm to the upperbound/random case. We can observe that the two ratios are both in the order of constant, which exactly verifies our theoretical results.



(a) Data Mining. (b) Machine Learning. (c) Computer Vision.

Fig. 4. Average acceptance of different algorithms, where x-coordinates are test years followed by an increasing order of network sizes.



(a) Data Mining. (b) Machine Learning. (c) Computer Vision.

Fig. 5. Diversity ratios under two cases, where C2P-Upperbound denotes the diversity ratio of C2P to the theoretical upperbound, C2P-Random denotes that of C2P to random recommendations, and x-coordinates are test years followed by an increasing order of network sizes.

VII. Conclusion

In this paper, we propose a novel algorithm named Cross-layer 2-hop Path (C2P) algorithm. The algorithm recommends a user's cross-layer two-hop neighbors to him and thus, each recommendation item is a combination of user relationships in both two layers that can better meet user demands. The proposed algorithm is proved to be efficiently implementable. We evaluate its performance by two metrics, i.e., acceptance and diversity, and the results show that C2P algorithm is optimal in terms of acceptance and for diversity, it is in the same order of theoretical upperbound. And finally, the effectiveness of the algorithm is validated on three real datasets.

Acknowledgment

This work was supported by NSF China (No. 61532012, 61325012 and 61602303).

References

[1] J. Chen, W. Geyer, C. Dugan, M. Muller, and I. Guy, "Make new friends, but keep the old: recommending people on social networking sites," in *Proc. ACM Human Factors in Computing Systems*, 2009, pp. 201–210.

[2] J. Su, A. Sharma, and S. Goel, "The effect of recommendations on network structure," in *Proc. ACM WWW*, 2016, pp. 1157–1167.

[3] S. Huang, J. Zhang, L. Wang, and X.-S. Hua, "Social friend recommendation based on multiple network correlation," *IEEE Trans. Multimedia*, vol. 18, no. 2, pp. 287–299, 2016.

[4] X. He, M. Gao, M.-Y. Kan, and D. Wang, "Birank: Towards ranking on bipartite graphs," *IEEE Trans. Knowledge and Data Engineering*, vol. 29, no. 1, pp. 57–71, 2017.

[5] P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, and R. Zadeh, "Wtf: The who to follow service at twitter," in *Proc. ACM WWW*, 2013, pp. 505–514.

[6] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proc. ACM SIGKDD*, 2011, pp. 448–456.

[7] Z. Wang, J. Liao, Q. Cao, H. Qi, and Z. Wang, "Friendbook: a semantic-based friend recommendation system for social networks," *IEEE Trans. Mobile Computing*, vol. 14, no. 3, pp. 538–551, 2015.

[8] L. E. Chatzieleftheriou, M. Karaliopoulos, and I. Koutsopoulos, "Cachingaware recommendations: Nudging user preferences towards better caching performance," in *Proc. IEEE INFOCOM*, 2017.

[9] Z. Wang, Y. Zhang, Y. Li, Q. Wang, and F. Xia, "Exploiting social influence for contextaware event recommendation in eventbased social networks," in *Proc. IEEE INFOCOM*, 2017.

[10] D. Serbos, S. Qi, N. Mamoulis, E. Pitoura, and P. Tsaparas, "Fairness in package-to-group recommendations," in *Proc. ACM WWW*, 2017, pp. 371–379.

[11] B. Oselio, A. Kulesza, and A. O. Hero, "Multi-layer graph analysis for dynamic social networks," *IEEE J. Selected Topics in Signal Processing*, vol. 8, no. 4, pp. 514–523, 2014.

[12] M. De Domenico, C. Granell, M. A. Porter, and A. Arenas, "The physics of spreading processes in multilayer networks," *Nature Physics*, vol. 12, no. 10, pp. 901–906, 2016.

[13] *Microsoft Academic Graph*, 2017, https://www.microsoft.com/en-us/research/project/microsoft-academic-graph/.

[14] S. Chang, Y. Zhang, J. Tang, D. Yin, Y. Chang, M. A. Hasegawa-Johnson, and T. S. Huang, "Streaming recommender systems," in *Proc. ACM WWW*, 2017, pp. 381–389.

[15] X. Yang, Y. Guo, and Y. Liu, "Bayesian-inference-based recommendation in online social networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 24, no. 4, pp. 642–651, 2013.

[16] F. M. F. Wong, Z. Liu, and M. Chiang, "On the efficiency of social recommender networks," *IEEE/ACM Trans. Networking*, vol. 24, no. 4, pp. 2512–2524, 2016.

[17] J. Lang and P. K. Skowron, "Multi-attribute proportional representation." in *Proc. ACM AAAI*, 2016, pp. 530–536.

[18] L. Fu, J. Zhang, and X. Wang, "Evolution-cast: Temporal evolution in wireless social networks and its impact on capacity," *IEEE Trans. Parallel and Distributed Systems*, vol. 25, no. 10, pp. 2583–2594, 2014.

[19] Z. Wang, D. Zhang, X. Zhou, D. Yang, Z. Yu, and Z. Yu, "Discovering and profiling overlapping communities in location-based social networks," *IEEE Trans. Systems, Man, and Cybernetics: Systems*, vol. 44, no. 4, pp. 499–509, 2014.

[20] P. He, J. Wang, W. Feng, and L. Li, "Exploring influential nodes using multi-attribute information," in *Proc. IEEE ICNC*, 2015, pp. 473–478.

[21] P. Bródka, P. Kazienko, K. Musiał, and K. Skibicki, "Analysis of neighbourhoods in multi-layered dynamic social networks," *International J. Computational Intelligence Systems*, vol. 5, no. 3, pp. 582–596, 2012.

[22] M. Magnani and L. Rossi, "The ml-model for multi-layer social networks," in *Proc. IEEE ASONAM*, 2011, pp. 5–12.

[23] J. Hao, S. Cai, Q. He, and Z. Liu, "The interaction between multiplex community networks," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 21, no. 1, p. 016104, 2011.

[24] S. Lattanzi and D. Sivakumar, "Affiliation networks," in *Proc. ACM STOC*, 2009, pp. 427–434.

[25] R. Hermoso, S. Ilarri, R. Trillo, and M. del Carmen Rodríguez-Hernández, "Push-based recommendations in mobile computing using a multi-layer contextual approach," in *Proc. ACM Advances in Mobile Computing and Multimedia*, 2015, pp. 149–158.

[26] L. C. Totti, P. Mitra, M. Ouzzani, and M. J. Zaki, "A query-oriented approach for relevance in citation networks," in *Proc. ACM WWW*, 2016, pp. 401–406.

[27] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[28] M. E. Newman, "Clustering and preferential attachment in growing networks," *Physical review E*, vol. 64, no. 2, p. 025102, 2001.

[29] G. Adomavicius and Y. Kwon, "Improving aggregate recommendation diversity using ranking-based techniques," *IEEE Trans. Knowledge and Data Engineering*, vol. 24, no. 5, pp. 896–911, 2012.