Fusing Diversity in Recommendations in Heterogeneous Information Networks

Sharad Nandanwar
Indian Institute of Science
sharadnandanwar@csa.iisc.ernet.in

Aayush Moroney
Indian Institute of Science
aayush.moroney@csa.iisc.ernet.in

M. N. Murty
Indian Institute of Science
mnm@csa.iisc.ernet.in

ABSTRACT

In the past, hybrid recommender systems have shown the power of exploiting relationships amongst objects which directly or indirectly effect the recommendation task. However, the effect of all relations is not equal, and choosing their right balance for a recommendation problem at hand is non-trivial. We model these interactions using a Heterogeneous Information Network, and propose a systematic framework for learning their influence weights for a given recommendation task. Further, we address the issue of redundant results, which is very much prevalent in recommender systems. To alleviate redundancy in recommendations we use Vertex Reinforced Random Walk (a non-Markovian random walk) over a heterogeneous graph. It works by boosting the transitions to the influential nodes, while simultaneously shrinking the weights of others. This helps in discouraging recommendation of multiple influential nodes which lie in close proximity of each other, thus ensuring diversity. Finally, we demonstrate the effectiveness of our approach by experimenting on real world datasets. We find that, with the weights of relations learned using the proposed non-Markovian random walk based framework, the results consistently improve over the baselines.

CCS CONCEPTS

•Information systems →Recommender systems; Social recommendation; *Semi-structured data; Social networks; Learning to rank;*

KEYWORDS

Diversity in Recommendation; Multivariate Random Walk; Vertex Reinforced Random Walk; Heterogeneous Information Network

ACM Reference format:
Sharad Nandanwar, Aayush Moroney, and M. N. Murty. 2018. Fusing Diversity in Recommendations in Heterogeneous Information Networks. In *Proceedings of WSDM'18, February 5–9, 2018, Marina Del Rey, CA, USA,* 9 pages.
DOI: http://dx.doi.org/10.1145/3159652.3159720

1 INTRODUCTION

Recommender systems deal with the task of identifying objects with which a user is likely to interact in the near future. The objects could be movies to watch in case of user-movie recommendation, products to purchase in user-product recommendation, users to friend in user-user recommendation and so on. Most of the recommender systems focus on cases where users are the recipients of recommendations. We consider a more general definition where both recommendations and recipients of recommendations could be of any type. For example, matching the responsible genes to a particular disease can be casted as a recommendation problem [7]. Formally, we define the recommendation problem as follows

Definition 1.1. Given a set of recipient objects $\mathcal{S}$ of type s, and a set of recommendation objects $\mathcal{T}$ of type t, along with the snapshot at time T of the preference matrix A_{st} encoding past preferences of objects in $\mathcal{S}$ for objects in $\mathcal{T}$, the recommendation problem is to find a subset of $\mathcal{T}$ for each object in $\mathcal{S}$ with which it is likely to interact at time $(T + 1)$.

Collaborative filtering, a key technique used in recommender systems primarily works by aggregating and ranking the preferences of users with similar past behavior. Further, in a series of works [5, 19, 25, 28] it was observed that hybrid models utilizing auxiliary information about the recommendation object and the recipient object tend to do better. This auxiliary information is generally present in the form of interactions with objects of other types, which directly or indirectly influence the recommendations under consideration. More recently, Heterogeneous Information Networks (HIN) have become popular tools for modeling such interactions. Heterogeneous Information Network models the data using a graph where nodes represent the objects and edges depict the relationships amongst objects as in the underlying data. It is to be noted that unlike homogeneous network HIN can have more than one type of object and more than one type of relationships amongst them. A more detailed and formal definition of HIN is given by Sun et al. [20].
Markovian-walk based techniques like Personalized-Page Rank, Katz similarity etc. offer a powerful way of computing similarity in a graph. However, finding this similarity becomes non-trivial when accounting for multiple types of relations. The extent to which the similarity between one type of object influences the similarity between other type varies for each relation. We explain this using a toy restaurant information network shown in Fig. 1. The figure shows a network consisting of objects of type *User*, *Restaurant*, *Locality*, and *Cuisine*. Consider the case where restaurants are to be suggested to user u_4. Conventional collaborative filtering would fail here as the restaurants preferred in the past by u_4 are not available, leading to a cold start problem. However, based on the locality of user u_4 it is possible to suggest restaurant r_3. Similarly by looking at the preferences made by friends of u_4 restaurants r_2 and r_4 can be suggested. The final set of recommendations are determined based on how these preferences from various sources are combined.

WSDM'18, February 5–9, 2018, Marina Del Rey, CA, USA
 ISBN 978-1-4503-5581-0/18/02…$15.00
DOI: http://dx.doi.org/10.1145/3159652.3159720

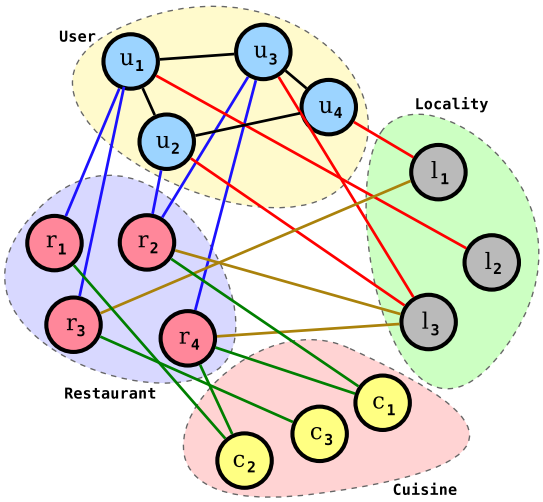

Figure 1: A toy Restaurant Information Network

One of the pressing issues in recommender systems is to do away with monotonous yet relevant recommendations. Techniques like collaborative filtering emphasize on obtaining the recommendations lying in the query proximity. However, it has been noted [27] that the utility of such recommendations is questionable. Too many of such similar results can degrade the User experience. This is easy to follow from common human psychology point of view as the system won't offer novelty for seasoned customers.

To further motivate the need of diversity in recommendation, we illustrate using our running restaurant example. In the given scenario, suppose it is chosen to recommend restaurants by exploiting the preferences expressed by friends of the user. In such a case the suggestions would be r_2 followed by r_4. However, it is worth noting that both r_2 and r_4 offer the same cuisine and share the same locality. It is very much intuitive to say that only one out of these two recommendations should be enough.

In this paper, we propose a random walk based approach for making recommendation in the underlying heterogeneous information network. To account for multiple types of relationships we create a parameterized model, where each parameter describes the role played by the corresponding relation in recommendation. For learning these parameters, we deploy a probabilistic optimization framework, wherein we systematically minimize the cross entropy loss using the available ground truth. Conventional random walk based approaches like personalized page-rank, etc. use a Markovian-random walk over the graph. And, the stationary distribution obtained as a consequence of this Markovian-random walk is used for recommendation. However, this could lead to multiple recommendations which are in close vicinity of each other. To provide diversity, it is favorable to have a non-Markovian Random walk which uses a time variant transition matrix. We propose to use vertex reinforced random walk (VRRW) for this. After each unit time interval, VRRW updates the transition probabilities using a monotonically increasing function of current arrival probabilities. This is done to further encourage the transition to nodes which currently have high arrival probabilities. We summarize our key contributions below:

- *We make use of a sublinear Vertex Reinforced Random Walk, for achieving diversity in recommendations.*
- *We describe a principled approach for learning parameters of a multivariate non-Markovian random walk over a heterogeneous graph, by minimizing the cross entropy loss.*
- *We propose a stochastic gradient descent based algorithm for the same.*
- *We show the effectiveness of the proposed approach over state-of-the-art techniques using real world datasets.*

The rest of the paper is organized as follows; we briefly survey the related work in Sec. 2. Next, we introduce the necessary background, in Sec. 3, which is helpful for understanding this paper. The vertex reinforced random walk (VRRW) is introduced in Sec. 4. In Sec. 5 we describe our framework for learning the importance of different relations in a recommendation task. We present experimental results in Sec. 6, followed by conclusion in Sec. 7. We make the source code and datasets used in our experiments publicly available at https://github.com/sharadnandanwar/DivFuse.

2 RELATED WORK

Diversity. Diversity in ranking has been studied very well in machine learning and data mining literature [17]. However, these studies have been largely concerned with the problem of ranking the web documents. Carbonell et al. [1] were first to study the interplay of relevance and novelty for ranking and summarization of text documents. They proposed to maximize marginal relevance for top ranked results. Marginal relevance is high if the document is relevant to the query, while at the same time bearing minimal similarity to other ranked results. Later Zhai et al. [26] in their work undertook the study as subtopic retrieval problem, with an aim to maximize the number of diverse subtopics amongst ranked results. Based on the definition of diversity, the subsequent works in this direction can be broadly classified into 3 paradigms namely *i)* content based, *ii)* novelty based, and *iii)* coverage based [3].

Diversity on Graphs. `Grasshopper`[29], one of the earliest works in diversified ranking on graphs, proposes a sequential algorithm to compute a diverse set of nodes. At each intermediate stage the selected nodes are switched to the absorbing state. `DivRank` [10] proposed by Mei et al. introduced a framework for diversified ranking in networks. Prior approaches like `Personalized Page-Rank` [6] considered a random walk model based on a Markovian process. Unlike this, `DivRank` considers a vertex-reinforced random walk model which is based on a non-Markovian process. The number of chances of arriving at a node increases with the number of visits to that node, in this way a single node absorbs the score of its strongly connected neighbors leading to a set of diverse nodes having high ranks. An optimization perspective to the same problem is taken in [22], wherein a goodness measure is proposed to capture relevance and diversity along with an algorithm to maximize it. Dubey et al [4] aim to find a diversified set of centers (nodes) s.t. the conductance from these centers to the rest of graph has maximum entropy. Citing to the cubic complexity of the earlier approaches, Li et al. [9] introduced an efficient greedy algorithm with liner time and space complexity.

Recommendations in Heterogeneous Networks. In homogeneous networks the problem of recommendation has been studied

under various paradigms including ranking [8, 14], semi-supervised learning in graphs [11, 12] etc. Unlike homogeneous networks, each of the multiple relation types present in a heterogeneous information networks carries a different semantic meaning. In [13] it was shown that application of techniques like `Page-Rank` [14], `HITS` [8] etc. (which were developed for homogeneous networks) to HIN, yields absurd results. Sun et al. [21] proposed `PathSim`, a meta-path framework, for network based similarity search. However, it only considered symmetric meta-paths. In [18] a generalized framework `HeteSim` was proposed which allows to compute similarity between any two types of objects using any possible meta-path. Although these works studied that different meta-paths lead to different rankings, they do not provide a mechanism for choosing the meta-paths. Relationship prediction models were introduced in [19] and [25] which aim to learn the role of different meta-paths in relationship prediction task. Guerraoui et al [5] extend meta-path framework for cross domain recommendation by learning alter-ego profiles.
In line with random surfer model, Pham et al. [16] proposed to use multivariate Markov chain to calculate node proximity w.r.t. a given query node along with an optimization framework for learning the influence weights between different types of objects.

3 PRELIMINARIES

3.1 Random Walk over Graph

Consider an undirected weighted graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ with set of vertices $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$, and set of edges $\mathcal{E} \subset \mathcal{V}\times\mathcal{V}\times\mathbb{R}^+$; where each edge is a 3-tuple (v_i, v_j, w_{ij}) s.t. $v_i, v_j \in \mathcal{V}$ and $w_{ij} \in \mathbb{R}^+$. Then, the adjacency matrix $\mathbf{A}$ corresponding to $\mathcal{G}$ is defined as,

$$A_{ij} = \begin{cases} w_{ij} & \text{if } (v_i, v_j, w_{ij}) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases}.$$

Given a graph $\mathcal{G}(\mathcal{V},\mathcal{E})$, and an initial vertex x_0 ($\in \mathcal{V}$); a random walk over the graph makes a random choice of transition to, one of its neighboring vertices, x_1; from x_1 to, one of its neighboring vertices, x_2 and so on. The transition probabilities to the neighboring vertices are captured using a transition matrix P, which is derived from the adjacency matrix as follows,

$$P_{ij} = \frac{A_{ij}}{\sum_j A_{ij}}$$

It is worth noting here that the choice of transition to x_2 when at vertex x_1 does not depend on the initial choice x_0. Similarly, the transition made from x_2 will not depend on x_1 and so on. This tends to form a Markov Chain [2]. Given the present, future doesn't depend upon the past.
For the random walk on $\mathcal{G}$ we assume that the initial vertex x_0 is drawn from some distribution Π^0. Likewise, at time t, we denote this distribution using Π^t where $\Pi^t(v_0)$ is the probability of being at vertex v_0 at time t. We call Π^t as the state vector at time t. The distribution of the random walk is then expressed as,

$$\Pi^{t+1} = P^\mathsf{T}\Pi^t$$

In a limiting sense (after infinitely many iterations), the probability of being at vertex v_i does not depend on the initial choice. This limiting distribution is known as stationary distribution and is denoted by Π^*.

3.2 Heterogeneous Information Network

Very often graphs involve more than one kind of objects interacting with each other. We call such a graph having multiple types of objects and relations as Heterogeneous Information Network (HIN) which is defined [20] as follows,

Definition 3.1. An Information Network is defined as a directed graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ with an object type mapping function $\tau : \mathcal{V} \to \mathcal{O}$ and a link type mapping function $\varnothing : \mathcal{E} \to \mathcal{R}$ where each object $v \in \mathcal{V}$ belongs to one particular object type $\tau(v) \in \mathcal{O}$, each link $e \in \mathcal{E}$ belongs to a particular relation $\varnothing(e) \in \mathcal{R}$. When the type of objects $|\mathcal{O}| > 1$ or the type of relations $|\mathcal{R}| > 1$, then the network is called **Heterogeneous Information Network**.

If two links belong to the same relation type, the two links share the same starting object type as well as the ending object type. However, it is to be noted that the reverse is not true. Two relations can have same starting as well as ending objects and can yet be different. For example, *user*-visits-*restaurant*, and *user*-likes-*restaurant* are two different relations.

3.3 Multivariate Random Walk over HIN

In random walks over a homogeneous graph, each state is associated with a single node or vertex. However, unlike this, in HIN each state is associated with multiple objects one from each type. Hence, for a random walk over HIN each state is a tuple with the size same as that of the number of object types. For example, in restaurant information network illustrated in Fig. 1 each state during random walk consists of one object each of type *restaurant*, *user*, *locality*, and *cuisine*.
For a HIN $\mathcal{G}(\mathcal{V},\mathcal{E})$, we assume that the set of object types is $\mathcal{O} = \{o_1, o_2, \ldots, o_k\}$. Further, for each object type, let the set of nodes be $O_1, O_2, \ldots, O_k$, such that $O_1 \bigcup O_2 \bigcup \ldots \bigcup O_k = \mathcal{V}$. Thus, each state s of the random walk belongs to $O_1 \times O_2 \times \ldots \times O_k$. We use Π_j^t (a $|O_j| \times 1$ vector) to denote the state vector of nodes, with object type o_j at time instance t. The state vector of type o_j at time $(t+1)$ is influenced by state vectors at time t of all other object types with which it shares a relation present in $\mathcal{R}$. The transition matrix for each relation is created using the corresponding adjacency matrix of relation in a manner similar to that of the homogeneous graph. We denote a relation $\mathcal{R}$ using (r, o_i, o_j) where r is the unique relation identifier, o_i is the starting object type and o_j is the ending object type. Further, to denote the transition matrix of the same we use P_r. The rule of random walk is then,

$$\Pi_j^{(t+1)} = \sum_{(r,i,j)\in\mathcal{R}_{*,j}} \alpha_r \, P_r^\mathsf{T} \, \Pi_i^t \qquad \text{for } j = 1, 2, \ldots, k \tag{1}$$

$$\text{s.t.} \quad \alpha_r \geq 0 \qquad \text{for all } r \in \mathcal{R}, \text{ and}$$

$$\sum_{(r,i,j)\in\mathcal{R}_{*,j}} \alpha_r = 1 \qquad \text{for } j = 1, 2, \ldots, k$$

where, $\mathcal{R}_{*,j}$ ($\subset \mathcal{R}$) is set of relations having o_j as the ending object. Eqn. 1 expresses the state probability distribution at time $(t+1)$ as a weighted combination of $P_r^\mathsf{T}\,\Pi_i^t$ for all $(r,i,j) \in \mathcal{R}_{*,j}$. The constraints on α_r ensure that the state vector at time $(t+1)$ remains a valid probability distribution for each object type. For a given HIN, if the relations can be uniquely identified using starting and ending object (i.e. there does not exist any two relations sharing

same starting and ending object) then, the Eqn. 1 can be expressed in matrix notation as follows

$$\begin{bmatrix} \Pi_1^{t+1} \\ \Pi_2^{t+1} \\ \vdots \\ \Pi_k^{t+1} \end{bmatrix} = \begin{bmatrix} \alpha_{11}P_{11}^\top & \alpha_{12}P_{12}^\top & \dots & \alpha_{1k}P_{1k}^\top \\ \alpha_{21}P_{21}^\top & \alpha_{22}P_{22}^\top & \dots & \alpha_{2k}P_{2k}^\top \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{k1}P_{k1}^\top & \alpha_{k2}P_{k2}^\top & \dots & \alpha_{kk}P_{kk}^\top \end{bmatrix} \begin{bmatrix} \Pi_1^t \\ \Pi_1^t \\ \vdots \\ \Pi_k^t \end{bmatrix} \tag{2}$$

$$\Pi^{t+1} = \widetilde{\mathbf{P}}\,\Pi^t$$

In the above, for each relation (r, i, j) we use ij as subscript instead of r, as it can uniquely identify the relation as mentioned above. In case, where relations could not be uniquely identified using starting and ending object, each sub-matrix in $\widetilde{\mathbf{P}}$ is a weighted combination of transition matrices sharing corresponding starting and ending objects. It can be verified that Eqn. 2 has a unique solution which is a positive vector corresponding to eigenvalue of 1. For further details on this, we refer readers to the text [2]. Unlike homogeneous case, Π^t here is not a probability distribution, however, $\Pi_j^t \ \ \forall j$, is a probability distribution vector.

4 NON-MARKOVIAN RANDOM WALK

We now introduce the non-Markovian random walk, which we exploit in our work to ensure diversity.

Conventional random walk models like Page-Rank find a stationary distribution assuming that the transition probabilities remain constant over the entire period of time. In practice, this may not be true. For example, consider two restaurants which are open at the same time in a locality, and offer same cuisine. The first one, however offers a better deal for the price customers pay. It is quite natural, that even though on initial days both restaurants were witnessing same number of footfalls, slowly there will be a noticeable change. The first restaurant will eat away the business of the second, and there will be a significant difference in the number of footfalls. In line with this argument, it is justified to have a transition matrix which changes with time.

In order to achieve this we propose to use Vertex-Reinforced Random Walk (VRRW). First introduced by Pemantle [15], the VRRW is based on the idea that the future transition probabilities are influenced by the number of times the ending node has been visited in the past. We first explain VRRW for a homogeneous network $\mathcal{G}(\mathcal{V}, \mathcal{E})$ and then generalize it to HIN. For a node $v \in \mathcal{V}$, let the number of times it has been visited by random walk $(x_0, x_1, \dots, x_t)$ up to time t be $Z^t(v)$, then

$$\begin{aligned} Z^t(v) &= Z^{(t-1)}(v) + \mathbb{1}(x_t = v) \\ &= Z^0(v) + \sum_{i=0}^{t} \mathbb{1}(x_i = v) \end{aligned} \tag{3}$$

where, $Z^0(v)$ is the initial visit count of vertex v, before the random walk has started. In our study, we consistently take $Z^0(v)$ as zero. The transition probabilities for a vertex reinforced random walk $(x_0, x_1, \dots, x_t)$ are then updated as follows

$$P^{t+1}(v_i, v_j) = \frac{P^0(v_i, v_j) Z^t(v_j)}{\sum_{v_k \in \mathcal{V}} P^0(v_i, v_k) Z^t(v_k)} \tag{4}$$

where P^0 is the initial reinforcement matrix at time $t = 0$. It is difficult to compute $Z^t(v_j)$ at time t based on a single instance of random walk. In [10], it was proposed to compute $Z^t(v_j)$ based on multiple instances of random walk starting with same configuration at time $(t-1)$. This amounts to using $E[Z^t(v_j)]$ in place of $Z^t(v_j)$. Taking expectation of Eqn. 3 on both sides, we have

$$\begin{aligned} E[Z^t(v)] &= E[Z^0(v)] + \sum_{i=0}^{t} \Pi^i(v) \\ &= \sum_{i=0}^{t} \Pi^i(v) \quad \{\text{putting } Z^0(v) = 0\} \end{aligned} \tag{5}$$

Then, Eqn. 4 can be accordingly expressed as,

$$P^{t+1}(v_i, v_j) = \frac{P^0(v_i, v_j) E[Z^t(v_j)]}{\sum_{v_k \in \mathcal{V}} P^0(v_i, v_k) E[Z^t(v_k)]} \tag{6}$$

Where, $E[Z^t(v)]$ is computed using Eqn. 5.

The reinforcement defined in Eqn. 4 is called as linear reinforcement. In Linear VRRW, the modified transition probabilities are linearly proportional to the number of times the vertex has been visited in the past. It was shown, in [23], that in a linear VRRW there is only a finite set of nodes which is infinitely visited with positive probability. This characteristic is not desired from a recommendation perspective as localizing on a finite set of nodes will eventually lead to lack of unseen or new recommendations.

Deviating from [10], we propose to use sublinear vetex reinforced random walks here. Transition probabilities in sublinear VRRW are defined as

$$P^{t+1}(v_i, v_j) = \frac{P^0(v_i, v_j) \left(Z^t(v_j)\right)^\rho}{\sum_{v_k \in \mathcal{V}} P^0(v_i, v_k) \left(Z^t(v_k)\right)^\rho} \tag{7}$$

where $0 < \rho < 1$. It is to be mentioned that the same is known as superlinear VRRW when $\rho > 1$. For the sublinear case, [24] shows that the random walk almost surely visits, infinitely many nodes infinitely often.

Similar to the case of linear reinforcement, for sublinear reinforcement $\left(Z^t(v_k)\right)^\rho$ can be approximated using $E[\left(Z^t(v_k)\right)^\rho]$, which can be computed as follows,

$$\begin{aligned} E\left[\left(Z^t(v)\right)^\rho\right] &= E\left[\left(Z^0(v)\right)^\rho\right] + \sum_{i=0}^{t} \left(\Pi^i(v)\right)^\rho \\ &= \sum_{i=0}^{t} \left(\Pi^i(v)\right)^\rho \quad \{\text{putting } Z^0(v) = 0\} \end{aligned} \tag{8}$$

Now that we have explained the sublinear VRRW for homogeneous network, we extend it to more general HIN. Given a HIN with set of object types $\mathcal{O} = \{o_1, o_2, \dots, o_k\}$ and set of relation types $\mathcal{R}$, then for each relation type $(r, o_a, o_b) \in \mathcal{R}$, the transition matrix corresponding to sublinear VRRW is defined as follows,

$$P_r^{t+1}(v_i, v_j) = \frac{P_r^0(v_i, v_j)\, E\left[\left(Z^t(v_j)\right)^\rho\right]}{\sum_{v_k \in \mathcal{V}} P_r^0(v_i, v_k)\, E\left[\left(Z^t(v_k)\right)^\rho\right]} \tag{9}$$

and, $E\left[\left(Z^t(v)\right)^\rho\right] = \sum_{i=0}^{t}\left(\Pi_b^i(v)\right)^\rho$, where b is the object type obtained using the mapping $\tau : \mathcal{V} \rightarrow \mathcal{O}$ i.e. $\tau(v) = b$.

5 LEARNING THE PARAMETERS

Sec. 3.3 explains the multivariate random walk over HIN. However, determining the parameter α_r for each relation in $\mathcal{R}$ is a non-trivial task. In this section, we propose a principled approach to learn these parameters using the given snapshot of a HIN. Recalling our recommendation task, it is worth noting that the plain random walk model explained in Sec. 3.3 does not help here because it does not capture personalization. Random walk with restart provides a mechanism to personalize the recommendations for a given query. Random walk with restart at each step, either transitions to one of the randomly chosen neighbors, or jumps back to one of the query nodes. In a random walk over HIN $\mathcal{G}(\mathcal{V},\mathcal{E})$, with set of object types $\mathcal{O}$ and set of relation types $\mathcal{R}$; for object type o_j, we use β_j to denote the probability of jumping to one of the initial query nodes. The rule of random walk expressed in Eqn. 1 is then accordingly modified as

$$\begin{aligned}\Pi_j^{(t+1)} = &\sum_{(r,i,j)\in\mathcal{R}_{*,j}} \alpha_r \left(P_r^t\right)^\top \Pi_i^t + \beta_j q_j \qquad \text{for } j = 1,2,\ldots,k\\ \text{s.t.}\quad &\alpha_r \geq 0 \qquad \text{for all } r \in \mathcal{R},\\ &\beta_j \geq 0 \qquad \text{for } j = 1,2,\ldots,k\\ &\left(\sum_{(r,i,j)\in\mathcal{R}_{*,j}} \alpha_r\right) + \beta_j = 1 \qquad \text{for } j = 1,2,\ldots,k\end{aligned} \tag{10}$$

where q_j is the probability vector corresponding to initial query distribution for object type o_j. The transition matrix P_r^t is for sublinear VRRW and is updated using Eqn. 9.

5.1 Cross Entropy Loss Minimization

We next focus our attention on learning the parameters α's and β's in Eqn. 10. For this, we introduce a probabilistic learning framework based on cross entropy cost function. To recall, for a given set of parameter values α's and β's there exists a stationary probability distribution Π_j^* for each object type o_j. The stationary distribution is obtained by iterative application of Eqn. 10. Although, the stationary distribution Π_j^* is obtained for all object types, at the end we are only interested in the distribution of target object type (the object type concerning the recommendations). We refer to this distribution as Π_{target}^*. However, for the sake of brevity, here in this section 5.1 we make an abuse of notation, and will write Π instead of Π_{target}^*.

For a query $\mathbf{q} = \{q_1, q_2, \ldots, q_k\}$, we use Π_q to denote the stationary distribution, and Θ_q to denote the observed response for the target objects. It is to be mentioned here that in majority of the cases the observations are only available in the form of binary responses (whether the recommendation was useful or not) but in a few cases like movie recommendation to users, a rating on a ordinal scale is also available.

Given Π_q and Θ_q for a query $\mathbf{q}$, for each pair (v_i, v_j) of target objects, we define the following

$$\begin{aligned}\overline{p_{ij}} &= \mathbb{1}\left(\Theta_q(v_i) > \Theta_q(v_j)\right)\\ o_{ij} &= \Pi_q(v_i) - \Pi_q(v_j), \quad \textit{and}\\ p_{ij} &= \frac{e^{\lambda o_{ij}}}{1+e^{\lambda o_{ij}}}, \qquad \text{where } \lambda \text{ is a const.} > 0\end{aligned}$$

$\overline{p_{ij}}$ is the actual probability of node v_i being ranked higher than node v_j. However, unfortunately it is difficult to obtain this probability for each pair. To address this, we have defined $\overline{p_{ij}}$ as 1 if v_i is having higher affinity than v_j in observed response Θ_q. Similarly, the probability of recommending node v_i prior to node v_j is obtained by using the sigmoid function on the difference in corresponding values in stationary probability distribution Π_q. We then define the cross entropy cost for pair (v_i, v_j) as follows,

$$\begin{aligned}C_{ij} &= -\overline{p_{ij}}\,\log p_{ij} \;-\; (1-\overline{p_{ij}})\,\log(1-p_{ij})\\ &= -\lambda\overline{p_{ij}}o_{ij} \;+\; \log(1+e^{\lambda o_{ij}}) \qquad \{\text{substituting for } p_{ij}\}\end{aligned} \tag{11}$$

The loss corresponding to a query q is then defined as the sum of C_{ij} over all possible pairs (v_i, v_j), i.e.

$$L_q = \sum_{(v_i,v_j)} C_{ij} \quad \text{s.t.} \quad \tau(v_i) = \tau(v_j) = \text{ target type}$$

We now compute the gradient of C_{ij} with respect to the parameters to be learned. Gradient of C_{ij}, w.r.t. α_r corresponding to each relation in $\mathcal{R}$ is computed as

$$\frac{\partial\ C_{ij}}{\partial \alpha_r} = \left(-\lambda\overline{p_{ij}} + \frac{\lambda e^{\lambda o_{ij}}}{1+e^{\lambda o_{ij}}}\right)\left(\frac{\partial\ \Pi_q(v_i)}{\partial \alpha_r} - \frac{\partial\ \Pi_q(v_j)}{\partial \alpha_r}\right). \tag{12}$$

Gradient w.r.t. β_j for $j = 1,2,\ldots,k$ is also computed in a similar fashion, which we omit writing here.

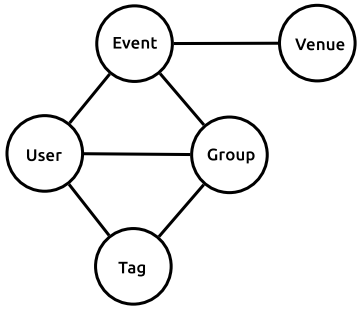

Figure 2: Schema diagram of Meetup dataset used in experiments.

5.2 Partial Derivative of Stationary Distribution

We further take a closer look at computing $\frac{\partial\ \Pi_q(v)}{\partial \alpha_r}$ $\left(\text{and } \frac{\partial\ \Pi_q(v)}{\partial \beta_j}\right)$. Although we provide a general framework for recommendation in any heterogeneous information network; for the ease of understanding, we describe computation of partial derivative of stationary probability distribution Π using Meetup schema shown in Fig. 2. We consider the task of recommend groups to the user. The stationary probability distribution corresponding to the Meetup schema

is,

$$\begin{aligned}
\Pi^*_U &= \alpha_{TU}\, P^\intercal_{TU}\, \Pi^*_T + \alpha_{GU}\, P^\intercal_{GU}\, \Pi^*_G + \alpha_{EU}\, P^\intercal_{EU}\, \Pi^*_E + \beta_U\, q_U \\
\Pi^*_T &= \alpha_{GT}\, P^\intercal_{GT}\, \Pi^*_G + \alpha_{UT}\, P^\intercal_{UT}\, \Pi^*_U + \beta_T\, q_T \\
\Pi^*_G &= \alpha_{EG}\, P^\intercal_{EG}\, \Pi^*_E + \alpha_{UG}\, P^\intercal_{UG}\, \Pi^*_U + \alpha_{TG}\, P^\intercal_{TG}\, \Pi^*_T + \beta_G\, q_G \\
\Pi^*_E &= \alpha_{UE}\, P^\intercal_{UE}\, \Pi^*_U + \alpha_{GE}\, P^\intercal_{GE}\, \Pi^*_G + \alpha_{VE}\, P^\intercal_{VE}\, \Pi^*_V + \beta_E\, q_E \\
\Pi^*_V &= \alpha_{EV}\, P^\intercal_{EV}\, \Pi^*_E + \beta_V\, q_V
\end{aligned}$$

where the subscripts U, T, G, E, and V denote *user*, *tag*, *group*, *event*, and *venue* object type respectively. Likewise, the subscript TU denotes the relation *users*-associated with-*tag* and so on. By looking at Eqn. 12, it appears that only gradient of stationary distribution of object to be recommended (*group*) is of interest to us. However, from above it is easy to verify that there exists a cyclic dependency amongst all the stationary distributions and hence their gradients. Due to this, it is required to find gradients of all stationary distributions. We illustrate gradient w.r.t. one parameter α_{TU} below

$$\begin{aligned}
\frac{\partial\, \Pi^*_U}{\partial\, \alpha_{TU}} &= \alpha_{GU} \frac{\partial\, P^\intercal_{GU}\Pi^*_G}{\partial\, \alpha_{TU}} + \alpha_{EU} \frac{\partial\, P^\intercal_{EU}\Pi^*_E}{\partial\, \alpha_{TU}} + \alpha_{TU} \frac{\partial\, P^\intercal_{TU}\Pi^*_T}{\partial\, \alpha_{TU}} + P^\intercal_{TU}\Pi^*_T \\
\frac{\partial\, \Pi^*_T}{\partial\, \alpha_{TU}} &= \alpha_{GT} \frac{\partial\, P^\intercal_{GT}\Pi^*_G}{\partial\, \alpha_{TU}} + \alpha_{UT} \frac{\partial\, P^\intercal_{UT}\, \Pi^*_U}{\partial\, \alpha_{TU}} \\
\frac{\partial\, \Pi^*_G}{\partial\, \alpha_{TU}} &= \alpha_{EG} \frac{\partial\, P^\intercal_{EG}\Pi^*_E}{\partial\, \alpha_{TU}} + \alpha_{UG} \frac{\partial\, P^\intercal_{UG}\Pi^*_U}{\partial\, \alpha_{TU}} + \alpha_{TG} \frac{\partial\, P^\intercal_{TG}\Pi^*_T}{\partial\, \alpha_{TU}} \\
\frac{\partial\, \Pi^*_E}{\partial\, \alpha_{TU}} &= \alpha_{UE} \frac{\partial\, P^\intercal_{UE}\Pi^*_U}{\partial\, \alpha_{TU}} + \alpha_{GE} \frac{\partial\, P^\intercal_{GE}\Pi^*_G}{\partial\, \alpha_{TU}} + \alpha_{VE} \frac{\partial\, P^\intercal_{VE}\Pi^*_V}{\partial\, \alpha_{TU}} \\
\frac{\partial\, \Pi^*_V}{\partial\, \alpha_{TU}} &= \alpha_{EV} \frac{\partial\, P^\intercal_{EV}\Pi^*_E}{\partial\, \alpha_{TU}}\ ,
\end{aligned} \tag{13}$$

Partial derivatives w.r.t. the other parameters can be obtained in a similar manner.

Unlike the case of conventional random walk, in VRRW the transition matrix keeps changing during the random walk, and is dependent upon the stationary probability distribution of the ending node. Hence, the computation of partial derivatives like $\frac{\partial\, P^\intercal_{TU}\Pi^*_T}{\partial\, \alpha_{TU}}$ is not straightforward and is computationally more involved. Using the definition of sublinear reinforced random walk, the transition matrix is modified based on the number of times the ending object has been visited. This has been specified in Eqn. 8. However, at convergence (as the length of walk tends to infinity), the same can be approximated using the stationary distribution. Thus,

$$E\left[(Z^*(v))^\rho\right] \approx \left(\Pi^*_b(v)\right)^\rho \quad \text{for all} \quad v \ \text{ s.t. } \tau(v) = b.$$

Without loss of generality, we consider a relation having starting object of type x and ending object of type y, with stationary distributions Π_x and Π_y respectively. We denote the initial transition matrix for this relation using P^0 $\left(= \left[P^0_{ij}\right]_{n\times m}\right)$. Then, the reinforced transition matrix P at convergence is given by

$$P = S^{-1}\, P^0\, Y^\rho$$

where,

$$S = diag\left(\sum_i P^0_{1i}\left(\Pi_y(i)\right)^\rho, \sum_i P^0_{2i}\left(\Pi_y(i)\right)^\rho, \ldots, \sum_i P^0_{mi}\left(\Pi_y(i)\right)^\rho\right)$$

and $Y = diag\Big(\Pi_y(1), \Pi_y(2), \ldots, \Pi_y(n)\Big)$. We describe the partial derivative of $P^\intercal \Pi_x$ w.r.t. the parameter α.

$$\begin{aligned}
\frac{\partial\, P^\intercal \Pi_x}{\partial\, \alpha} &= P^\intercal\, \frac{\partial\, \Pi_x}{\partial\, \alpha} + \rho\, Y^{\rho-1}\, P^\intercal\, S^{-1}\Pi_x \odot \frac{\partial\, \Pi_y}{\partial\, \alpha} \\
&\quad - \rho\, \Pi_y^{\rho-1} \odot \Pi_x S^{-2}\, P\, Y^{\rho-1} \frac{\partial\, \Pi_y}{\partial\, \alpha}
\end{aligned}$$

where $\odot$ is the element wise product or Hadamard product. The above is used in partial derivatives specified in Eqn. 13 for Meetup schema. It is easy to follow that upon substitution, the equations in 13 form a linear system of equations. This can be efficiently solved using linear solvers. Also, there always exists a solution to this system of equations. However, due to lack of space we omit the proof of this.

5.3 Learning Algorithm

We now describe the algorithm for learning the required parameters. For a given HIN, the algorithm starts by computing the initial transition probability matrix corresponding to each relation type. Also, the parameters are initialized uniformly in the beginning. We use the projected stochastic gradient descent method for learning the optimal values of the parameters. In each epoch we pick a random subset of k queries (mini-batch of size k), and compute the stationary probability distribution corresponding to it. Next, we use these stationary probability distributions along with values of the parameters known from the previous iteration to find the gradient of stationary distribution with respect to each parameter. The gradient corresponding to k queries in a epoch are accumulated, and parameters are updated using gradient descent with step size η/c. Here, η is the specified learning rate parameter, and c is the current epoch count. To ensure the constraints in Eqn. 10, we project the obtained parameter values to $\mathbb{R}^+ \bigcup\{0\}$. Also, we normalize the new learned parameters, such that the parameters for incoming relations to an object along with the corresponding query parameter sum upto 1.

Algorithm 1 gives the pseudocode for the above described approach.

6 EXPERIMENTS

We empirically evaluate the effectiveness of the proposed approach against the state-of-the-art techniques and other baselines. On Meetup schema shown in Fig. 2, we consider the following recommendation problems:

- Recommending *Groups* to *User*.
- Recommending *Tags* to *Group*.

We specifically choose these tasks as intuitively it sounds appealing to have diversity in the first task and no diversity in the second task.

6.1 Dataset

For our experiments, we crawled the data from the Meetup website[1]. Meetup is an online social networking website, which helps users with similar interests to organize and conduct an online/offline event. Users can join the groups based on their interests and can conduct or participate in events organized by the group.

[1] www.meetup.com

Algorithm 1 Learn Parameters using Stochastic Gradient Descent.

Input:
$\mathcal{G}(\mathcal{V},\mathcal{E})$ ▷ Given HIN snapshot
Q ▷ Set of queries
d ▷ Recommendation object type
Θ ▷ Set of observed responses corresponding to queries
k ▷ Sample size for SGD
Output:
$\Lambda = [\boldsymbol{\alpha}, \boldsymbol{\beta}]$ ▷ Learned parameters

$A \leftarrow \{A_r : A_r$ is adj. matrix for each relation in $\mathcal{G}(\mathcal{V},\mathcal{E})\}$
$P \leftarrow \{P_r : P_r$ is prob. transition matrix for each $A_r\}$
$\Lambda^0 = [\alpha^0, \beta^0]^\top \leftarrow$ Init. all relations and query params.
$i \leftarrow 1$ ▷ Maintains current SGD iteration count
while not converged **do**
  $\widehat{Q} \leftarrow$ Random subset of Q of size k
  $\widehat{\Theta} \leftarrow$ subset of Θ corresponding to $\widehat{Q}$
  grad $\leftarrow \mathbf{0}$
  for each $q \in \widehat{Q}$ **do**
    $\Pi^*_q \leftarrow$ compute stationary distribution using Eqn. 10
    for each parameter α_r do
      Compute $\frac{\partial \Pi^*_q}{\partial \alpha_r}$ as in Eqn. 13
      Use $\frac{\partial \Pi^*_q}{\partial \alpha_r}$ to compute $\frac{\partial L_q}{\partial \alpha_r} = \sum_{(v_i,v_j)} \frac{\partial C_{ij}}{\partial \alpha_r}$
      $grad(\alpha_r) \leftarrow grad(\alpha_r) + \frac{\partial L_q}{\partial \alpha_r}$
  $\Lambda^i \leftarrow \Lambda^{i-1} - \frac{\eta}{i}\frac{\mathbf{grad}}{\|\mathbf{grad}\|}$
  $\Lambda^i \leftarrow$ Project each element in Λ^i to $\mathbb{R}^+ \bigcup \{0\}$.
  Normalize Λ^i to ensure constraints in Eqn. 10.
  $i \leftarrow i + 1$
return $\Lambda^t = [\alpha^t, \beta^t]$

City	Users	Groups	Events	Tags	Venues
Bangalore	179,415	2,304	36,525	14,939	4,213
Hyderabad	117,309	997	32,527	10,520	2,453

Table 1: Statistics of Meetup dataset for different cities.

For two cities namely Bangalore, and Hyderabad, we collected the raw data using the Meetup API. Further, we considered the time range from the day Meetup was introduced up to July 31, 2017. For each city we collected the list of Meetup groups they have. Then, for each group we collected the tags and users associated with them and information about events that the group has conducted. Next, for the events we collect the venues where they have been conducted and list of users attending the event. Using attendance of events we only crawled information about the users, who have attended at least one event. We provide the statistics of the raw dataset in Table 1.

6.2 Algorithms

In addition to the proposed technique (`Div-HeteRec`), we consider two state-of-the-art techniques and two more related baseline algorithms to show the effect of diversity. We describe these techniques below:

(1) **Random Walk with Restart (`RWR`)**: In this conventional random walk with restart model there is no distinction between various node types. All the relation types and node types are treated uniformly and a homogeneous network is created. Corresponding to the given query, it then computes a single stationary distribution which is later exploited to make recommendations.

(2) **Collaborative Filtering using NMF (`NMF`)**: Here Non-Negative Matrix Factorization is used to implement the conventional collaborative filtering model. For *Groups* to *User* and *Tags* to *Group* recommendation it takes the Group-User and Tag-Group matrices respectively and applies NMF on them. Empirically the number of dimensions (k) is fixed to be 20. After factorization, the matrix is reconstructed and the magnitude of entries which were missing earlier is used to make recommendations.

(3) **Uniform Heterogeneous Recommendation (`Uni-HeteRec`)**: This is the completely stripped off version of our proposed approach. The baseline helps in highlighting the effectiveness of introducing the diversity and learning the influence of relations. The parameters α's and β's are initialized such that for an object all its dependencies get equal importance.

(4) **Learned Heterogeneous Recommendation (`Learn-HeteRec`)**: Unlike in our proposed approach, here the transition matrix is static. However, we learn the optimal parameter values using the framework described in Sec. 5.1. It is to be noted that this approach is different from the HeteRS mentioned in [16] as the underlying loss function is different.

For sublinear reinforcement in `Div-HeteRec` we use $\rho = 1/2$. Also, with $\rho = 0$, `Div-HeteRec` is same as `Learn-HeteRec`.

6.3 Setup

Taking a glance at the data, we found the data to be highly sparse, indicating that the use of Meetup has not matured still in the cities considered. The number of users are above $100k$ for the considered cities, but surprisingly very few of these users are actively using the portal. Because of this reason, we found it helpful to filter off the not so useful objects by preprocessing the data. We remove all the users wo have attended less than 5 events over this entire duration. Similarly for groups we remove those which had conducted less than 5 events. Also, we remove events which had less than 5 people attending them.
After preprocessing the data we partition it into train, validation, and test sets. We describe the strategy adapted for partitioning next. We partition the given data into 60%, 20%, and 20% as train, validation and test dataset respectively. While partitioning we only consider the relation concerning the recommendation problem. For example, for group to user recommendation we perform the split only on user-group adjacency matrix (and correspondingly group-user). However, specifically for group to user recommendation task, after moving the groups for a user to test and validation set, we also

Algorithm	Precision@k					Recall@k					nDCG@k				
	$k=1$	$k=2$	$k=3$	$k=5$	$k=10$	$k=1$	$k=2$	$k=3$	$k=5$	$k=10$	$k=1$	$k=2$	$k=3$	$k=5$	$k=10$
RWR	.1195	.1064	.0958	.0819	.0635	.0656	.1172	.1561	.2186	.3352	.1195	.1323	.1461	.1698	.2104
NMF	.0054	.0040	.0043	.0076	.0065	.0018	.0024	.0050	.0170	.0279	.0320	.0382	.0439	.0528	.0679
Uni-HeteRec	.1150	.1120	.1045	.0854	.0651	.0634	.1238	.1745	.2376	.3516	.1150	.1366	.1562	.1797	.2196
Learn-HeteRec	.1544	.1295	.1156	.0945	.0698	.0895	.1469	.1883	.2511	.3656	.1544	.1656	.1806	.2034	.2426
Div-HeteRec	**.1600**	**.1317**	**.1177**	**.0956**	**.0713**	**.0906**	**.1471**	**.1941**	**.2601**	**.3767**	**.1600**	**.1676**	**.1843**	**.2085**	**.2494**

Table 2: *Group* to *User* Recommendation for Bangalore Meetup dataset.

Algorithm	Precision@k					Recall@k					nDCG@k				
	$k=1$	$k=2$	$k=3$	$k=5$	$k=10$	$k=1$	$k=2$	$k=3$	$k=5$	$k=10$	$k=1$	$k=2$	$k=3$	$k=5$	$k=10$
RWR	.1810	.1392	.1199	.0973	.0724	.1188	.1838	.2304	.2987	.4121	.1810	.1931	.2101	.2375	.2792
NMF	.0018	.0009	.0015	.0046	.0046	.0010	.0010	.0029	.0148	.0244	.0443	.0510	.0615	.0749	.0903
Uni-HeteRec	.1582	.1240	.1027	.0835	.0613	.1057	.1630	.2014	.2621	.3696	.1582	.1710	.1838	.2080	.2480
Learn-HeteRec	.1791	.1484	.1265	.1021	.0729	.1197	.1943	.2450	.3100	.4141	.1791	.2012	.2200	.2510	.2942
Div-HeteRec	**.2031**	**.1588**	**.1301**	**.1024**	**.0737**	**.1345**	**.2039**	**.2465**	**.3134**	**.4270**	**.2031**	**.2166**	**.2302**	**.2560**	**.2969**

Table 3: *Group* to *User* Recommendation for Hyderabad Meetup dataset.

Algorithm	Precision@k					Recall@k					nDCG@k				
	$k=1$	$k=2$	$k=3$	$k=5$	$k=10$	$k=1$	$k=2$	$k=3$	$k=5$	$k=10$	$k=1$	$k=2$	$k=3$	$k=5$	$k=10$
RWR	.0046	.0046	.0046	.0037	.0046	.0023	.0046	.0069	.0084	.0276	.0046	.0046	.0060	.0069	.0136
NMF	.0046	.0023	.0015	.0065	.0119	.0023	.0023	.0023	.0169	.0729	.0184	.0166	.0159	.0196	.0264
Uni-HeteRec	.0092	.0092	.0168	.0147	.0143	.0069	.0099	.0322	.0445	.0852	.0092	.0110	.0223	.0283	.0434
Learn-HeteRec	**.1244**	**.1037**	**.0875**	**.0654**	**.0484**	**.0883**	**.1329**	**.1651**	**.1997**	**.2887**	**.1244**	**.1349**	**.1481**	**.1648**	**.1976**
Div-HeteRec	.0552	.0483	.0399	.0350	.0253	.0399	.0637	.0791	.1059	.1444	.0553	.0629	.0693	.0829	.0981

Table 4: *Tag* to *Group* Recommendation for Bangalore Meetup dataset.

Algorithm	Precision@k					Recall@k					nDCG@k				
	$k=1$	$k=2$	$k=3$	$k=5$	$k=10$	$k=1$	$k=2$	$k=3$	$k=5$	$k=10$	$k=1$	$k=2$	$k=3$	$k=5$	$k=10$
RWR	.0000	.0000	.0034	.0031	.0015	.0000	.0000	.0077	.0098	.0098	.0000	.0000	.0041	.0052	.0052
NMF	.0000	.0000	.0000	.0000	.0018	.0000	.0000	.0000	.0000	.0077	.0128	.0095	.0122	.0154	.0207
Uni-HeteRec	.0383	.0255	.0204	.0174	.0120	.0239	.0289	.0341	.0443	.0643	.0383	.0344	.0343	.0397	.0469
Learn-HeteRec	**.2071**	**.1483**	**.1159**	**.0879**	**.0590**	**.1189**	**.1670**	**.1994**	**.2570**	**.3341**	**.2071**	**.1858**	**.1922**	**.2187**	**.2479**
Div-HeteRec	.1176	.0805	.0605	.0485	.0322	.0635	.0886	.9846	.1343	.1777	.1176	.1013	.0994	.1164	.1332

Table 5: *Tag* to *Group* Recommendation for Hyderabad Meetup dataset.

remove the events associated with a group which were attended by the user to maintain consistency. Even after preprocessing task we will be left with users associated with less than 5 groups. For performing the split we only consider the users who are members of at-least 5 groups, while the other users are retained in train set. Similar is the case with tag to group recommendation where we only consider groups with at-least 5 tags for performing the split. For evaluation, we use Precision, Recall, and Normalized Discounted Cumulative Gain as the measures for both the tasks. For a given query q with set of observed responses O_q and set of recommendations R_q. Precision is defined as $|O_q \bigcap R_q|/|R_q|$, and Recall as $|O_q \bigcap R_q|/|O_q|$. The nDCG measure is computed as

$$nDCG = \frac{1}{IDCG} \sum_{i}^{|R_q|} \frac{rel_i}{\log_2(i+1)}$$

where, $IDCG$ is the ideal discounted cumulative gain which considers the set of relevant documents only and is defined as,

$$IDCG = \sum_{i}^{|R_q|} \frac{rel_i}{\log_2(i+1)}$$

It should be noted that unlike $nDCG$ where rel_i is the relevance of i^{th} recommendation in R_q, in $IDCG$ rel_i is the relevance of i^{th} recommendation in O_q. For precion@k, recall@k, and nDCG@k we consider R_q as the set of top k recommendations only.

6.4 Results and Discussion

For the Meetup dataset, we present the results for *Group* to *User* recommendation in Table 2 and Table 3, and results for *Tag* to *Group* recommendation in Table 4 and Table 5 for both the cities respectively.

We first analyze the results for *Group* to *User* recommendation. To start with, conventional collaborative filtering model `NMF` performs worst amongst all. This reasserts the fact that exploiting axillary information helps in recommendation task. Assigning equal importance to all relations seems to perform worse than the plain random walk with restart (`RWR`) model. However, when the weights are learned using the proposed framework the performance improves. Further, it is easy to observe from Table 2 and Table 3, that the diverse recommendations made using `Div-HeteRec` outperforms all other approaches. The improvement achieved by using

Div-HeteRec over Learn-HeteRec is more pronounce in the Hyderabad Meetup dataset. This is due to the fact that, Bangalore has more of information technology (IT) culture and majority of the groups are pertaining to IT only. Hence, there is a lack of diversity amongst groups in Bangalore. As a result of this diversity in recommended groups does not make much sense for this data.
Next we analyze the results for *Tag* to *Group* recommendation. Both RWR and Collaborative Filtering using NMF fail drastically in this task. Also, the Uni-HeteRec model which was performing satisfactorily in previous task fails. The same (uniform) parameter values which were doing reasonably well in previous task do not work in this setting. This emphasizes the fact that, for different tasks, different relations carry different priorities. However, unlike the previous case where Div-HeteRec was performing best here we find that Learn-HeteRec outperforms all other approaches. This is in-line with the intuition that the tags of groups are closely related to each other and suggesting a diverse set of tags should hurt the performance.
Summarizing our observations, we find that

- Making diverse recommendations in cases where they are intuitive also improves the performance of a recommender system.
- learning the importance of relations always helps; different recommendation tasks will require different weightages to be assigned to various relations.

7 CONCLUSION

In this paper, we introduced the need for diversity in recommendations in a heterogeneous information network (HIN). We proposed a sublinear vertex reinforced random walk based mechanism for integrating diversity. This emulates rich getting richer mechanism by increasing transitions to nodes which have been visited more frequently in the past. Further, in a HIN, different relation types have significantly different priorities for a given recommendation task. In a multivariate random walk framework we proposed a cross entropy cost based learning framework for systematically learning these priorities. Finally, we demonstrated the effectiveness of our approach using real-world Meetup dataset. So, naturally for recommendations requiring diversity the proposed Div-HeteRec is performing the best and for situations where diversity is not essential, the proposed Learn-HeteRec is ideally suited. It is interesting to explore, in the future, a hybrid model that aptly combines both diverse and non-diverse requirements. We make the code and datasets used in our experiments publicly available.

REFERENCES

[1] Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 335–336.

[2] Wai-Ki Ching, Ximin Huang, Michael K Ng, and Tak Kuen Siu. 2013. Markov Chains: Models, Algorithms and Applications. (2013).

[3] Marina Drosou and Evaggelia Pitoura. 2010. Search result diversification. *ACM SIGMOD Record* 39, 1 (2010), 41–47.

[4] Avinava Dubey, Soumen Chakrabarti, and Chiranjib Bhattacharyya. 2011. Diversity in ranking via resistive graph centers. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM.

[5] Rachid Guerraoui, Anne-Marie Kermarrec, Tao Lin, and Rhicheek Patra. 2017. Heterogeneous recommendations: what you might like to read after watching interstellar. *Proceedings of the VLDB Endowment* 10, 10 (2017), 1070–1081.

[6] Taher H Haveliwala. 2002. Topic-sensitive pagerank. In *Proceedings of the 11th international conference on World Wide Web*. ACM, 517–526.

[7] Daniel S Himmelstein and Sergio E Baranzini. 2015. Heterogeneous network edge prediction: a data integration approach to prioritize disease-associated genes. *PLoS computational biology* 11, 7 (2015), e1004259.

[8] Jon M Kleinberg. 1999. Hubs, authorities, and communities. *ACM computing surveys (CSUR)* 31, 4es (1999), 5.

[9] Rong-Hua Li and Jeffery Xu Yu. 2013. Scalable diversified ranking on large graphs. *IEEE Transactions on Knowledge and Data Engineering* 25, 9 (2013), 2133–2146.

[10] Qiaozhu Mei, Jian Guo, and Dragomir Radev. 2010. Divrank: the interplay of prestige and diversity in information networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. Acm, 1009–1018.

[11] Sharad Nandanwar and M Narasimha Murty. 2016. Structural neighborhood based classification of nodes in a network. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1085–1094.

[12] Sharad Nandanwar and M Narasimha Murty. 2018. Overlap-Robust Decision Boundary Learning for Within-Network Classification. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. AAAI.

[13] Zaiqing Nie, Yuanzhi Zhang, Ji-Rong Wen, and Wei-Ying Ma. 2005. Object-level ranking: bringing order to web objects. In *Proceedings of the 14th international conference on World Wide Web*. ACM, 567–574.

[14] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web.* Technical Report. Stanford InfoLab.

[15] Robin Pemantle. 1992. Vertex-reinforced random walk. *Probability Theory and Related Fields* 92, 1 (1992), 117–136.

[16] Tuan-Anh Nguyen Pham, Xutao Li, Gao Cong, and Zhenjie Zhang. 2015. A general graph-based model for recommendation in event-based social networks. In *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*. IEEE, 567–578.

[17] Filip Radlinski, Paul N Bennett, Ben Carterette, and Thorsten Joachims. 2009. Redundancy, diversity and interdependent document relevance. In *ACM SIGIR Forum*, Vol. 43. ACM, 46–52.

[18] Chuan Shi, Xiangnan Kong, Yue Huang, S Yu Philip, and Bin Wu. 2014. Hetesim: A general framework for relevance measure in heterogeneous networks. *IEEE Transactions on Knowledge and Data Engineering* 26, 10 (2014), 2479–2492.

[19] Yizhou Sun, Rick Barber, Manish Gupta, Charu C Aggarwal, and Jiawei Han. 2011. Co-author relationship prediction in heterogeneous bibliographic networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*. IEEE, 121–128.

[20] Yizhou Sun and Jiawei Han. 2012. Mining heterogeneous information networks: principles and methodologies. *Synthesis Lectures on Data Mining and Knowledge Discovery* 3, 2 (2012), 1–159.

[21] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment* 4, 11 (2011), 992–1003.

[22] Hanghang Tong, Jingrui He, Zhen Wen, Ravi Konuru, and Ching-Yung Lin. 2011. Diversified Ranking on Large Graphs: An Optimization Viewpoint. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '11)*. ACM, New York, NY, USA, 1028–1036. DOI : http://dx.doi.org/10.1145/2020408.2020573

[23] Stanislav Volkov. 2001. Vertex-reinforced random walk on arbitrary graphs. *Annals of probability* (2001), 66–91.

[24] Stanislav Volkov. 2006. Phase transition in vertex-reinforced random walks on with non-linear reinforcement. *Journal of Theoretical Probability* 19, 3 (2006), 691–700.

[25] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 283–292.

[26] Cheng Xiang Zhai, William W Cohen, and John Lafferty. 2003. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. ACM, 10–17.

[27] Mi Zhang and Neil Hurley. 2008. Avoiding monotony: improving the diversity of recommendation lists. In *Proceedings of the 2008 ACM conference on Recommender systems*. ACM, 123–130.

[28] Shenghuo Zhu, Kai Yu, Yun Chi, and Yihong Gong. 2007. Combining content and link for classification using matrix factorization. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 487–494.

[29] Xiaojin Zhu, Andrew B Goldberg, Jurgen Van Gael, and David Andrzejewski. 2007. Improving Diversity in Ranking using Absorbing Random Walks.. In *HLT-NAACL*. 97–104.