Directed Edge Recommender System

Ios Kotsogiannis
Duke University
iosk@cs.duke.edu

Elena Zheleva
University of Maryland
elena@cs.umd.edu

Ashwin Machanavajjhala
Duke University
ashwin@cs.duke.edu

ABSTRACT

Recommender systems have become ubiquitous in online applications where companies personalize the user experience based on explicit or inferred user preferences. Most modern recommender systems concentrate on finding relevant items for each individual user. In this paper, we describe the problem of directed edge recommendations where the system recommends the best item that a user can gift, share or recommend *to another user* that he/she is connected to. We propose algorithms that utilize the preferences of both the sender and the recipient by integrating individual user preference models (e.g., based on items each user purchased for themselves) with models of sharing preferences (e.g., gift purchases for others) into the recommendation process. We compare our work to group recommender systems and social network edge labeling, showing that incorporating the task context leads to more accurate recommendations.

Keywords

Recommender Systems, Social Networks, Directed Edge Recommenders

1. INTRODUCTION

Sharing is an important part of building and maintaining close relationships. When sharing, a person selects an item and gives it to someone else to use. Some examples include giving gifts, recommending products and sharing stories to read. The reasons for sharing range from pleasing and delighting someone to establishing one's self as an expert on a given topic, and studies have shown that the decision process behind choosing items to share with others is different from the one behind choosing items for personal use [6, 8, 16]. Many e-commerce companies are tapping into the commercial potential of sharing by building interfaces that allow users to share seamlessly with each other. Moreover, companies can help users find shareable content and recommend what items users should share. Since sharing is inherently an asymmetric action, recommending what to share requires a solution that reflects the preferences and needs of the sender while incorporating information about the recipient's taste.

Here, we present the problem of *directed edge recommendations*, in which a recommender system helps users find what to share and give to others. At a first glance, the directed edge recommendation problem is very similar to the generic recommendation problem where the data is represented by a set of 3-tuples (u, i, ρ), an user-item pair associated with a rating, and the goal is to predict ρ' for a new pair (u', i'). However, it has two subtle but important differences: 1) instead of recommending an item using information about one user, we have to recommend an item that is relevant for a *pair* of users, 2) the underlying distribution for directed edge activity (e.g., gift purchases) can be different from the distribution for other user activity (e.g., purchases for one's self). More specifically, in the context of *directed edge recommendations*, the relevant variables are four: *the sender, the recipient, the item i, and the rating for that item.* Hence, instead of 3-tuples we have 4-tuples of the form (u, r, i, ρ).

Directed edge recommender systems are also similar to group recommender systems because both take into consideration multiple individual preference models. However, while the goal of group recommender systems is to recommend items that have a high degree of agreement with the individual models, the goal of a directed edge recommender system is to recommend items that a specific user is likely to share with others, thus taking into consideration that user's sharing preferences and goals. Moreover, directed edge recommendations do not have to agree with both the sender and the recipient individual preferences. If we take an online bookstore as an example, traditional recommender systems would recommend books that a user is likely to buy and read, directed edge recommender systems would suggest books that a user is likely to gift to a given friend, and group recommender systems would suggest books that a book club is likely to be interested in discussing. If we take an online movie rental company as an example, a traditional recommender system would highlight movies that a person is likely to watch, a directed edge recommender system would recommend movies that a user is likely to share with a friend, and a group recommender system would suggest movies that a family can watch together.

We propose models which infer the sharing preferences of a user with respect to a specific recipient and recommend items that match these inferred preferences. In contrast with current state of the art techniques, our proposed models make use of both edge level and node level properties

WSDM 2017, February 06-10, 2017, Cambridge, United Kingdom
© 2017 ACM. ISBN 978-1-4503-4675-7/17/02...$15.00
DOI: http://dx.doi.org/10.1145/3018661.3018729

to infer the best model for each directed edge individually. We also propose mixing techniques for combining traditional recommender systems for solving the directed edge recommendation problem.

Roadmap

In section 2 we define our framework and formally define the directed edge recommendation problem. In section 3 we discuss the related areas of collaborative filtering, group recommendations and edge labeling. In section 5 we describe our baselines, including ones that use standard collaborative filtering techniques to provide with recommendations on the edge level, and expand them by proposing mixing the preferences of the sender and the recipient into one model. In section 6 we propose our novel algorithms for directed edge recommendations which incorporate edge features. In section 7 we empirically evaluate our methods on two different datasets and discuss the results.

2. PROBLEM DESCRIPTION

Let V denote a set of individuals, and $\mathcal{I}$ a set of items (e.g., products, books, movies, etc.). We assume a directed social network $G = (V, E)$ where V corresponds to the individuals, and an edge $e = (u, v) \in E$ represents a directed relationship from u to v. We define *actions* as interactions between users and items. We define two types of actions – personal and shared actions. A *personal action* is a tuple $p = (v, i)$ where $v \in V$ and $i \in \mathcal{I}$. Examples of personal actions include a book that an individual bought, or a movie than an individual rated. Let $\mathcal{P}$ denote the set of all personal actions, and let $\mathcal{P}_v = \{p \in \mathcal{P} | \exists i, p = (v, i)\}$ denote the set of personal actions performed by individual v. A *shared action* is an *ordered* triple $a = (s, r, i)$ where $s \in V$ and $r \in V$ are the active (sender) and passive (recipient) participants of the action respectively, with $(s, r) \in E$ and i is the item shared with the action. Examples of shared actions includes gifts sent from individual s to individual r, movies or books that individuals recommended to one another, etc. Let $\mathcal{A} = \{a_1, \dots\}$ be the set of all shared actions and $\mathcal{A}_s$ the set of actions where user s is the active user.

Definition 1 (Directed Edge Relevance). *We define the directed edge relevance as a function* Rel: $E \times \mathcal{I} \to \mathbb{R}$ *that given a directed edge* $e = (u, v)$ *and an item* i*, returns a relevance score of item* i *w.r.t. a shared action from* u *to* v*.*

We are now ready to define the problem of directed edge recommendations.

Problem 1 (Top-k Edge Recommendation). *Given a social graph* $G = (V, E)$*, a set of personal actions* $\mathcal{P}$ *and a set of shared actions* $\mathcal{A}$*, and an edge relevance function* Rel*, we want to construct a list of items* $\mathbf{I}_{(s,r)}$ *for* $\forall (s, r) \in E$ *such that:*

a. $|\mathbf{I}_{(s,r)}| = k$*, i.e. recommend* k *items*

b. $\forall (s, r, i) \in \mathcal{A} : \quad \nexists i \in \mathbf{I}_{(s,r)}$*, i.e., do not recommend items that have been part of a previous shared action with* s *and* r*.*

c. $\mathbf{I}_{(s,r)}$ *is sorted in non-increasing order with respect to the relevance function* Rel*.*

3. RELATED WORK

We give a brief overview of related work on recommender systems, edge labeling, and psychology of sharing.

3.1 Recommender systems

Two approaches to recommender systems are content based filtering and collaborative filtering [2]. *Content-based filtering* infers a user interest model by looking at the items that this user has liked, purchased, or interacted with in the past and then recommends new items that match this user interest model [14]. *Collaborative filtering* recommends items to a user based on the items that similar users have liked, purchased and interacted with [7, 20]. Within collaborative filtering models, there are memory-based (neighborhood-based) [10] and model-based (matrix factorization) [12] methods. Hybrid recommender systems combine collaborative with content based filtering in a single framework [2]. Context aware recommender systems consider additional information, such as time and location, to provide relevant information based on the context [3]. Another context dimension that recommender systems use is the social network of users. Yang et al. [23] survey methods for recommending items to individual users by incorporating preferences of connected users. Unlike directed edge recommendations, all of these approaches solve the problem of recommending items to individuals for personal use.

Group recommender systems address the problem of providing recommendations for a group of users instead of recommendations for a single individual. Given an item set and a set of users, the goal is to recommend a subset of items specific to the user group such that (a) the recommended items are highly relevant to the members of the group, and (b) there is a low disagreement between group members [5, 15, 18]. In contrast to group recommenders, directed edge recommenders combine individual user models (that of a sender and a recipient) to recommend an item to a single individual (namely the sender). Additionally, our models treat prior activity on directed edges (e.g., gift purchases) differently from activity on individual nodes (e.g., purchases for one's self). We compare our algorithms to Amer-Yahia et al.'s group recommender algorithms [5] on the directed edge recommendation probem in the Experiments section.

3.2 Edge labeling and link prediction

The problem of finding directed edge recommendations is closely related to the problem of edge labeling in social networks. Edge labeling in graphs refers to the problem of inferring properties of existing edges. In the context of social networks, edge labeling has been used to find whether two people are friends or foes [4], whether a pair of individuals have an advisor-advisee relationship [22], and for assigning topic distributions for connected pairs of users [21]. The work by Tang et al. [21] is the closest to our work, hence we compare our models to their work in the experiments section. They propose a generative model for topical affinity propagation which infers the topic-level influence between any two connected individuals in a social network. They apply their model to the problem of expert finding and show that their model has better accuracy compared to PageRank. One of the main differences between their work and ours is that their model uses only node properties to infer edge labels and it does not incorporate properties of edges like prior edge activity. Their method also relies on having a well-connected network between the individuals since it was developed for topic propagation, whereas our methods can work even on sparsely connected networks. Specifically, our methods can recommend to pairs of users who are not connected to the rest of the network.

The directed edge recommendation problem is also complementary to the problem of link prediction. Given a social network, the goal of link prediction is to infer new edges in the network [23]. While the goal of our work is to make recommendations for existing edges, directed edge recommenders can work on both existing edges and new edges that are inferred by a link prediction algorithm.

3.3 Psychology of sharing

Our work is built on the premise that the items which people choose to give and share with others are not necessarily the same as the items that they choose for individual use. While sometimes users share self-relevant information, in order to receive social support and approval [9], other times they send pieces of content to others because of a shared interest [17]. What we share is also affected by the number of people with which we share [6], the device on which we share [11] and how we want to be perceived by others [13, 17]. Studies on the gifting behaviors of people have shown that gifts can act as statements of the giver's perception of the recipient [19]. Belk has demonstrated that in gift-giving, sometimes, the sender's taste dominates gift selection, other times it is dominated by the perceived recipient's taste [8, 16]. The goal of directed edge recommenders is to learn from historical data the sharing preferences of users in different situations and domains.

4. ALGORITHM APPROACH

All the algorithms we propose to solve the top-k edge recommendation problem in Sections 5 and 6 will represent each item i using a multidimensional feature vector $i.\mathbf{f} \in \mathbb{R}^d$. Our algorithms compute for an edge $e = (s, r)$ a *model* $e.\mathbf{m} \in \mathbb{R}^d$, that represents the kinds of items that the edge is likely to co-occur with in a shared action. Relevance is computed as:

$$\mathsf{Rel}(e, i) \ := \ e.\mathbf{m} \cdot i.\mathbf{f}^T \qquad (1)$$

Our algorithms differ in how the edge model $e.\mathbf{m}$ is computed. In Section 5, algorithms makes use of personal actions $\mathcal{P}$ alone to compute $e.\mathbf{m}$. We name such algorithms as *node-based* recommenders. In Section 6, algorithms make use of both the set of personal actions $\mathcal{P}$ as well as the shared actions $\mathcal{A}$ to compute $e.\mathbf{m}$ and we name them *edge-based* recommenders.

5. NODE RECOMMENDATIONS

First, we propose node-based algorithms that can be used for the edge recommendation problem. A common theme with these algorithms is that they use the personal actions $\mathcal{P}$ to compute the edge $e.\mathbf{m}$. For each node $v \in V$, we define a *feature profile* of individual $v.\mathbf{h}$ as the sum of the feature vectors of all the items i, such that (v, i) is a personal action.

$$\forall v \in V : \quad v.\mathbf{h} = \sum_{(v,i)\in\mathcal{P}} i.\mathbf{f}$$

5.1 Baselines

First we consider two baselines that use simple statistics of the observed data. Given a directed edge $e = (s, r)$, we infer $e.\mathbf{m}$ with the following two ways.

Sender Popular. Given a set of personal actions $\mathcal{P}$, for each directed edge $e = (s, r)$ the edge model is determined completely by the sender s's feature profile. That is:

$$e.\mathbf{m} = v.\mathbf{h}$$

This can be interpreted as follows: when choosing a gift, a sender picks gifts for a receiver (a shared action) from among the items that he/she is most likely to purchase (as a personal action). Note that in this scenario, no information about the receiver r is used to determine the edge recommendation. We call this algorithm SENDERPOP, as the recommendations made to an edge (s, r) correspond to s's most likely personal actions.

Recipient Popular. We also consider a recipient popular algorithm, similar to SENDERPOP. RECIPIENT POP considers the recipient's personal actions, and recommends items that a receiver is most likely to receive are chosen from the receiver r's personal actions. More specifically, for a directed edge $e = (s, r)$:

$$e.\mathbf{m} = r.\mathbf{h}$$

5.2 Node-Level Collaborative Filtering

We next consider using standard collaborative filtering (CF) techniques on the node level. Given a directed edge $e = (s, r)$ our CF based algorithms recommend items based on the sender node s's personal actions and the preferences of nodes similar to s or based on node r's items and the preferences of nodes similar to r. Again we focus on using the CF algorithms to create a ranking on the feature space for each edge $e \in E$. Memory-based CF algorithms create a user model $r.\mathbf{m} \in \mathbb{R}^d$, that represents the items that the user r is likely to personally interact with. They do so by grouping users with similar interests, then identify the neighborhood of each user and create the model for him. The steps that a neighborhood-based CF algorithm takes are the following: (a) compute the similarity $sim(r, w)$ between users r and w, (b) then for a target user r find the most similar users to him and aggregate their preferences to derive model $r.\mathbf{m}$ for the target user.

Applying collaborative filtering on the node level is equivalent of making a recommendation for each node's next personal action. In our gifts example this means predicting either s's or r's next purchase, not a gift-specific purchase.

First, we define the similarity of two users v and v' as follows: $sim(v, v') = v.\bar{\mathbf{h}} \cdot v'.\bar{\mathbf{h}}$, where $v.\bar{\mathbf{h}} = \frac{v.\mathbf{h}}{\|v.\mathbf{h}\|_1}$. The θ-neighborhood of a node v is defined as: $N_v = \{w \in V \mid sim(v, w) \geq \theta\}$. Then model for a user v is:

$$v.\mathbf{m} = \sum_{w \in N_v} sim(v, w) \cdot w.\mathbf{h}$$

The model for a node v as computed above (using CF) is denoted by CFN(v). Using this scoring mechanism (CF) we propose 3 approaches for the top-k edge recommendation problem: (a) sender based, (b) recipient based, and (c) composite.

Sender Node-Based CF. Given a directed edge $e = (s, r)$ the algorithm computes $\text{CFN}(s)$ and sets $e.\mathbf{m} = \text{CFN}(s)$. We denote this model as CFN-S, and is indicative of the sender's next personal activity in the system.

Recipient Node-Based CF. Given a directed edge $e = (s, r)$ the algorithm computes $\text{CFN}(r)$ and sets $e.\mathbf{m} = \text{CFN}(r)$. Similarly, we denote this model as CFN-R, and is indicative for the recipient's next personal activity in the system.

The drawback with the traditional collaborative filtering techniques is that they do not make use of the additional edge labels and their different semantics in order to provide a directed edge model.

5.3 Composition at the Node Level

We consider the composition of CFN-S and CFN-R and propose three different composition methods of them: the additive, the altruistic, and the multiplicative method.

Additive Composition. The intuition behind this model is that we want to give high ranks to features that are highly ranked either by the CFN-S or CFN-R models. Given a directed edge $e = (s, r)$:

$$e.\mathbf{m} = \alpha \cdot \text{CFN}(s) + (1 - \alpha) \cdot \text{CFN}(r), \quad \alpha \in (0, 1)$$

We denote this additive model with CFN-ADD. Also, note that when we choose $\alpha = 0.5$ the additive node model is equivalent to the average relevance group recommender algorithm proposed in [5] named AR. It simply ranks features based on the nodes' combined personal activities.

Altruistic Composition. We extend CFN-ADD with a novel technique that allows us to learn the value of α. More specifically, we assume that each active user s of an action, is either characterized as altruistic or selfish. For an action (s, r, i) altruistic users s choose an item i that is more relevant to the recipient r of the action, while selfish users choose items that are more relevant to their own interests. For a user $s \in V$ let $z_s \in [0, 1]$ his altruism variable, and we set $\alpha = 1 - z_s$ for all active actions of user s. More specifically, the proposed model is:

$$e.\mathbf{m} = z_s \text{CFN}(r) + (1 - z_s)\text{CFN}(s)$$

We denote this model with CFN-ALT.

Learning Altruism. Our goal is to lean the values of the altruism variables $\{z_s\}_{s \in V}$, to do so first we assume the following generative process for the personal and shared actions of the users. $\forall v \in V : P_v \sim \mathit{mult}(\theta_v)$ i.e., the personal actions of all users follow a multinomial distribution parameterized by the *taste* θ_v of the user. Similarly for all shared actions we have: $\forall (s, r, i) \in A : i \sim z_s \mathit{mult}(\theta_r) + (1 - z_s)\mathit{mult}(\theta_s)$. For a single shared action $a_j = (s, r, i)$ we have $P(a_j | \theta_s, \theta_r, z_s) = z_s \theta_{r_i} + (1 - z_s)\theta_{s_i}$, and for all shared actions we have: $P(A|\theta_S, \theta_R, z_S) = \prod_{(s,r,i) \in A} \left(z_s \theta_{r_i} + (1 - z_s)\theta_{s_i} \right)$. Then the log-likelihood of θ_s, θ_r, z_s is $L(\theta_S, \theta_R, z_S; A) = \sum_{(s,r,i) \in A} \left(z_s \theta_{r_i} + (1 - z_s)\theta_{s_i} \right)$. We propose algorithm 1 to find the value of the altruism variables z_s. We also define the shared action histories of a user: $v.\bar{\mathbf{s}} = v.\mathbf{s}/\|v.\mathbf{s}\|_1$ and $v.\bar{\mathbf{r}} = v.\mathbf{r}/\|v.\mathbf{r}\|_1$ with $v.\mathbf{s} = \sum_{(v,r,i) \in A} i.\mathbf{f}$, and $v.\mathbf{r} = \sum_{(s,v,i) \in A} i.\mathbf{f}$ respectively.

Multiplicative Composition. The intuition behind the multiplicative composition is assigning high ranks to features that are highly ranked by both the CFN-S or CFN-R models. Given a directed edge $e = (s, r)$:

$$e.\mathbf{m} = \text{CFN}(s) \odot \text{CFN}(r)$$

Where $\odot$ denotes the pairwise multiplication of two vectors. We denote this multiplicative model with CFN-M. This model can also be thought as a group recommender, since it ranks features based on the nodes' combined personal activities.

Algorithm 1 ALTRUISMEST

```
1: $\forall v \in V$ : assign $\theta_v = v.\bar{\mathbf{h}}$
2: Compute $z_S^* = \text{argmax}_{z_S} L(\theta_S, \theta_R, z_S; A)$
3: $\forall v \in V$ : update $\theta_v = v.\bar{\mathbf{h}} + (1 - z_s)v.\bar{\mathbf{s}} + z_s v.\bar{\mathbf{r}}$
4: Repeat 2-3, until convergence.
```

6. DIRECTED EDGE RECOMMENDATIONS

Now we present collaborative filtering techniques that given a directed edge $e = (u, v)$ make use of both the personal actions $\mathcal{P}$ and shared actions $\mathcal{A}$ in order to infer feature scores for the edge e. We consider the shared actions as separate entities from personal actions, and utilize them for creating edge feature scores. Before we continue with the algorithms we define the edge feature profile $e.\mathbf{f}$ as follows:

$$\forall (s, r) \in E : \ (s, r).(h) = \sum_{(e,i) \in \mathcal{A}} i.\mathbf{f}$$

6.1 Simple Baseline

First, we define a simple baseline algorithm that creates the edge feature scores $e.\mathbf{m}$ based on the overall popularity of features appearing on items shared between users. Given the training set of observed actions $\mathcal{A}_{tr}$, and the graph $G = (V, E)$, we compute the vector of all edge activities $\mathbf{m_{all}}$.

$$\mathbf{m_{all}} = \sum_{e \in E} e.\mathbf{h}$$

Then for any directed edge $e \in E$ the proposed edge feature scores are $e.\mathbf{m} = \mathbf{m_{all}}$. We refer to this model as MOSTPOP as it represents what are the most popular features over all the observed activities.

6.2 Edge-Level Collaborative Filtering

We expand the on the collaborative filtering techniques of section 4.2 by directly inferring the feature scores of the edges. Given a directed edge $e = (u, v)$: we compute the edge feature scores based on either the features of items whose senders are users similar to u or on features of items whose recipients are users similar to v.

First we define two new quantities on nodes in V. For a node v, we define the sent-aggregation and the received-aggregation vectors, $v.\mathbf{s}$ and $v.\mathbf{r}$ respectively. More specifically, for a node $v \in V$:

$$v.\mathbf{s} = \sum_{(v,w) \in E} (v, w).\mathbf{h} \qquad v.\mathbf{r} = \sum_{(w,v) \in E} (w, v).\mathbf{h}$$

These vectors are representative of the features of items that a specific user v either sent to or received from other users. In the following we propose three edge-level CF approaches:

(a) a sender-based, (b) a recipient-based, and (c) their composition.

Sender Based. Given a directed edge $e = (u, v)$, this algorithm finds what are the features of items *given from* similar senders to u and accordingly computes $e.\mathbf{m}$. More specifically, for a directed edge $e = (u, v)$ we infer the edge feature scores $e.\mathbf{m}$ as follows:

$$e.\mathbf{m} = \sum_{w \in N_u} sim(u, w) \cdot w.\mathbf{s}$$

Note that for edges where the sender is fixed, the inferred edge feature scores $e.\mathbf{m}$ will be the same. We use the name CFs to refer to this algorithm.

Recipient Based. Similarly we propose the following recipient based edge recommender algorithm. Given a directed edge $e = (u, v)$ the edge scores are inferred as follows:

$$e.\mathbf{m} = \sum_{w \in N_v} sim(v, w) \cdot w.\mathbf{r}$$

Given a directed edge $e = (u, v)$, this algorithm finds what are the features of items *received from* similar recipients to v and accordingly computes $e.\mathbf{m}$. We use the name CFr to refer to this algorithm. Also note, that for CFr to work we need to have prior knowledge of the personal activities of the recipient (e.g., the target recipient is an existing member in our system), otherwise we cannot make a personalized recommendation.

6.3 Mixing at the edge level

Similar to the node level recommenders we also propose an additive and a multiplicative composition. The intuition and the ideas behind each composite edge-level model are similar to the ones we had for the composite algorithms for section 4.2, but here the semantics are different.

Additive Composition. First we present the additive model which we will refer to as CFsr-a. The intuition behind this algorithm is that given a directed edge $e = (u, v)$ we want to provide with feature scores that are high if: (a) they are either representative of items that *senders* similar to u like *to give*, or (b) they are representative of items that *recipients* similar to v like *to receive*. More specifically given a directed edge $e = (u, v)$ the edge scores are inferred as follows:

$$e.\mathbf{m} = \alpha \cdot CFS(u) + (1 - \alpha) \cdot CFR(v)$$
$$\text{Where: } \alpha \in [0, 1]$$

Multiplicative Composition. We also present a multiplicative algorithm: CFsr-m. The intuition behind it is to examine the edges separately and assign high ranks to features if they are representative of items that *both* senders similar to u like to send, and recipients similar to v like to receive. More specifically, given a directed edge $e = (u, v)$ the edge scores are inferred as follows:

$$e.\mathbf{m} = \sum_{w \in N_u} \sum_{z \in N_v} sim(u, w) \cdot sim(v, z) \cdot (w, z).\mathbf{h}$$

We believe this approach to be the most suitable for the problem of edge recommender. Not only does it consider edge properties as separate entities, but given a directed edge $e = (u, v)$ it highly ranks edge features where both endpoints are similar with the respective endpoints of e.

7. EXPERIMENTS

Next we describe the datasets, experimental setup and compare our proposed models to the algorithms described in section 5, as well as a set of *group recommender* models [5] and an *edge labelling* model [21]. Both the group recommender and the edge labelling models do not consider prior information from the edge labels, i.e. $e.\mathbf{h}$.

7.1 Dataset description

We used two datasets in our experimental evaluation: a films dataset consisting of a collection of movies, their directors and actors, and a gifts dataset which includes personal purchases and gift transactions between individuals. In the movies dataset the feature space is the genres of the movies, and in the gifts dataset the feature space are hierarchical categories of products. In the experiments we conducted we make recommendations of the feature space instead of direct recommendations of items. Hence, in the movies dataset given a director, actor pair we predict the genre of their next movie, and in the gifts dataset, we predict the category of the next gift a sender will give to a recipient.

Movies Dataset. We use the movies dataset from [21], and create a network $G = (V, E)$ where the nodes are actors and directors, we use the edges provided by the dataset. For the set of features $\mathcal{F}$ we use the top 11 genres of movies appearing in the dataset (comedy, adventure, action, drama, thriller, romance, horror, music, documentary, science-fiction, animation). We assign one genre on each movie based on the number of relevant keywords associated with that movie. Any movies that do not not have relevant keywords are discarded from our study. The value of the resulting genre for a movie is We denote the resulting set of movies with M. In order to populate the set of actions $\mathcal{A}$ we do the following: for any movie $m \in M$ in the dataset we find its director d and its top billing actor a using the API provided by [1], then if both the director and the actor exist in the network, we add the triple (d, a, m) in $\mathcal{A}$. In the resulting graph for each edge e, $e.\mathbf{f}$ is an aggregation of the genres of all the movies in which a director and an actor participated together. Since we do not use any external dataset of movies that the director and the actor participated outside our current dataset, we do not have a set of personal actions $\mathcal{P}$. Instead, in order to generate $v.\mathbf{h}$, we simply aggregate the labels of their incident edges. In total we have 4824 movies, 1598 directors, and 2315 actors in the dataset. As we already said, the feature scores on the nodes (actors, directors) is an aggregation of the feature scores on edges incident to each node. This means that the node-level and edge-level collaborative filtering approaches are equivalent when applied to this dataset. For this reason we applied only the node-level collaborative filtering algorithms.

Gifts Dataset. LivingSocial is an e-commerce company which connects businesses to customers through online deals. We collected a sample of personal and gift purchases that happened through the LivingSocial website interface. The sample was created by picking $100,000$ random gift purchases. For each sampled gift, we included all personal and gift purchases of the gifter and the giftee that happened prior to the gift. We create a network where the nodes are the users and a directed edge (u, v) signifies a relationship between users. For each gifting appearing in the dataset we add the triple (s, r, i) in $\mathcal{A}$, where s is the sender, r is the recipient, and i is the gift being given. To populate $\mathcal{P}$ we use the purchase history of the senders and recipients as taken from the dataset. To define the feature space $\mathcal{F}$ we used the high-level taxonomy of LivingSocial, which consists of 8 major categories of products sold within the online shop. Each item exchanged belongs to only one major category, hence the feature vector of the items is binary, and $\forall i \in \mathcal{I}: \ \|i.\mathbf{h}\|_1 = 1$. The feature scores of an edge $e = (u, v)$ correspond to an aggregation of the features of items that u gifted to v. The feature scores on a node v are an aggregation of the features of items that v purchased for himself. In this dataset the distinction between personal and shared actions is clear, despite both of them sharing the same feature space (products category), $\mathcal{P}$ corresponds to personal purchases, and $\mathcal{A}$ corresponds to gifts. In this dataset we apply both the node-level and edge-level algorithms with their composite counterparts.

7.2 Experimental setup

We perform experiments to evaluate the performance of the proposed models as follows. First we examine the performance improvement of the composite models over their single user versions. Then, we examine how well the new edge-level models perform against their node-level techniques. Lastly, we compare our algorithms to the group recommendation models in [5] as well as the message passing edge labelling algorithm in [21].

Algorithms. We implement all the node-level algorithms of section 5 on both the movies and the gifts dataset. We set the neighborhood threshold parameter at $\theta = 0.5$ and the composite parameter for the additive algorithm of section 5.3 $\alpha = 0.5$. We implement the edge-level algorithms of section 6 on the gifts dataset, again the parameters are set to $\theta = 0.5$ and $\alpha = 0.5$. From [5] we also implement the following algorithms: Least Misery (LM), RV, and RP for an extensive comparison with the group recommenders. For RP and RV we set the composite parameter at 0.5. These algorithms were tested on both datasets. Lastly we also use the TAP algorithm of [21], which is a message passing algorithm used to derive edge labels for the edges of a network based on known node labels. We evaluate its performance on the movies dataset. TAP is the only algorithm that makes additional use of the underlying social graph in order to infer the feature scores on the edges.

Evaluation. For each dataset we split the set of actions $\mathcal{A}$ to testing $\mathcal{A}_{ts}$ and training $\mathcal{A}_{tr}$. We infer all the models using $\mathcal{A}_{tr}$ and the goal is to correctly predict dominant feature of items in $\mathcal{A}_{ts}$. This means that our ground truth will always be one feature, for that reason every time we compare a list $\mathbf{I}_e$ of k recommendations only one of them will be relevant. We report the average recall (or hit ratio) as well as the average discounted cumulative count (DCG). Recall shows if the model in question can make the correct prediction, then DCG is a more nuanced metric that shows how high in the list the correct prediction is. In our case where the ground truth for each testing instance is exactly one feature: f_k the recall *recall* and *DCG* of a list of k recommendations $\mathbf{I}_e$ are defined as follows:

$$recall(\mathbf{I}_e) = \{^{1 \ \ if \ f_k \in \mathbf{I}_e}_{0 \ \ else} \qquad DCG(\mathbf{I}_e) = \frac{1}{log_2(j)}$$

Where $\mathbf{I}_{e_j} = f_k$, if $j = 1$ then $DCG(\mathbf{I}_e) = 1$, and if $f_k \notin \mathbf{I}_e$ then $DCG(\mathbf{I}_e) = 0$. To evaluate the performance we perform a 10-fold cross validation test on both datasets. For the movies dataset the node feature profiles $v.\mathbf{h}$ are estimated after each split. At each split we run all algorithms and measure their recall and DCG, after all runs we report the average recall, as well as the average DCG. Each recommender provides us with $k = \{1, 2, 3, 4\}$ different features as recommendations for each testing instance. With $rec_k(alg)$ we denote the average recall of algorithm alg for k recommendations, and similarly with $DCG_k(alg)$ we denote the average DCG of algorithm alg for k recommendations.

	CFSR-M	CFN-ADD	CFN-ALT	LM	RP
$k = 3$	56.9%	52.1%	54.8%	52.2%	43.4%
$k = 4$	82.1%	77.1%	77.4%	76.9%	63.3%

Table 1: DCG for Gifts Dataset

	CFN-M	CFN-ADD	LM	RP	TAP
$k = 3$	64.8%	65.6%	50.0%	33.6%	43.7%
$k = 4$	70.3%	71.2%	58.1%	35.9%	45.7%

Table 2: DCG for Movies Dataset

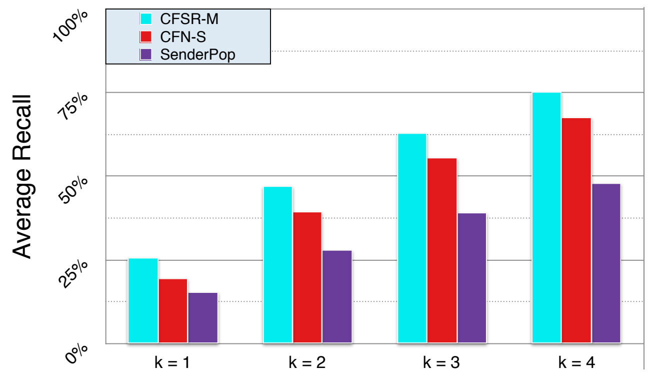
7.3 Model performance

First, we report the overall performance of the algorithms in our two datasets. In figure 1 we see the results for gifts where both the composition and the edge-level recommendation techniques have been applied. In figure 1a we show how our models fare against the baselines, note that instead of showing all the baselines, for each category of we pick the baseline that dominates the others. In the case of gifts we show SENDERPOP and CFN-S since they outperform the other baseline algorithms for all values of k. The results are not surprising, we see our proposed model CFSR-M fairing significantly better than the baselines. In figure 1b we make a comparison with the algorithms from [21]. Again we see CFSR-M achieving the best recall values for any k. We also see that two group recommenders AR and LM achieve good recall values, while RV that takes into account the variation of group disagreement fares worse than the baselines.

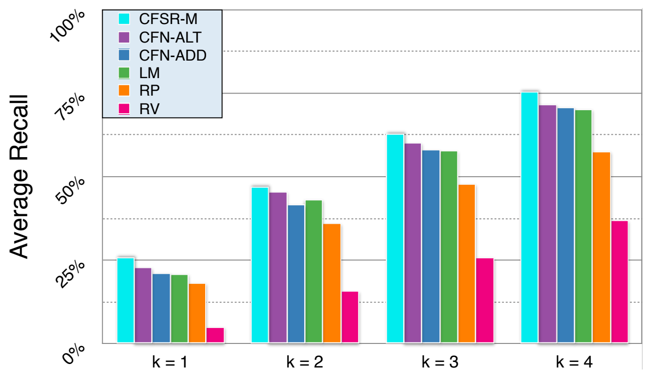
In figure 2 we see the results for the movies dataset were from our proposed models only the node-based recommenders have been applied. In figure 2a we show how our models fare against the baselines, again we do not report all the baselines. In this case the MOSTPOP and CFN-S (directors) dominate their respective categories for any k and we report them. In figure 2b we make a comparison with the algorithms from [21], as well as the TAP algorithm from [21].

In tables 1 and 2 we show a comparison of the algorithms in terms of DCG. Again we see the proposed models to offer better DCG than the competitive algorithms. This means that our algorithms provide with correct recommendations in their respective lists.

In figure 3 we summarize the improvement that the mixing models have over the simple ones. The values of each bar are the average recall differential of each mixed model against the *best performing* single node model for a single k value. Each group of bars corresponds to a different k value of recommendations. Each colored bar corresponds to a different composite model (additive, multiplicative and altruistic), more specifically the CFN-ADD group of bars corresponds to

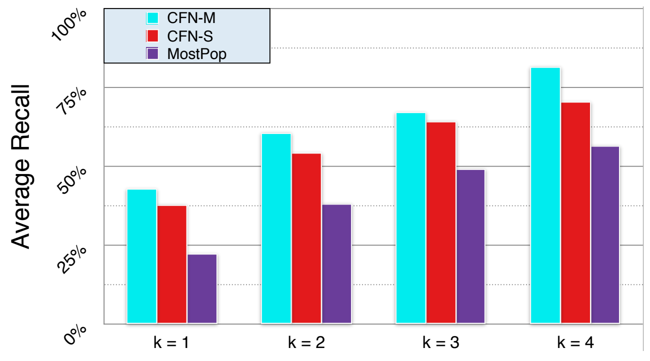


(a) Comparison with the best baseline approaches (SENDERPOP, CFN-S).

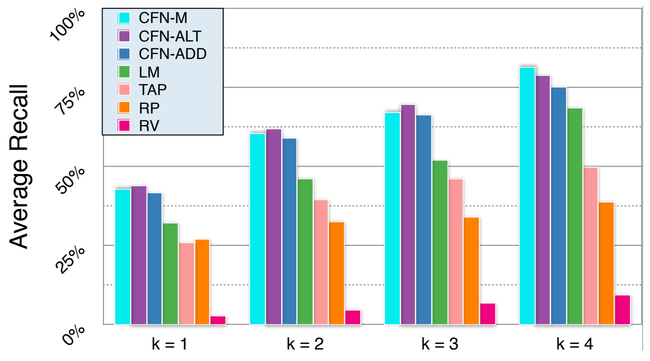


(b) Comparison of with group recommender algorithms (LM, RV, RP)

Figure 1: **Comparison of algorithms for gifts dataset**. The multiplicative directed edge recommender (CFSR-M) has overall higher recall than group recommenders. As the value k increases we see the gap closing, which is expected since the ground truth is 1 feature.



(a) Comparison with the best baseline approaches (MOSTPOP, CFN-Directors)



(b) Comparison with group recommender algorithms (LM, RV, RP) and the edge labelling algorithm TAP.

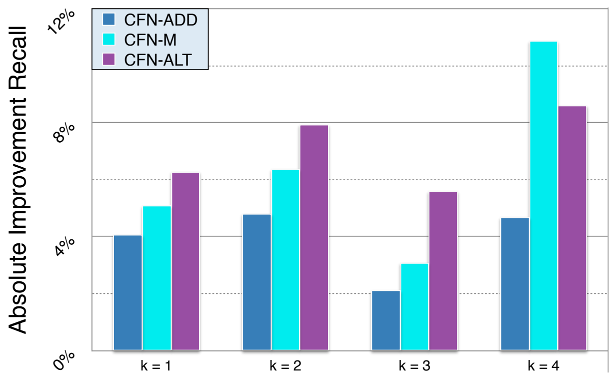
Figure 2: **Comparison of algorithms for movies dataset**. The multiplicative directed edge recommender (CFSR-M) has overall higher recall than group recommenders (LM, RP, RV) and the edge labelling algorithm (TAP). As the value k increases we see the gap closing, which is expected since the ground truth is 1 feature.

$rec_k(\text{CFN-ADD}) - max(rec_k(\text{CFN-S}), rec_k(\text{CFN-R}))$ for each value of k.

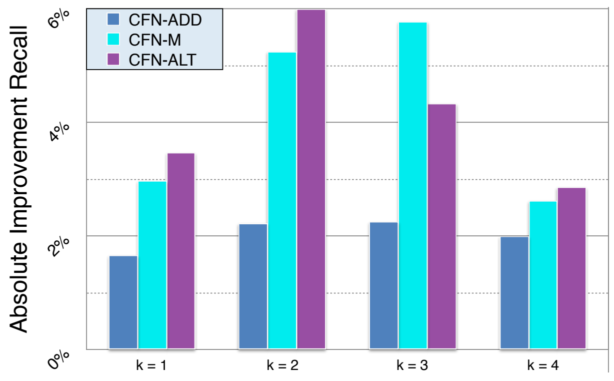
We see that the composite models in all configurations are superior to the single node ones. In the gifts dataset shown in figure 3a the difference is not as big as in the movies dataset shown in 3b. This is because for all values of k the recall values for the single node CF algorithms is significantly lower than the of the same algorithms in the movies dataset. For example in the gifts dataset for $k = 1$ the recall for CFN-S and CFN-R is 19.2% and 19.1% respectively, while the same algorithms have recall 37.6% and 34.6% respectively. The low performance of the single node algorithms leads to less improvement from their composition. Additionally in the gifts dataset we see that the multiplicative provides better improvement than the additive, while in the movies dataset the provide similar improvement. This is because the gap in terms of recall between CFN-S and CFN-R is smaller in the gifts dataset. A larger gap favors the additive approach, since for testing instances where the dominated algorithm fails the other algorithm can still provide with a good recommendation. Next, we measure the improvement that the edge based algorithms offer over their node based counterparts. We do so by comparing directly the collaborative filtering and composite algorithms of section 6 with the ones from section 5. Again we measure the absolute difference of average recall of the edge-level algorithms against equivalent node-level ones. More specifically we compare CFN-S with CFS, CFN-R with CFR, CFN-ADD with CFSR-A, and CFN-M with CFSR-M. In figure 4 we summarize our results, each group of columns corresponds to a different comparison and on the y-axis we plot their difference in achieved recall for various values of k. For example the first column shows $rec(\text{CFS}) - rec(\text{CFN-S})$ and so on.

Here the results are more clear. For any k value it's always better to choose an edge-based algorithm for recommending items on the edges. Even in the case of CFN-ADD which is used for group recommendation problems performs worse than CFSR-A. The most consistent improvement is shown by CFSR-M, which is no surprise. CFSR-M makes use of both nodes' information to find the most suitable edges and recommend based on them.

8. DISCUSSION AND FUTURE DIRECTIONS



(a) Gifts Dataset



(b) Movies Dataset

Figure 3: Absolute improvement in recall by using composite models instead of the single node ones. The y-axis corresponds to average recall differential. CFN-ADD corresponds to the improvement that the additive model has over the single node ones, and CFN-M corresponds to the improvement of the multiplicative over the single node ones. Each smaller column corresponds to a different k value.

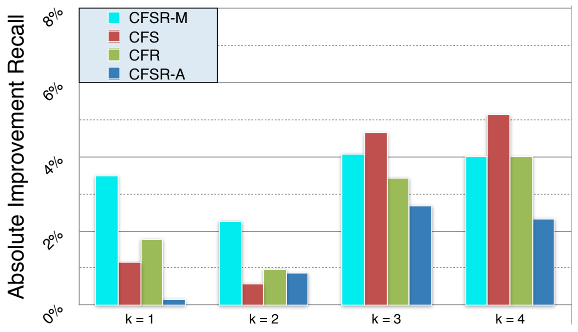


Figure 4: Absolute improvement in recall by using edge-based models instead of node-based. The y-axis corresponds to average recall differential. The first group of columns plots recall(CFS)- recall(CFN-S), each column in a group represents a different k value. This is on gifts dataset only.

While the idea of inferring a recipient's taste using historical data on the recipient's actions is intuitive, it comes with some privacy concerns. Directed edge recommendations may reveal sensitive information about the recipient that the sender would not have known otherwise, e.g., if the algorithm is using the private purchasing behavior of the recipient as its input. Moreover, using recipient's preference model for recommending relevant items to others can become a breach of trust between the recipient and the service where those actions are completed and recorded. There are at least two possible research directions to alleviate these concerns. One is developing privacy-preserving algorithms for directed edge recommendation and understanding the utility of such privacy-preserving algorithms against algorithms which do not use private recipient information. The second one is developing intuitive interfaces to facilitate information sharing for the purpose of directed edge recommendations, such as ones that allow you to declare gift preferences (i.e., what you like to receive as a gift) even if there is no specific occasion for creating a gift registry.

Another possible extension of this work is to directed *hyperedge* recommendations where either or both sender and recipient are represented by sets of users. One example is recommendations for wedding gifts where one or more individuals e buy a present for a pair of other individuals. Another example is lecture personalization where given a professor, a set of students and potential material to cover, the recommender would find the most relevant material. This type of recommender systems would have components of both group recommender systems and directed edge recommenders.

9. CONCLUSIONS

In this work, we define the problem of *directed edge recommendations*. We are motivated by numerous real life examples where people share different items with different people e.g., a teenager will recommend different movies to his friends than to his parents, or an adult will buy different gifts for her spouse than for her co-workers. People create connections with each other and these connections dictate their sharing habits. We propose new models where the recommendations are made on the level of people's connections, inferring the sharing preferences of people with respect to each person they are connected to. We also discuss the current state of the art and how it applies to our problem. We compare our new algorithms to collaborative filtering and group recommender techniques, as well as edge labeling methods which can also provide solutions for directed edge recommendation problem. We demonstrate how we can apply the existing techniques to our problem and propose two composition methods for the standard collaborative filtering technique. However, those solutions are not tailored to the specific problem we are solving, and our main contribution is in proposing new directed edge-level recommendation models with clear semantics. Our models consider the taste of both the sender and recipient, as well as the sharing preferences of the sender. Lastly, we experimentally evaluate our proposed algorithms on two real datasets against group recommending and edge labeling algorithms. Our empirical evaluation shows that the proposed directed edge-level models achieve higher recall compared to the collaborative filtering baselines, the group recommenders and edge labeling. We also show great results for the composition technique we propose over the standard collaborative filtering techniques which can be applied in other problems as well.

10. REFERENCES

[1] The open movie database. http://www.omdbapi.com/. Accessed: 2016-01-10.

[2] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *TKDE*, 17(6):734–749, 2005.

[3] G. Adomavicius and A. Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, pages 217–253. Springer, 2011.

[4] P. Agrawal, V. K. Garg, and R. Narayanam. Link label prediction in signed social networks. In *IJCAI*, 2013.

[5] S. Amer-Yahia, S. B. Roy, A. Chawlat, G. Das, and C. Yu. Group recommendation: Semantics and efficiency. *VLDB*, 2(1):754–765, Aug. 2009.

[6] A. Barasch and J. Berger. Broadcasting and narrowcasting: How audience size affects what people share. *Journal of Marketing Research*, 51(3):286–299, 2014.

[7] N. Barbieri, G. Manco, and E. Ritacco. Probabilistic approaches to recommendations. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 5(2):1–197, 2014.

[8] R. W. Belk. It's the thought that counts: A signed digraph analysis of gift-giving. *Journal of Consumer Research*, 3(3):155–162, 1976.

[9] E. Buechel and J. Berger. Facebook therapy? why do people share self-relevant content online? In *Association for Consumer Research Conference*, 2012.

[10] C. Desrosiers and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook*, pages 107–144. Springer, 2011.

[11] D. H.-L. Goh, R. P. Ang, A. Y. Chua, and C. S. Lee. Why we share: A study of motivations for mobile media sharing. In *Active media technology*, pages 195–206. Springer, 2009.

[12] Y. Koren and R. Bell. Advances in collaborative filtering. In *Recommender systems handbook*, pages 145–186. Springer, 2011.

[13] C. S. Lee and L. Ma. News sharing in social media: The effect of gratifications and prior experience. *Computers in Human Behavior*, 28(2):331–339, 2012.

[14] P. Lops, M. de Gemmis, and G. Semeraro. Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*, pages 73–105. Springer, 2011.

[15] J. Masthoff. Group recommender systems: Combining individual models. In *Recommender systems handbook*, pages 677–702. Springer, 2011.

[16] C. Mayet and K. Pine. The psychology of gift exchange, 2010.

[17] A. Nadkarni and S. G. Hofmann. Why do people use facebook? *Personality and individual differences*, 52(3):243–249, 2012.

[18] M. O'Connor, D. Cosley, J. A. Konstan, and J. Riedl. Polylens: A recommender system for groups of users. In *ECSCW*, pages 199–218, 2001.

[19] B. Schwartz. The social psychology of the gift. *American Journal of Sociology*, 73(1):1–11, July 1967.

[20] Y. Shi, M. Larson, and A. Hanjalic. Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. *ACM Comput. Surv.*, 47(1):3:1–3:45, May 2014.

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

[22] B. Taskar, M.-F. Wong, P. Abbeel, and D. Koller. Link prediction in relational data. In *Advances in neural information processing systems*, 2003.

[23] X. Yang, Y. Guo, Y. Liu, and H. Steck. A survey of collaborative filtering based social recommender systems. *Computer Communications*, 41:1 – 10, 2014.