Modeling Temporal effects of Group Dynamics on user preferences using Hypergraphs

Ankit Sharma, Krishna S. Muppalla, Jaideep Srivastava
Department of CS&E, University of Minnesota, Minneapolis 55455

I. ABSTRACT

Personalized recommendations based upon previous behavioral and personal data of a user is an increasingly important problem. Specially, with internet becoming ubiquitous in tandem with ongoing enhancements in human computer interaction, the amount of fine grained user data like social interaction, group interaction, user-item/ user-user transaction data, etc. is ever increasing and therefore, posing new challenges in modelling user preferences. This work aims to model evolving user behavior using Hypergraphs which have been proposed as an effective tool to capture such complex higher order user relationships with multiple entities (like users, items, etc) involved. Most of the current hypergraph based models lack the temporal dimension which is an essentially important aspect, specially when modeling user preferences which are inherently variable over time. We are proposing a hypergraph label propagation approach with temporal regularization framework which is shown to be easily adaptable to capture various temporal constraints through penalization. Given a user's previous transactions and social interaction history the model ranks the various items for a queried user.

II. INTRODUCTION

The movie-rating network consists of different kind of nodes like user nodes and movie nodes. Traditional recommendation algorithms, such as Collaborative Filtering only consider the user-item rating matrix and fail to take advantage of other kinds of social media information. Recently, there has been considerable interest in making use of social media information to enhance the recommendation performance. For example, some previous works employed ordinary graphs to model tagging data for recommendation problems [7]. Users can participate in different activities like watching the movie, rating the movie and tagging the movie. The ordinary graph model fails to capture the user-tagging relations. We model the relationships as hyper-edges (eg: user-user, user-movie, user-movie-tag, etc) of a hyper-graph, which is tri-modal i.e., contain three types of vertices's: users, movies and tags. Therefore, if we splits the past relationship and transaction data available into smaller time-steps then we arrive at snapshot of this hyper-graph in various time instances in past. Our aim is to comprehend users preference for the various movies (items). We are proposing a hypergraph label propagation approach with temporal regularization and simultaneously capturing various tagging based relationship constraints through penalization. This approach reveals the ranks for the various item nodes for a particular queried user.

III. RELATED WORK

Our work is also related to graph-based ranking and hypergraph learning [6, 8, 9, 13]. Zhou et al. propose a manifold ranking algorithm which ranks data objects with respect to the intrinsic geometrical structure in the data [13]. They first construct a weighted graph and set the query point, then let all data points spread their ranking scores to their nearby neighbors via the weighted graph. The spread process is repeated until a global stable state is achieved. Agarwal [8] proposes to model the data objects as a weighted graph, and incorporate this graph structure into the ranking function as a regularizer. In this way, the obtained ranking function varies smoothly over the graph. Zhou et al. develop a general framework which is applicable to classification, clustering and embedding on hypergraph data [6]. These studies only focus on classification, clustering and embedding on hypergraphs.

IV. PROBLEM

A. Problem Definition

The task we are trying to solve is providing movie recommendations to a user based on the movies he has watched previously, the tags and the ratings he has assigned to those movies and the users he interacted since he has watched a particular movie.

B. Problem Statement

Let *G(V,E,w)* denote a hypergraph where V is the set of vertices, E is the set of hyperedges, and w is a weight function. Each hyperedge e in E is a subset of V . We denote the hypergraph for the snapshot $t = t_k$ using the incidence matrix $H(t_k) = \{E(t_k), V(t_k)\}$. $E(t_k)$ is the set of hyper-edges in the hyper-graph during the time $t = t_k$ over the vertices's $V(t_k)$. Let $f_i(t_k)$ denote the rank of a vertex v_i (irrespective of item or user vertex) and y_i is the initial label assigned of a vertex v_i. We therefore have a vector of initial labels $\mathbf{y}$ which we refer to as the query vector. (Note that we shall only initialize either a particular node or a node and its friends or a very small subset of nodes with respect to whom we want to find out the ranking of item nodes). Our aim is to find out the ranking vector $\mathbf{f} = \{\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_m\}$ where each $\mathbf{f}_k$ being the rank of all the vertices's in snapshot t_k of other nodes with respect to this initial label vector. Out of this complete rank vector the rank of the item nodes shall give the likeliness of an item being bought in future by a user (described by the query vector).

V. METHODOLOGY

A. Regularization Framework

We shall now present the regularization framework that we use to propagate the initial label vector to come up with the rankings. We are trying to solve the problem on very similar lines to that of [1] while extending it to temporal dynamic setting and enco-operating domain level constraints as well. The cost function is shown in the equation (1) where m is the number of snapshots we took of the data over time, $\delta(e, t_k)$ is the degree of hyperedge e, $w(e, t_k)$ is the weight of hyperedge $e(t_k)$ and $d(v_i, t_k)$ is the degree of a vertex $v_i(t_k)$ which is nothing but the number of edges of which this vertex is a part of. These values of weights are quiet domain dependent and degree of a hyperedge is the number of other hyperedges it is overlapping with.

$$Q(\mathbf{f}) = \sum_{k=1}^{m} \left(\frac{1}{2} \sum_{i,j=1}^{n} \sum_{e \in E} \frac{1}{\delta(e, t_k)} \sum_{\{v_i(t_k), v_j(t_k)\} \subseteq w(e,t_k)} \left\| \frac{f_i(t_i)}{\sqrt{d(v_i, t_k)}} - \frac{f_j(t_i)}{\sqrt{d(v_j, t_k)}} \right\|^2 \right) + \sum_{k=1}^{m} \left(\gamma \sum_{i \in Users} \| f_i(t_k) - f_i(t_{k-1}) - \alpha \|^2 \right) \tag{1}$$

The above cost function contains three terms of which the second term is the penalization if the predicted label is not same as the initial labels assigned. The first terms comes from the hypergraph label propagation literature [6] and has been recurrently used across literature in several papers [1][2]. The second term basically checks that the nodes whose labels or ranks we know should indeed have the same ranks and the first term makes sure that two vertices that share many hyperedges (like many people watching the same item hyperedge or are being a part of same group hyperedge) are likely to have similar rankings. Note that the outer most summation (over all snapshots) in both the first and the second terms makes sure that this regularization is enforced in each snapshot. Another thing to observe is that the first and the second terms enforce regularization only within the hypergraph snapshot of particular interval. Therefore, we add a third term which connects the constraints between different snapshots. We consider the same user's vertex in different snapshots as its *avatars*. Previous work in user preference modeling [4] suggests that the user's current preference (*avatar*) is more similar to his recent preferences (*avatars*) as compared to those much farther back in time. We incorporate this by decaying the *avatar*'s label as compared to that in previous snapshots by a constant factor α. Our model currently learns this factor and it is same across all the users.

Our aim is therefore to minimize our cost function $Q(\mathbf{f})$. Note that $\mathbf{f} = \{\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_m\}$ where each $\mathbf{f}_k$ being the rank of all the vertices's in snapshot t_k. Therefore the variables are $\mathbf{f}$ and α. The optimal ranking is given by the ranking of the item vertices's in the final or current snapshot (t_m). Writing in a more compact format the cost function for each $\mathbf{f}_{t_k}$ becomes:

$$Q(\mathbf{f}) = \sum_{k=1}^{m} Q(\mathbf{f_{t_k}}) \tag{2}$$

where,

$$Q(\mathbf{f}_{t_k}) = \mathbf{f}_{t_k}^T (\mathbf{I} - (\mathbf{D_v})_{\mathbf{t_k}}^{-1/2} \mathbf{H_{t_k} W_{t_k}} (\mathbf{D_e})_{\mathbf{t_k}}^{-1} \mathbf{H_{t_k}^T} (\mathbf{D_v})_{\mathbf{t_k}}^{-1/2}) \mathbf{f}_{t_k} + \mu (\mathbf{f}_{t_k} - \mathbf{y}_{t_k})^T (\mathbf{f}_{t_k} - \mathbf{y}_{t_k}) + \gamma (\mathbf{f}_{t_k} - \mathbf{f}_{t_{k-1}} - \alpha \mathbf{j}_{user})^T (\mathbf{f}_{t_k} - \mathbf{f}_{t_{k-1}} - \alpha \mathbf{j}_{user}), \tag{3}$$

represents the cost for the snapshot at $t = t_k$. Note that at $t = 1$ the third term shall be zero as there is no previous snapshot left to put this constrain against. $\mathbf{j}_{user}$ is a vector indicating which are vertices's are user vertices's and which are item or non-user i.e. $\mathbf{j}_{user}(i) = 1$ if v_i is a user vertex and the total number of vertices's is the length of this vector. Given that we have to minimize over a vector $\mathbf{f}$ which is naturally divided into m subsets for each of the snapshots and a scaler α, making alternate optimization [5] a natural choice. (Alternate optimization has been used in case of two variables in context of hypergraph label propagation by [3]) It can be easily shown that $Q(\mathbf{f}_{t_k})$ is convex (quadratic) with respect to $\mathbf{f}_{t_k}$ and also $Q(\mathbf{f})$ convex (quadratic) with respect to α. Taking gradient with respect to $\mathbf{f}_{t_k}$,

$$\frac{dQ}{d\mathbf{f}_{t_k}} = (\mathbf{I} - \Delta'_{t_k}) \mathbf{f}_{t_k} + \mu (\mathbf{f}_{t_k} - \mathbf{y}_{t_k}) + \gamma (\mathbf{f}_{t_k} - \mathbf{f}_{t_{k-1}} - \alpha \mathbf{j}_{user}) = 0 \tag{4}$$

where

$$\Delta'_{t_k} = (\mathbf{D_v})_{\mathbf{t_k}}^{-1/2} \mathbf{H_{t_k} W_{t_k}} (\mathbf{D_e})_{\mathbf{t_k}}^{-1} \mathbf{H_{t_k}^T} (\mathbf{D_v})_{\mathbf{t_k}}{}^{-1/2} \tag{5}$$

The solution to the above is a solution to the linear equation,

$$\mathbf{f_{t_k}} = ((1 + \mu + \gamma)\mathbf{I} - \Delta'_{t_k})^{-1} (\mu \mathbf{y}_{t_k} + \gamma \mathbf{y}_{t_{k-1}} + \gamma \alpha \mathbf{j}_{user}) \tag{6}$$

with the recursive iteration for inverse calculation being:

$$\mathbf{f_{t_k}} = (\Delta'_{t_k} - (\mu + \gamma)\mathbf{I}) \mathbf{f_{t_k}} + \mu \mathbf{y}_{t_k} + \gamma \mathbf{y}_{t_{k-1}} + \gamma \alpha \mathbf{j}_{user} \tag{7}$$

Similarly, gradient with respect to α is :

$$\frac{dQ}{d\alpha} = -\gamma \sum_{k=2}^{m} (\mathbf{f}_{t_k} - \mathbf{f}_{t_{k-1}} - \alpha \mathbf{j}_{user}) = 0, \tag{8}$$

which results in:

$$\alpha = \frac{1}{(m-1)} \frac{\sum_{k=2}^{m} (\mathbf{f}_{t_k} - \mathbf{f}_{t_{k-1}})}{\mathbf{j}_{user}}, \tag{9}$$

Algorithm 1 HYPER-TEMPORAL-RECOM ($H_{t_1}, ...H_{t_m}$, $W_{t_1}, ...W_{t_m}$, $\mathbf{j}_{user}$, $\mathbf{y}_{t_k}$)

```
1: Calculate $(\mathbf{D_v})_{\mathbf{t_k}}$ and $(\mathbf{D_e})_{\mathbf{t_k}}$.
2: $z = 0$, initialize $\mathbf{f_{t_k}}$ randomly, $\alpha = 1$
3: repeat
4:     $z = z + 1$
5:     for $k \in \{1, 2, ..., m\}$ do
6:         Find $\mathbf{f_{t_k}}$ using the equation (6) indirectly using methods like Jacobi iterations for inverse calculation using the equation (7). The values for $\mathbf{f_{t_p}}$ for $p \neq k$ should be taken from the (z-1) iteration.
7:     end for
8:     Find $\alpha$ using the equation (9) and the values for $\mathbf{f_{t_p}}$ for $p \neq k$ should be taken from the $(z-1)^{th}$ iteration.
9:     Calculate the current cost $q_z$ using the equation (3).
10: until $(q_z - q_{z-1}) > \epsilon$
11: return $\mathbf{f_{t_m}}$
```

B. Algorithm

The algorithm for the proposed optimization is shown below. For every k we have $\mathbf{y}_{t_k}$ initialized to 1 at the index of the current query user and zero otherwise (we have other initialization schemes described in the experimentation section).

VI. EXPERIMENTATION

A. Dataset

The data used for the experiments is obtained from GroupLens. It is an extension of MovieLens10M dataset, published by GroupLens research group. This dataset describes 5-star rating and free-text tagging activity from MovieLens, a movie recommendation service. We consider the data from 2006-2008. It has 2106 users, 9566 movies and 9039 tags. The data is then divided into 12 snapshots with 3 months of data in each snapshot.

B. Comparison and Scenarios

We compare our temporal hypergraph model with non-temporal or static hypergraph model. The static hypergraph model collapses all the data in different snapshots into a single hypergraph and we therefore, also do not consider any avatars in this case. Therefore, this model is same as that of Bu et al. [1].

- Scenario 1: Initialization of the current time frame avatar.
- Scenario 2: Initialization of all timeframe avatars.
- Scenario 3: Initialization of current time frame avatar and movies he watched in current timeframe.
- Scenario 4: Initialization of all avatars and all movies they have watched.

C. Performance Metric

The performance in case of all the above-mentioned scenarios is evaluated based on the hit-rate. Hit-rate is a measure of how often a list of recommendations contains content that the user is actually interested in. In this case it is number of movies the algorithm correctly recommended to the user in that in top-N recommendations..

TABLE I: Hit Rates of Temporal vs Non-Temporal

N	Scenario	Temporal Hit Rate	Non-Temporal Hit Rate
5	1	13	0
10	1	36	17
20	1	72	19
50	1	232	79
100	1	425	218
5	2	7	0
10	2	8	17
20	2	30	19
50	2	87	79
100	2	137	218
5	3	8	1
10	3	26	13
20	3	67	34
50	3	226	121
100	3	425	338
5	4	36	1
10	4	77	13
20	4	139	34
50	4	350	121
100	4	610	338

VII. RESULTS

A. Results

Hit-rates for all the scenarios are as shown in the table. Here N is the top N movies being recommended by the algorithm and the label flag indicates the scenario being considered.

B. Discussion

From the hit-rates obtained we can see that the recommendation obtained considering the temporal aspect are better in almost all the scenarios and across various N. This supports our hypothesis that over time user interacts with other users and his preference therefore changes over time as per his newer social interactions. Our temporal model therefore, successfully captures the temporal nature of both social interactions and their effect over evolving *avatars* of an individual. In the non-temporal model the time dimension is completely missing resulting in poor accuracy. The user in snapshot a is different from the user in snapshot b. Whereas, in case of non-temporal the user from start to beginning is the same.

VIII. FUTURE WORK

In this work we have restricted ourselves to a simple linear decay of user preferences however, it fairly possible that different users have different possible rates of preference change. In future we can work on learning different parameter for individual users. We also look forward to test our model for other datasets with different kinds of social interactions available. As a next step, it would also be interesting to compare other models which although model user preferences temporally but not consider the social interactions, without temporal social model.

IX. ACKNOWLEDGEMENT

This work has been supported in part by the NSF Award IIS-1422802.

REFERENCES

[1] Bu, Jiajun and Tan, Shulong and Chen, Chun and Wang, Can and Wu, Hao and Zhang, Lijun and He, Xiaofei, *Music recommendation by unified hypergraph: combining social media information and music content*. In Proceedings of the international conference on Multimedia (MM '10). ACM, New York, NY, USA, 391-400.

[2] TaeHyun Hwang, Ze Tian, Jean-Pierre Kocher, and Rui Kuang. *Learning on Weighted Hypergraphs to Integrate Protein Interactions and Gene Expressions for Cancer Outcome Prediction*,Proc. of Eighth IEEE International Conference on Data Mining (ICDM), pages 293-302, 2008.

[3] Ze Tian, TaeHyun Hwang and Rui Kuang. *A Hypergraph-based Learning Algorithm for Classifying Gene Expression and ArrayCGH Data with Prior Knowledge*, Bioinformatics, Vol. 25, No. 21, pages: 2831-2838, 2009.

[4] Koren, Yehuda. *Collaborative filtering with temporal dynamics.* Communications of the ACM 53.4 (2010): 89-97.

[5] Bezdek, James C., and Richard J. Hathaway. *Some notes on alternating optimization.* Advances in Soft ComputingAFSS 2002. Springer Berlin Heidelberg, 2002. 288-300.

[6] Zhou, Dengyong, Jiayuan Huang, and Bernhard Schlkopf. *Learning with hypergraphs: Clustering, classification, and embedding.* Advances in Neural Information Processing Systems. 2006.

[7] I. Konstas, V. Stathopoulos, and J. M. Jose. *On social networks and collaborative recommendation.* In Proc. the 32nd ACM SIGIR Conference on Research and Development in Information Retrieval, Boston, MA, 2009.

[8] S. Agarwal. *Ranking on graph data.* In Proc. the 23rd International Conference on Machine Learning, Pittsburgh, PA, 2006.

[9] S. Agarwal, K. Branson, and S. Belongie. *Higher order learning with graphs.* In Proc. the 23rd International Conference on Machine Learning, Pittsburgh, PA, 2006

[10] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Scholkopf. *Ranking on data manifolds.* In Advances in Neural Information Processing Systems 16, Cambridge, MA, 2003.