User preference modeling using Temporal Hypergraphs

***Abstract*—Personalized recommendations based upon previous behavioral and personal data of a user is an increasingly important problem. Specially, with internet becoming ubiquitious 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 challanges 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 inheretly 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 domain constraints through penalization. Given a previous user transaction and interaction history the model rank the various items for a queried user.**

I. Introduction

MMOG game network consists of different kind of nodes like player nodes and item nodes. Players can participate in different kind of activities like item buying and thus building trade relations, trust relations, group relation, etc. We model these relationships as hyper-edges (eg: item-buying, trust, trade, group, etc) of a hyper-graph which is bi-modal i.e. contain two types of vertices: users (player) and items. 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 items. We are proposing a hypergraph label propagation approach with temporal regularization and simultaneously capturing various domain constraints through penalization. This approach reveals the ranks for the various items nodes for a particular queried user.

II. Problem

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 $V(t_k)$. Let $f_i(t_k)$ denote the rank of a verterx 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 intial lablels $\mathbf{y} = [y_1, y_2, ...y_n]$ 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 $\mathbf{f} = [f_1, f_2, ..., f_n]$ 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).

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.

$$
\begin{aligned}
\text{Q(f)} = & \sum_{k=1}^{m} \left(\frac{1}{2} \sum_{i,j=1}^{n} \sum_{e \in E} \frac{1}{\delta(e,t_k)} \right. \\
& \left. \sum_{\{v_i(t_k), v_j(t_k)\} \subseteq w(e,t_k)} \left\| \frac{f_i(t_i)}{\sqrt{d(v_i,t_k)}} - \frac{f_j(t_i)}{\sqrt{d(v_j,t_k)}} \right\|^2 \right) \\
& + \sum_{k=1}^{m} \left(\mu \sum_{i=1}^{n} \left\| f_i(t_k) - y_i(t_k) \right\|^2 \right)
\end{aligned}
\tag{1}
$$

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 buying 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 within the hypergraph snapshot of particular interval. Therefore, we add a third term which enforces constraint across snapshots. Previous work in the area of recommendation systems [4] in temporal settings suggests that if a particular user $v_i(t_k)$ buys an item I_a then he somehow inherits the personality or share the preferences with

a user who bought the same item in some previous snapshot. We incooperate this by constraining the labels of the items same across the snapshots and adding a linear decay in the preference of the user (i.e. rank of the user vertex). These both things are simuntaneously captured in the third term.

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 in snapshot t_k. Therefore the variables are $\mathbf{f}$ and α. The optimal ranking is given by the ranking of the item vertices in the final snapshot (t_m). Writing in a more compat format the cost fucntion for each $\mathbf{f}_{t_k}$ becomes:

$$Q(\mathbf{f}) = \sum_{k=0}^{m} Q(\mathbf{f_{t_k}}) \tag{2}$$

where,

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

represents the cost for the snapshot at $t = t_k$. Note that at $t = 0$ 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 are user vertices 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 is the length of this vector. Given that we have to minimize over a vector $\mathbf{f}$ which is natuarally 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 propogation 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 $\mathbf{f_t}$,

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

where

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

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

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

with the recursive iteration for inverse calculation being:

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

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:

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

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.

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

Calculate $(\mathbf{D_v})_{\mathbf{t_k}}$ and $(\mathbf{D_e})_{\mathbf{t_k}}$.
$z = 0$, initialize $\mathbf{f_{t_k}}$ randomly, $\alpha = 1$
repeat
$z = z + 1$
for $k \in \{1, 2, ..., m\}$ **do**
- Find $\mathbf{f_{t_k}}$ using the equation (6) indirectly using methods like jacobi iterations for inverse calculation using the iteration (7). The values for $\mathbf{f_{t_p}}$ for $p \neq k$ should be taken from the (z-1) iteration.

end for
- Find α 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.
- Calculate the current cost q_z using the equation (3).

until $(q_z - q_{z-1}) > \epsilon$
return $\mathbf{f_{t_m}}$
=0

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.