A Composite of Heterogeneous Sources Recommenders (CHR)

Md Masum Billa Shagar
College of Software
Taiyuan University of Technology, China

Jingyu Sun
College of Software
Taiyuan University of Technology, China

Dong Wei
College of Software
Taiyuan University of Technology, China

ABSTRACT
Recommender systems (RS) are succeeding in extensive acceptance in e-commerce applications to face the data overload problem. The system compares ranging from the user profile to item characteristics, geographic territory, social impact, past behaviors to present information of items that are likely of interest to the user. Generally, research shows that a discreetly devised model using specific interaction produces highly accurate recommendations on a particular dataset. The real business circumstance is more complicated, where diverse combinations of interactions play a vital role and favored in different proportions by a specific user. In this paper, we endeavor to generate a competent framework merging various heterogeneous item relationships by concurrently modeling based on two important questions. The first one is, at a specific point in time, what source of recommendation is a user likely to be responsive. And the other one is the optimal recommendation from an individual source. Our method adopts ideas from knowledge graph representations as well as several expert networks where each of them specializes in a different part of input space. We see that our approach produces more specific recommendations than other options and also presenting instinctive explanations behind the recommendations.

General Terms
Recommendation System, Knowledge graph, User-item relation, CHR

Keywords
Knowledge graph, user-item relation, CHR

1. INTRODUCTION
Recommender systems have been accumulated huge sources of data such as text, rating, etc. which represent different aspects of user preferences. Comprehension, prediction, and recommending activities signifies gaining the dynamics of an extensive type of interacting units: users' preferences [22,31], the circumstances of their performance (e.g. their location [5,25,42], their role in their social network [34,41,48], their time spending [1,43], etc.), and the connections between the actions themselves (e.g. which actions tend to co-occur [16,20], what are their persistent dynamics [8,32,36], etc.).

Several studies say interaction between users and items as well as interactions between items and items can be explained with precisely model actions. User-to-item interactions might reveal users' choices or items' characteristics, while item-to-item interactions define similarity or contextual links between items. Factorized Personalized Markov Chains (FPMC) [32] which obtains user-to-item and item-to-item interactions via low-rank matrix decomposition, or more recent methods such as TransRec [7] or CKE [45] which inquire to obtain related approaches using knowledge graph embedding procedures.

Studies generally continue placing new varieties of user-to-item or item-to-item links to enhance recommendation accuracy on a specific dataset such as location relation aid POI recommendation [42], "also-viewed" products improve rating prediction [28]. These predictions are normally correlated with a modified model which inquiries the correlation in features.

We investigate a more general-purpose method to define user-to-item and item-to-item correlations. In any given setting for interactions concurrently controlled by various types of relationships. At distinct times, a user might choose an item on a location nearby, particular preferences, friend's opinion or other distinct combinations. Distinct user weights these patterns in varied symmetries. We explore a model that detects personalized composition over distinct logic to a certain interaction at a specific point in time.

We capture this intuition with a new model—a composite of Heterogeneous sources recommenders (CHR). With regards to a personalized, probabilistic composition of heterogeneous item-to-item recommendations, CHR models are sequential recommendation problems. Our approach is developed based on the model translational metric embedding [2,6,7,37] principle. Our CHR model is a general framework that could be applied to any model recommender approach.

We distinguish CHR model towards various state-of-the-art recommendation approaches on multiple current and new datasets from real applications including Amazon, Google Local. Our results reveal that CHR can produce a more precise recommendation with regards to overall and top-n ranking performance.

2. RELATED WORK
Historical data of user-item interaction and exploring the pattern of users' preferences and items' properties is the traditional approach to the recommendation model. Based on these approaches Collaborative Filtering (CF) and exceptionally Matrix Factorization (MF) [22] have become widely popular. MF-based approaches have been introduced to the use of implicit feedback data e.g. clicks, check-ins, etc. due to the sparsity problem of explicit feedback data [13]. This approach has been extended to the optimized personalized ranking of items [31]. By modeling with latent embedding within metric space, such model performance could be improved [4,12,37,].

TimeSVD++ explored to temporal signal [21] by the users' actions render relevant context to produce more precise recommendations. TimeSVD++ was among the state-of-the-art methods on the Netflix prize. In sparse data, understanding the sequence of items as well as specific the preceding action by a user is quite adequate to evaluate the next action. User preference modeling and sequential pattern modeling [5,32] are the two parts in which sequential models usually decompose the problem. Factorized Personalized Markov Chain (FPMC) [32] is a traditional sequential recommendation model that combines MF which use to model user preferences and factorized Markov Chains to model sequential patterns. TransRec joins the two parts by forming each user as a translating vector from its immediate visited item to the next item [7].

In the recent deep learning revolution, various deep learning techniques have been demonstrated for a recommendation for precise accuracy [46]. The standard defining characteristic of deep learning is that it acquires deep representations, i.e., learning multiplied levels of representations and abstractions from data. Pertaining to the content-aware recommendation, deep learning-based models explore to use neural networks to extract item features e.g. images [17,40], text [18,39], etc. NeuMF [9] evaluates user preferences via Multi-Layer Perceptions (MLP) and AutoRecs predicts ratings using autoencoder by replacing traditional MF. A sequential and session-based recommendation has gained significant performance by CNN based models [38,36]. Recurrent Neural Networks (RNNs) have been utilized to acquire item transition patterns in sequences [11,15,23,30,35].

Recommendation methods learn user and item embeddings from user feedback. Based on item similarity or item relationship, few models explore to regularize item embeddings to subdue the sparsity of user feedback. Exploring to item-to-item similarities based on location, POI recommendation method PACE [42] learns user and item representations. "also viewed" product in sequence enhances rating prediction for Amazon product recommendation. In "cold-start" scenarios where data from related items mitigates the sparsity of interactions with new items, these approaches are significant. In Heterogeneous Information Network [44, 47] extracts item features from handcrafted meta-paths or meta-graphs to exploit complex relationships. CKE which uses knowledge graph embeddings from heterogeneous item relationships, our approach is more resembling with it.

Knowledge base domains that concentrate on modeling recurring, complicated relationships between various entities, translating embeddings e.g. TransE [2] and TransR [24]) have obtained state-of-the-art efficiency and scalability. Collaborative Knowledge Base Embedding (CKE) [45] which applies translating embeddings as regularization, several translation-based recommendation models have been introduced e.g. TransRec [7], LRML [37], TransRev [6], which show better representation on different recommendation tasks. Our model also embraces the translational principle to model heterogeneous activities amid users, items, and links.

3. METHOD

In this paper, we develop a model on a unification of two concepts: (1) to create item-to-item recommendation methods by making use of the translational principle[2,7,45], and (2) to learn how to connect numerous origins of heterogeneous item-to-item links in order to blend multiplied 'logic' that users may pursue at a precise moment. We explain how to merge these approaches within a sequential recommendation structure and address parameter training, model complexity, etc. Our notation is compiled in Table 1.

Table 1. Notation

Notation	Description
$\mathcal{U}, I$	User and item set
S^u	historical interaction sequence for a user u: $(S_1^u, S_2^u, \ldots, S_{\lvert S^u \rvert}^u)$
$\mathcal{R}$	Item relationship set: $\{r_1, r_2, \ldots r_{\lvert\mathcal{R}\rvert}\}$
$\hat{\mathcal{R}}$	$\hat{\mathcal{R}} = RU\{r_0\}$ where r_0 stands for a latent link
$I_{i,r}$	Item set includes all items having relation $r \in \mathcal{R}$ with item i
$K \in N$	latent vector dimensionality
$\theta_u \in \mathbb{R}^K$	latent vector for user u where $u \in \mathcal{U}$
$\theta_i \in \mathbb{R}^K$	latent vector for item i where $i \in I$
$\theta_r \in \mathbb{R}^K$	latent vector for relation r where $r \in \hat{\mathcal{R}}$
$b_i \in \mathbb{R}$	bias term for item i where $i \in I$
$b_r \in \mathbb{R}$	bias term for relation r where $r \in \hat{\mathcal{R}}$
$d(x, y)$	Squared distance between point x and y
[n]	Set of natural numbers less or equal than n

Item-to-item filtering is a form of collaborative filtering for recommender systems based on the similarity between items estimating user's ratings on those items. Item-to-item filtering is a form of collaborative filtering for recommender systems based on the similarity between items estimating user's ratings on those items. Space characteristics of items will diversify and relationships depend on types of items. Pair items may be linked because they are alike in different aspects. So the items linking can be heterogeneous by function, category, style, location, etc. Exploiting the heterogeneous relationships of items, we attempt to develop an approach with the heterogeneous source links which can be applied to model an inadequate number of pertinent relations or extensively extracted item-to-item relations.For a given item i and a graph type r, we define 'link-item' lists holding items that present correlation r with the item i e.g. 'also viewed' or 'also bought' links for an item i. Using these relationships, we represent a recommender to model the activity among the three components two items i and i′ linked via a graph r applying a translational action [2]:

$$R(i'|i, r) = b_{i'} - d(\theta_i + \theta_r, \theta_{i'}) \quad (1)$$

Where $b_{i'}$ is a bias term. The concept is pulled from knowledge graph embedding techniques where two entities e.g. i = "James Cameron" and i′= "Avatar" should be 'close to' each other with a definite similarity action e.g. r = directed. While applied to model sequential similarities among items, such models merge conventional recommendation methods with the translational law [7].Related to Bayesian Personalized Ranking [31], we depreciate an objective differentiating the rate of relevant (i′) versus not-relevant (i^-) items:

$$T_I = -\sum_{(i,r,i',i^-)\in D_I} \ln\sigma\big(R(i'|i, r) - R(i^-|i, r)\big), \quad (2)$$

Where

$$D_I = \{(i, r, i', i^-) | i \in I \land r \in R \land i' \in I_{i,r} \land i^- \in I - I_{i,r}\}$$

Here related items i′ to be rated higher i.e. larger $R(i'|i, r)$ than not related items i^- given the circumstances of item i and relation r. Following we utilize the recommender $R(i'|i,r)$ to estimate how exactly i and i' are linked in terms of an accurate association r.Item-level favorites are only analyzed on being item-to-item predictions. Nevertheless, most of the real-world applications render various types of co-occurrence links e.g. also-viewed, also-bought, etc. here we investigate the problem of predicting what type of the relation a user is preferably to follow for the next activity. All items that a user chooses are considered to be linked to prior items the user has coupled with via explicit links or via latent transitions. With users' sequential feedback and item-to-item associations, we can represent the relative connections $\tau(u,k)$ given the circumstances of the user u and the kth item S_k^u from feedback sequence S^u :

$$\tau\,(u,k) = \begin{cases} \{r_0\} \\ \{r|S_{k+1}^u \in I_{S_k^u,r} \wedge r \in R\} \end{cases}$$

The initial condition in τ denotes that if two following items share no links, the relative links convert a 'latent' link r_0, which accounts for transitions that cannot be described by any explicit link. Contrarily, shared links are related. Then, furthermore, we set a translation-based recommender to model the activity among the three components:

$$R(r|u,i) = b_r - d(\theta_u + \theta_i, \theta_r) \qquad (3)$$

In contrast to (eq. 1) predicts the next item under given link while (eq. 3) predicts which link will be selected based on former item. Precisely, we represent a probability function P overall connections including r_0. The connection between the link (r) and circumstances (u,i) is represented by

$$P(r|u,i) = \frac{\exp\,(R(r|u,i))}{\sum_{r' \epsilon \mathcal{R}} \exp\,(R(r'|u,i))} \qquad (4)$$

We optimize the ranking between related and inappropriate connections by minimizing

$$T_R = -\sum_{(u,i,r,r^-) \epsilon \mathcal{D}_R} \ln \sigma(P(r|u,i) - P(r^-|u,i)) \qquad (5)$$

$$\mathcal{D}_R = \{(u, S_k^u, r, r^-)|u\epsilon\mathcal{U} \wedge k\epsilon[|S^u| - 1] \wedge r\epsilon\tau(u,k) \wedge r^-\epsilon\mathcal{R}\tau(u,k)\}$$

Being sequential recommenders, for instance, FPMC [32] and PRME [5]), we build sequential recommender is combining users' long-term preferences and short-term item transitions. Our recommendation model uses a blend of explicit and latent item transitions. The composition model is inspired by the 'mixtures of experts' framework [14], which probabilistically mixes the outputs of various (weak) learners by weighting individual learners according to their connection to a given input. In our circumstance, each 'learner' is a contact type whose connection is predicted given a query item. Here the weights on individual item transition are trained on a user u and last revisited item i. Specifically, we define $R*(i'|u,i)$ as:

$$b_{i'} - \overbrace{d(\theta_i + \theta_{u,}\theta_{i'})}^{\text{long-term preference}} + \underbrace{P(r_0 + u, i)R(i'|i, r_0)}_{\text{latent short-term transitions}} + \overbrace{\sum_{r\epsilon R} P(r|u.i)R(i'|i,r)}^{\text{explicit short-term transitions}}$$

Relation r_0 is a latent item link to obtain item transitions that cannot be described by explicit relations. With contrast to learning explicit relations as in eq. (1), r_0 is learned from users' sequential feedback. By combining latent and explicit transitions, we can revise the recommender as:

$$R^*(i'|u,i) = R(i'|u,i) + \sum_{r\epsilon\hat{R}} \overbrace{P(r|u.i)}^{\text{probability of choosing r as the next relation}} \times \underbrace{R(i'|i,r)}_{\text{transition from i to i' using relation r}} \qquad (6)$$

The item-to-item prediction and the next-links prediction are simply combined into our sequential recommendation model R^*. Finally, the goal of sequential recommendation is to rank the next-item i' higher than unrelated items; the loss function we use is defined as:

$$T_S = -\sum_{(u,i,i',i^-) \epsilon \mathcal{D}_S} \ln \sigma\big(R^*(i'|u,i) - R^*(i^-|u,i)\big) \qquad (7)$$

Where $\mathcal{D}_S = \{(u, S_k^u, S_{k+1}^u, i^-)|u\epsilon\mathcal{U} \wedge k\epsilon[|S^u| - 1] \wedge i^-\epsilon I - S^u\}$

We utilize a multi-task learning method mutually to optimize all task using shared variables within a merged translational metric space which can reduce the model size and avoid over-fitting as well as viewed as a form of regularization that merge different sources of data.

Precisely we mutually learn the three tasks in a multi-task learning framework:

$$\min_{\ominus} T = T_S + \alpha T_I + \beta T_R + \lambda(\textstyle\sum_{i\epsilon I} b_i^2 + \sum_{r\epsilon\hat{\mathcal{R}}} b_r^2) \qquad (8)$$

$$\text{s.t.}\ ||\theta_u||_2 \le 1, ||\theta_i||_2 \le 1, ||\theta_r||_2 \le 1$$

$$\forall_u \epsilon \mathcal{U}, i\epsilon I, r\epsilon\hat{\mathcal{R}}$$

Here α and β are two hyper-parameters to regulate the exchange between the central task TS and subordinate tasks, and training variables$\ominus = \{\theta_u, \theta_i, \theta_r b_i, b_r\}$. We restrain the latent vectors to lie inside a unit ball. This regularization doesn't push vectors toward the origin like $\mathcal{L}_2$ regularization, and is effective in both knowledge graph embedding methods [2,24] and metric-based recommendation methods [7, 12, 37]. The bias expressions are regularized by a square penalty with a coefficient λ.

We plan the training procedure as follows:

(1) Sample three batches from$\mathcal{D}_S, \mathcal{D}_I$, and $\mathcal{D}_R$, respectively(2) Update parameters using an Adam [19] optimizer for objective T with the three batches (3) Control the norm for all $\theta_u, \theta_i, \theta_r$ by $\theta = \theta / \max(||\theta||_2, 1)$ (4) Repeat this procedure until convergenceWhen α= 0, we do not have semantic constraints on $R(i'|i,r)$, meaning that all relationships become latent relationships. When β = 0, we don't have a prior on choosing the next relationship, meaning the model would optimize $P(r|u,i)$ only to fit sequential feedback. We need to choose appropriate α > 0, β > 0 to achieve satisfactory performance on the main task TS.

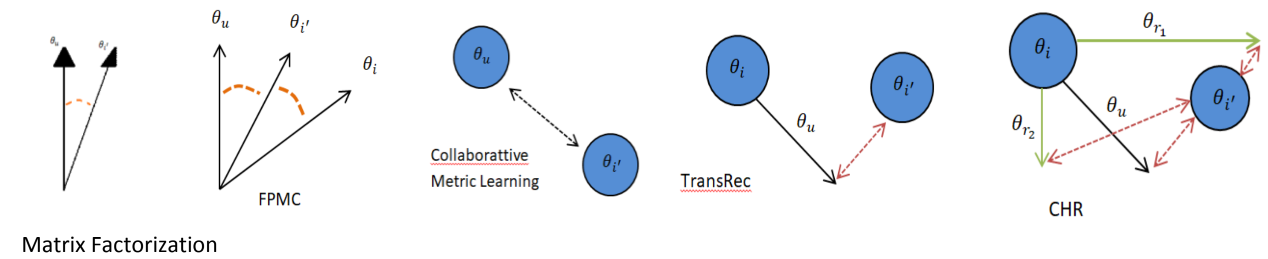

Figure 1: A simplified illustration of recommendation models in existing methods. The first two models are based on product spaces while the following three are metric based. The dashed lines indicate how a model calculates its preference score given a user u, last visited item i and next item i'. The width of lines in CHR indicates their weight according to $P(r|u,i)$.

4. EXPERIMENT

4.1 Datasets

We evaluate our method on 7 datasets from three large-scale real-world applications. The datasets diversify significantly in the domain, variability, feedback sparsity, and link sparsity. The datasets including large corpora of reviews as well as various types of related items were collected from Amazon.com [27] from May 1996 to July 2014. We analyze a range of broad sections including 'Automotive,' 'Beauty,' 'Clothing,' 'Toys,' and 'Games'. High sparsity and variability are the main characteristics of these datasets. Amazon crawled 'also viewed,' 'also bought,' 'bought together,' and 'buy after viewing' types of item links that we use in our model. A POI-based dataset [7] collected from Google Local which comprises user reviews scattered over 5 regions. To develop item relationships e.g. similar things, we first extract the top 100 categories e.g. restaurants, parks, attractions, etc. based on frequency. After then, for individual POI, based on its categories, we construct "similar things" links like "nearby attraction," "nearby park," etc. We also construct links called "nearby popular places," based on each POI's geo-location and popularity. Therefore, we obtain the sum of 101 link types.

We followed the alike preprocessing procedure from [7]. For all datasets, we treat the presence of a review as implicit feedback i.e. the user interacted with the item and use timestamps to determine the sequence order of actions. We discard users and items with fewer than 5 related actions. For partitioning, we split the historical sequence S^u for each user u into three parts:

(1) The most recent action $S^u_{|S^u|}$ for testing,

(2) The second most recent action $S^u_{|S^u|-1}$ for validation, and

(3) All remaining actions for training. Hyper-parameters in all cases are tuned by grid search using the validation set. Data statistics are shown in Table 2.

4.2 Comparison Methods

PopRec:

This is a simple baseline that ranks items according to their popularity. It recommends the most popular items to users and is not personalized.

Bayesian Personalized Ranking (BPR-MF) [19]:

BPR-MF is a state-of-the-art item recommendation model that takes Matrix Factorization as the underlying predictor and neglects the sequential signals.

Collaborative Metric Learning (CML) [12]:

Collaborative filtering is a classic method that learns metric embeddings for users and items.

Factorized Markov Chain (FMC):

By factorization of the item-to-item transition matrix, FMC captures the 'global' sequential dynamics which shared by all users, but cannot capture personalized behavior.

Factorized Personalized Markov Chains (FPMC) [32]:

FPMC employs a combination of matrix factorization and factorized Markov chains as its prediction, which catches users' long-term behavior and item-to-item transitions.

Translation-based Recommendation (TransRec) [7]:

Table 2: Data statistics

Datasets	Users	Items	Actions	Relationships	Related item	Avg. actions/users	Avg. actions/items	Avg. related items/item
Amazon Automotive	34315	40287	183567	4	1632467	5.35	4.56	40.52
Amazon Toys	57617	69147	410920	4	3943494	7.13	5.13	57.03
Amazon clothing	184050	174484	1068972	4	2927534	5.81	6.12	16.78
Amazon Beauty	52204	57289	394808	4	2082502	7.56	6.89	36.43

Amazon Games	31013	23715	287107	4	1030990	9.26	12.11	43.47
Google Local	350811	505516	2591026	101	48307315	7.39	5.13	95.56
Total	1.04M	0.88M	8.15M	-	60.04M	-	-	-

it is a one of the sequential recommendation method's state-of-the-art that models individual user as a translation vector to capture the transition from the current item to the next item.

Matrix Co-Factorization (MCF) [28]:

It concurrently factorizes a rating matrix and a binary item-to-item matrix based on "also viewed" products.

Collaborative Knowledge base Embedding (CKE) [45]:

A collaborative filtering method with regularizations from visual, textual and structural item information.

Finally, our method, composite of Heterogeneous sources recommenders (CHR), makes use of various recommenders to capture both long-term behaviors and (explicit/latent) item transitions in a unified translational metric space.

For a fair comparison, we execute all methods in TensorFlow with Adam [19] optimizer. All learning-based methods use BPR or SBPR loss functions to optimize personalized rankings. For PACE, MCF, and CKE, we do not use side information other than item relationships. For methods with homogeneous item relations i.e., PACE and MCF, we set two items as 'neighbors' if they share at least one relationship. For CKE, we employ TransE [2] to model item relationships. Regularization hyper-parameters are elected from {0.0001, 0.001, 0.01, 0.1} using our validation set. Our method can perform adequate performance using $\alpha = 1$, $\beta = 0.1$ and λ =1e-4 for all datasets excluding Steam. Because of high density, we apply $\alpha = 0.1$, $\beta = 0.1$ and $\lambda = 0$ for Steam.

4.3 Evaluation Metric

In this setting, we report the AUC, Hit Rate@10, and NDCG@10 as in [7, 37, 42]. The AUC measures the overall simply counts whether the ground-truth item is ranked among the top-10 items, while NDCG@10 is a position-aware ranking metric. We apply the strategy in [9, 20, 37] to evade huge computation on all user-item pairs for top-n ranking performance metrics. For each user u, we randomly sample 100 unrelated items that doesn't belong to Su, and rank these items with the ground-truth item. Based on rankings of these 101 items, HR@10 and NDCG@10 can be assessed.

In the second setting set, we examine a realistic recommendation situation that presents recommendations type by type. There are two purposes: 1) relevant links should be highly ranked, and 2) within each link, relevant items should be highly ranked. Precisely, we initially rank links, and then we illustrate at most 10 items from each link. Therefore, the ultimate position of item i is decided by its ranking within the relationship as well as the ranking of the relationship to which i belongs. We use NDCG to evaluate the ranking performance, which considers the positions of relevant items.

4.4 Recommender Performance:

Table 3 shows the results under the standard sequential recommendation setting. The number of latent dimensions K is set to 10 for all experiments. We notice our method CHR can outperform all baselines on all datasets in terms of both overall ranking and top-N ranking metrics. The outcomes explain the significant part of item-to- item links on understanding users' sequential behavior in contrast to sequential feedback based methods (FPMC and TransRec). Compared to methods that rely on item relationships as regularization (PACE, MCF, and CKE), the performance of our method presents the benefits of modeling item links and sequential signals combined. Probably because of a high density of sequential feedback (useful for learning latent transitions) and sparsity of related items (inadequate for capturing item similarities) in Steam, sequential methods FPMC and TransRec perform better performance than relationship-ware methods on Steam.

In this setting, we report the AUC, Hit Rate@10, and NDCG@10 as in [7, 37, 42]. The AUC measures the overall ranking performance whereas HR@10 and NDCG@10 measure Top-N recommendation performance. HR@10 simply counts whether the ground-truth item is ranked among the top-10 items, while NDCG@10 is a position-aware ranking metric. We apply the strategy in [9, 20, 37] to evade huge computation on all user-item pairs for top-n ranking performance metrics. For each user u, we randomly sample 100 unrelated items that doesn't belong to Su, and rank these

Table 3: Ranking results on different datasets under setting-1 (higher is better). The number of latent dimensions K for all comparison methods is set to 10. The best performance in each case underlined.

Datasets	Metric	PopRec	BPR-MF	CML	FPMC	TransRec	PACE	MCF	CKE	CHR	%improve
Amazon Automotive	AUC	0.6426	0.6395	0.6414	0.7233	0.7416	0.7233	0.7416	0.7341	0.8026	8.2%
	HR@10	0.3481	0.3323	0.3062	0.3210	0.3332	0.4424	0.4335	0.4335	0.5382	21.7%
	NDCG@10	0.2084	0.2003	0.1793	0.1981	0.2034	0.2371	0.2735	0.2607	0.3478	27.2%
Amazon Toy	AUC	0.6641	0.6863	0.7070	0.7164	0.7273	0.7610	0.7892	0.7914	0.8422	6.4%
	HR@10	0.3601	.3378	0.4015	0.4170	0.4474	.4590	0.5277	0.5183	0.6061	14.9%
	NDCG@10	0.2048	0.1926	0.2437	0.2651	0.2890	0.2820	0.3348	0.3284	0.4151	24.0%
Amazon	AUC	0.6964	0.6767	0.7029	0.6874	0.7328	0.7685	0.7884	0.7805	0.8150	3.4%

Beauty	HR@10	0.4003	0.3761	0.4070	0.3714	0.4125	0.4635	0.5196	0.5131	0.5550	6.8%
	NDCG@10	0.2277	0.2164	0.2532	0.2107	0.2666	0.2820	0.3292	0.3245	0.3635	10.4%
Amazon Games	AUC	0.7646	0.8107	0.8455	0.8523	0.8560	0.8632	0.8841	0.8849	0.9175	3.4%
	HR@10	0.4724	0.5752	0.6349	0.6501	0.6838	0.6355	0.7049	0.7080	0.7693	8.7%
	NDCG@10	0.2779	0.3249	0.4068	0.4576	0.4557	0.4044	0.4668	0.4528	0.5366	14.5%
Amazon Clothing	AUC	0.6609	0.6500	0.6527	0.6715	0.7034	0.7083	0.7529	0.7394	0.7882	4.7%
	HR@10	0.3661	0.3502	0.3307	0.3478	0.3608	0.3590	.4278	0.4299	0.4919	14.9%
	NDCG@10	0.2166	0.2064	0.1904	0.2076	0.2111	0.1984	0.2601	0.2561	0.3015	16.5%
Google Local	AUC	0.5811	0.7552	0.7676	0.7835	0.7927	0.7727	0.8560	0.8488	0.9330	9.0%
	HR@10	0.2454	0.5742	0.5571	0.5505	0.7103	0.5099	0.7231	0.7095	0.8532	18.0%
	NDCG@10	0.1380	0.4318	0.3995	0.4147	0.5400	0.3249	0.5484	0.5195	0.6091	11.1%
Steam	AUC	0.9067	0.9233	0.9117	0.9219	0.9247	0.9012	0.9184	0.9115	0.9312	0.7%
	HR@10	0.7292	0.7205	0.7481	0.7830	0.7842	0.7158	0.7668	0.7656	0.7983	1.8%
	NDCG@10	0.4728	0.4655	0.4699	0.5297	0.5287	0.4663	0.5059	0.4829	0.5598	5.7%

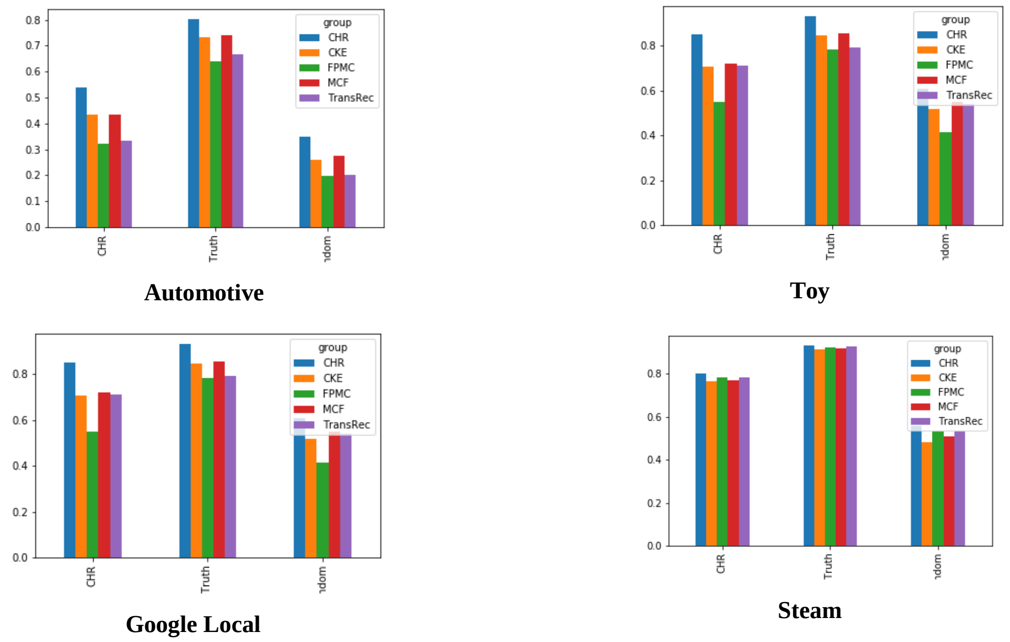

Figure 2: Ranking performance (NDCG) with different layout methods

items with the ground-truth item. Based on rankings of these 101 items, HR@10 and NDCG@10 can be assessed.

In the second setting set, we examine a realistic recommendation situation that presents recommendations type by type. There are two purposes: 1) relevant links should be highly ranked, and 2) within each link, relevant items should be highly ranked. Precisely, we initially rank links, and then we illustrate at most 10 items from each link. Therefore, the ultimate position of item i is decided by its ranking within the relationship as well as the ranking of the relationship to which i belongs. We use NDCG to evaluate the ranking performance, which considers the positions of relevant items.

We notice our method CHR can outperform all baselines on all datasets in terms of both overall ranking and top-N ranking metrics. The outcomes explain the significant part of item-to-item links on understanding users' sequential behavior in contrast to sequential feedback based methods (FPMC and TransRec). Compared to methods that rely on item relationships as regularization (PACE, MCF, and CKE), the performance of our method presents the benefits of modeling item links and sequential signals combined. Probably because of a high density of sequential feedback (useful for learning latent transitions) and sparsity of related items (inadequate for capturing item similarities) in Steam, sequential methods FPMC and TransRec perform better performance than relationship-ware methods on Steam.

5. CONCLUSION

In this work, we present a sequential recommendation method CHR, which learns the personalized composition of heterogeneous item-to-item recommendations. We represent all parameters in a unified metric space and adopt translational operations to model their interactions. Multi-task learning is applied to simultaneously learn the representations across the two items and user link representation via graph translation.

Extensive quantitative results on large-scale datasets from various real-world applications demonstrate the perfection of our method regarding both overall and Top-N recommend performance.

6. ACKNOWLEDGMENTS

We thank Sun JingYu for helpful discussion, advises in high-performance computing.

7. REFERENCES

[1] Alex Beutel, Paul Covington, Sagar Jain, Can Xu, Jia Li, Vince Gatto, and Ed H Chi. 2018. Latent Cross: Making Use of Context in Recurrent Recommender Systems. In WSDM'18.

[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In NIPS'13.

[3] Rich Caruana. 1997. Multitask Learning. Machine Learning (1997).

[4] Shuo Chen, Josh L Moore, Douglas Turnbull, and Thorsten Joachims. 2012. Playlist prediction via metric embedding. In KDD'12.

[5] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. 2015. Personalized ranking metric embedding for next new POI recommendation. In IJCAI'15.

[6] Alberto Garcia-Duran, Roberto Gonzalez, Daniel Onoro-Rubio, Mathias Niepert, and Hui Li. 2018. TransRev: Modeling Reviews as Translations from Users to Items. arXiv preprint arXiv:1801.10095 (2018).

[7] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2017. Translation-based Recommendation. In RecSys'17.

[8] Ruining He and Julian McAuley. 2016. Fusing similarity models with markov chains for sparse sequential recommendation. In ICDM'16.

[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In WWW'17.

[10] Balázs Hidasi and Alexandros Karatzoglou. 2017. Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. arXiv abs/1706.03847 (2017).

[11] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In ICLR'16.

[12] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge J. Belongie, and Deborah Estrin. 2017. Collaborative Metric Learning. In WWW'17.

[13] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In ICDM'08.

[14] Robert Jacobs, Michael Jordan, Steven Nowlan, and Geoffrey Hinton. 1991. Adaptive Mixtures of Local Experts. Neural Computation (1991).

[15] How Jing and Alexander J. Smola. 2017. Neural Survival Recommender. In WSDM'17.

[16] Santosh Kabbur, Xia Ning, and George Karypis. 2013. FISM: factored item similarity models for top-n recommender systems. In KDD'13.

[17] Wang-Cheng Kang, Chen Fang, Zhaowen Wang, and Julian McAuley. 2017. Visually-Aware Fashion Recommendation and Design with Generative Image Models. In ICDM'17.

[18] Dong Hyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional Matrix Factorization for Document Context-Aware Recommendation. In RecSys'16.

[19] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014).

[20] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In KDD'08.

[21] Yehuda Koren. 2010. Collaborative filtering with temporal dynamics. Commun. ACM (2010).

[22] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization techniques for recommender systems. Computer (2009).

[23] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. In CIKM'17.

[24] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In AAAI'15.

[25] Bin Liu, Yanjie Fu, Zijun Yao, and Hui Xiong. 2013. Learning geographical preferences for point-of-interest recommendation. In KDD'13.

[26] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In KDD'15.

[27] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-based recommendations on styles and substitutes. In SIGIR'15.

[28] Chan young Park, Dong Hyun Kim, Jinoh Oh, and Hwanjo Yu. 2017. Do "AlsoViewed" Products Help User Rating Prediction?. In WWW'17.

[29] Apurva Pathak, Kshitiz Gupta, and Julian McAuley. 2017. Generating and Personalizing Bundle Recommendations on Steam. In SIGIR'17.

[30] Wenjie Pei, Jie Yang, Zhu Sun, Jie Zhang, Alessandro Bozzon, and David MJ Tax. 2017. Interacting Attention-gated Recurrent Networks for Recommendation. In CIKM'17.

[31] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In UAI'09.

[32] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In WWW'10.

[33] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet

collaborative filtering. In WWW'15.

[34] Kai Shu, Suhang Wang, Jiliang Tang, Yilin Wang, and Huan Liu. 2018. Crossfire: Cross media joint friend and item recommendations. In WSDM'18.

[35] Elena Smirnova and Flavian Vasile. 2017. Contextual Sequence Modeling for Recommendation with Recurrent Neural Networks. In DLRS@RecSys'17.

[36] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In WSDM'18.

[37] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent Relational Metric Learning via Memory-based A ention for Collaborative Ranking. In WWW'18.

[38] Trinh Xuan Tuan and Tu Minh Phuong. 2017. 3D Convolutional Networks for Session-based Recommendation with Content Features. In RecSys'17.

[39] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative Deep Learning for Recommender Systems. In KDD'15.

[40] [40] Suhang Wang, Yilin Wang, Jiliang Tang, Kai Shu, Suhas Ranganath, and Huan Liu. 2017. What your images reveal: Exploiting visual contents for point-of-interest recommendation. In WWW'17.

[41] [41] Le Wu, Yong Ge, Qi Liu, Enhong Chen, Richang Hong, Junping Du, and Meng Wang. 2017. Modeling the evolution of users' preferences and social links in social networking services. IEEE TKDE (2017).

[42] [42] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. 2017. Bridging Collaborative Filtering and Semi-Supervised Learning: A Neural Approach for POI Recommendation. In KDD'17.

[43] Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan. 2014. Beyond clicks: dwell time for personalization. In RecSys'14.

[44] 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 WSDM'14.

[45] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative Knowledge Base Embedding for Recommender Systems. In KDD'16.

[46] Shuai Zhang, Lina Yao, and Aixin Sun. 2017. Deep Learning based Recommender System: A Survey and New Perspectives. arXiv abs/1707.07435 (2017).

[47] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. 2017. Metagraph based recommendation fusion over heterogeneous information networks. In KDD'17.

[48] Tong Zhao, Julian McAuley, and Irwin King. 2014. Leveraging social connections to improve personalized ranking for collaborative filtering. In CIKM'14.

[49] Bowman, M., Debray, S. K., and Peterson, L. L. 1993. Reasoning about naming systems. .

[50] Ding, W. and Marchionini, G. 1997 A Study on Video Browsing Strategies. Technical Report. University of Maryland at College Park.

[51] Fröhlich, B. and Plate, J. 2000. The cubic mouse: a new device for three-dimensional input. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems

[52] Tavel, P. 2007 Modeling and Simulation Design. AK Peters Ltd.

[53] Sannella, M. J. 1994 Constraint Satisfaction and Debugging for Interactive User Interfaces. Doctoral Thesis. UMI Order Number: UMI Order No. GAX95-09398., University of Washington.

[54] Forman, G. 2003. An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res. 3 (Mar. 2003), 1289-1305.

[55] Brown, L. D., Hua, H., and Gao, C. 2003. A widget framework for augmented interaction in SCAPE.

[56] Y.T. Yu, M.F. Lau, "A comparison of MC/DC, MUMCUT and several other coverage criteria for logical decisions", Journal of Systems and Software, 2005, in press.

[57] Spector, A. Z. 1989. Achieving application requirements. In Distributed Systems, S. Mullender