Improving Recommendation Accuracy using Networks of Substitutable and Complementary Products

Tong Zhao[1], Julian McAuley[2], Mengya Li[3] and Irwin King[1]
[1]Department of Computer Science and Engineering,
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
[2]Department of Computer Science and Engineering, UC San Diego, La Jolla, CA, USA

[3]State Information Center, China
{tzhao, king}@cse.cuhk.edu.hk, imyli1024@gmail.com, jmcauley@eng.ucsd.edu

***Abstract*—Recommender systems are ubiquitous in applications ranging from e-commerce to social media, helping users to navigate a huge selection of items and to meet a variety of special needs and user tastes. Incorporating contextual knowledge into such systems—such as relational information—has proven to be an effective way to improve recommendation accuracy. A popular line of research aims to model relationships between *users*, through their connections in a social network. In contrast, we aim to model complex relationships between *products*, using data based on co-purchase and co-browsing behavior. Modeling such networks presents a variety of challenges, in particular because the features that make two items complementary (or likely to be co-purchased) are far more complex than mere similarity. To model these complex relationships we develop a method based on pairwise ranking and embedding learning to build representations of items based on their co-purchasing and co-browsing statistics. We conduct experiments on *Amazon* product data to demonstrate that modeling such relationships significantly improves accuracy compared to competitive baselines.**

I. Introduction

Recommender systems are indispensable as a means of tailoring experiences to users, in order to generate personalized suggestions of content ranging from music and movies to beers. Critically, recommender systems rely on *user feedback* (e.g. in the form of a rating) in order to build rich and high-fidelity models of user's preferences. Such systems are then evaluated in terms of their ability to predict ratings or rank products in accordance with users' real behavior.

One of the most successful and ubiquitous approaches is to cast rating estimation tasks in terms of Matrix Factorization (MF) [26, 25, 10]. MF models assume that each user and each item is associated with a low-dimensional latent factor representation, so that the compatibility between the user and the item (i.e., their rating) can be modeled in terms of the inner product between these two representations. Additional contextual knowledge can then be incorporated into such models to further improve recommendation accuracy. One form of 'contextual knowledge' is the set of social relations between users. A variety of approaches [16, 35, 6, 2] have demonstrated that modeling the similarity between users in terms of their social relations can significantly improve rating estimation accuracy. Although useful, social relationships are not always available in many recommendation scenarios, and even when present are potentially only weakly relevant in terms of modeling preferences.

In contrast, in this paper we aim to build recommender systems that model relationships among *products*. Such relationships are appealing as a means of building higher-fidelity recommender systems as they are readily available in many real-world recommendation scenarios, and *a priori* seem likely to contain information that is highly relevant to users' preferences. Two types of relationships between products, as shown in [18], are product pairs that are considered to be *substitutable* or *complimentary*. 'Substitute' goods are those that are considered to be interchangeable—such as one t-shirt for another, while 'complement' goods are those that might be purchased together, such as a t-shirt and a matching pair of jeans, or a cellphone and a charger.

Although similarity-based measures (such as those used to model social relations) seem appropriate to model substitute relationships, they cannot be used to model complementary items, since questions like 'which t-shirt matches these jeans' depend on a calculation that is inherently more complex than mere similarity.

To address the above complexities, we propose a rating estimation framework, *RSC*, in which we leverage pairwise ranking methods to model substitutable items, while at the same time we use an embedding learning method to model complementarity. We combine a rating estimation measure with our two network models to construct a joint objective, in which model learning can be easily parallelized using an asynchronous version of stochastic gradient descent (ASGD). We conduct experiments on the publicly-available *Amazon* product corpus, where our results demonstrate that the proposed framework significantly increases the accuracy of rating estimation and can be efficiently applied to large datasets.

978-1-5090-6182-2/17/$31.00 ©2017 IEEE

II. Related Work

Recommender systems focus on analyzing patterns of interest in items to provide personalized suggestions to users. This task can be cast in terms of rating estimation, for which a variety of collaborative filtering and matrix factorization methods have been proposed [26]. Such methods focus on representing user-item-rating matrices in terms of low-dimensional latent factors; building on top of simple factorization methods, [10] proposed *SVD++* by taking 'neighbor bias' into consideration, while [25] presented Bayesian Probabilistic Matrix Factorization (BPMF). There are also several methods [16, 35, 27, 34, 17, 8, 19] that focus on incorporating relational contextual knowledge into recommender systems. [2] decomposed the social trust relations between users into multiple aspects to improve rating estimation accuracy. [6] proposed *TrustSVD* by extending SVD++ to incorporate social trust relations. [15] proposed *SoRec* to collaboratively factorize both rating matrix and social network matrix. [7] combined SoRec and topic matrix factorization to improve recommendation performance. What the above works have in common is that they make use of relational knowledge between *users*. More recently, [18] presented a method to infer networks of substitutable and complementary items, though the goal there was to perform network inference rather than to perform rating prediction.

Since our proposed framework shall use an embedding learning method—*Skip-Grams* [20]—to help learn items' representations from complement relations, we briefly review related work in this area. Embedding learning has proven to be an effective method for learning representations, with successes shown in many NLP tasks, such as sentiment analysis [33, 29], query rewriting [4], paragraph modeling [12], syntagmatic and paradigmatic relationship modeling [28], etc., as well as in user modeling [22, 30, 31], item modeling [5], online advertising [13] and recommendation with text modeling [32, 1]. These successes persuade us to exploit embedding learning methods to model items' complement networks for rating estimation.

III. Data Description

The datasets we use in this paper were collected from *Amazon.com* [18] and are publicly available.[1] We select seven representative top-level categories for use in this paper, including *Baby*, *Beauty*, *Electronics*, *Home and Kitchen*, *Movies and TV*, *Tools and Home Improvement* and *Toys and Games*. Each of these data sets contain users' ratings on items, ranging from 1 to 5, in addition to two types of relationships:

1) *Customers Who Viewed This Item Also Viewed ('also viewed')*
2) *Frequently Bought Together ('bought together')*

Based on the assumption that users tend to compare and view similar items together, and following [18], we refer to *'also viewed'* relations as 'substitutes', and *'bought together'* relationships as encoding complements (up to some noise). Unlike substitute relationships, complements encode a relationship that is very different from mere similarity; for example, an *iPhone* will have substitute relationships with other phones, and complement relationships with products like batteries, cases, or chargers. These relationships encode 'population level' co-browsing and co-purchasing trends, and the complexities of these relationships require a new method to model them. For further details about the data and how it was collected see [18] and *Amazon*'s own tech report [14].

The statistics of the data used in this paper are shown in Table I. On the one hand, the datasets have millions of users and items, but on the other, they are extremely sparse (i.e., relatively few reviews per user/item); this is where we expect product relationships to help us, since co-browsing and co-purchasing relationships are available even for products with very few reviews. We aim to leverage such relationships between items to overcome the rating sparsity problem, for which we propose the following framework to improve rating estimation accuracy.

[1] http://jmcauley.ucsd.edu/data/amazon/

IV. Modeling Substitutable and Complementary Items for Rating Estimation

In this section, we detail the modeling of networks of substitutable and complementary items and then propose a unified framework to leverage the substitute network (*S*) and complement network (*C*) to improve rating estimation accuracy. The notation used in this paper is summarized in Table II.

A. Modeling Substitute Networks

Substitute networks are extracted from users' co-browsing behavior, indicating a similarity relationship among items. To model such similarities, we propose to optimize a pairwise ranking function since pairwise ranking methods have shown success in modeling sparse networks for link prediction [11, 37]. We treat the substitute network S as an undirected graph whose links are items that tend to be viewed together. Our underlying assumption is that items that are viewed together ought to be more similar to each other than items that are not; thus it is natural to optimize an objective function based on pairwise comparison. Similar to [24], we maximize an AUC measure through the following objective:

$$\max_V \sum_{i \in I} \sum_{k \in S_i} \sum_{j \notin S_i} \mathbf{1}(sim(i,k) > sim(i,j)), \tag{1}$$

where $sim(i,j)$ is a similarity measure and $\mathbf{1}$ is an indicator. When optimizing for the AUC, it is common practice to work with a differentiable function such as a sigmoid ($\sigma(x) = \frac{1}{1+e^{-x}}$) to approximate the indicator. Based on this trick, our problem can be reformulated in terms of maximizing the following objective:

$$\begin{aligned} \max_V \mathcal{S}(V) &= \max_V \sum_{i \in I} \sum_{k \in S_i} \sum_{j \notin S_i} \ln \sigma(sim(i,k) - sim(i,j)) \\ &= \max_V \sum_{i \in I} \sum_{k \in S_i} \sum_{j \notin S_i} \ln \sigma(V_i^T (V_k - V_j)). \end{aligned} \tag{2}$$

TABLE I
AMAZON DATASET STATISTICS.

Dataset	#users	#items	#ratings	density	#also viewed	density	#bought together	density
Baby	531,891	248,275	916,769	6.9×10^{-6}	2,251,812	3.6×10^{-5}	50,148	8.1×10^{-7}
Beauty	1,210,281	651,519	2,026,943	2.6×10^{-6}	6,211,990	1.4×10^{-5}	160,539	3.8×10^{-7}
Electronics	1,404,138	749,459	2,059,917	1.9×10^{-6}	4,953,648	8.8×10^{-6}	196,588	3.5×10^{-7}
Home and Kitchen	2,511,624	1,143,992	4,260,181	1.5×10^{-6}	9,631,709	7.3×10^{-6}	208,972	1.6×10^{-7}
Movies and TV	749,894	229,783	1,447,706	8.4×10^{-6}	338,884	6.4×10^{-6}	132,060	2.5×10^{-6}
Toys and Games	1,342,924	885,839	2,254,507	1.9×10^{-6}	9,234,636	1.2×10^{-5}	259,244	3.3×10^{-7}
Tools and Home Impr.	1,212,486	752,447	1,928,202	2.1×10^{-6}	3,521,588	6.2×10^{-6}	156,585	2.7×10^{-7}

TABLE II
NOTATION.

Symbol	Discription
M	number of users
N	number of items
K	number of latent factors
I	the set of all items
$U \in \mathbb{R}^{K\times M}$	user latent factor matrix in LF models
$V', V \in \mathbb{R}^{K\times N}$	item latent factor matrices
$R \in \mathbb{R}^{M\times N}$	observed rating matrix
S_i	the set of items linked with item i in S
C_i	the set of items linked with item i in C

Here $sim(i,k)$ is modeled based on matrix factorization $V_i^T V_k$, where $V_i, V_k \in V$ is the latent vector representation of item i and k, and $V_i, V_k \in R^{K\times 1}$.

B. Modeling Complement Networks

Complement networks are built upon users' co-purchasing behavior. In contrast to substitute networks, complementarity suggests an inherently more complex relationship than mere similarity among items. As mentioned before, a t-shirt and a matching pair of jeans, or a cellphone and a charger, might be regarded as being complementary, even if they are not 'similar'. The properties of complementary items reflect co-occurrence relationships such that the occurrence (purchase) of one item's property might trigger the co-occurrence (co-purchase) of other items linked to it in the complement network.

Therefore, to model the co-occurrence of properties of these items in complement networks, we use an embedding learning method to learn the representation of items based on their complements. Specifically, we build the complement network as a directed graph C, since certain relationships are highly asymmetric (e.g. users often buy phones without buying a charger separately, but rarely buy a charger unless they have bought a phone). Given an item i, let C_i denote the set of items that are frequently bought together with i. Motivated by the success of *Skip-Gram* (SG) models used in NLP, where word representations are learned in terms of the representations of other words appearing in the same context, we adopt a similar idea to learn items' latent vector representations from their complement relationships. The basic idea of using SG models is to learn an item i's latent representation such that it is effective at predicting the occurrence of other items in C_i. Formally, given an item i, and a complement set C_i, the objective of our SG model here is to maximize the average

Input: Observed rating matrix R, a substitute network S, a complement network C, hyperparameters $h, \alpha, \beta, \lambda, \eta$.
Output: User latent matrix U and item latent matrix V
Initialize U and V (uniformly at random);
for *each item* i **do**
  Construct S_i and C_i;
end
Training:
for *iterations* **do**
  for *each item* i **do**
    Uniformly sample a user $p : R_{pi} \neq 0$;
    $U_p = U_p + \eta \frac{\partial \mathcal{R}(U,V)}{\partial U_p}$;
    $V_i = V_i + \eta \frac{\partial \mathcal{R}(U,V)}{\partial V_i}$;
    Uniformly sample an item $k \in S_i$ and an item $j \notin S_i$;
    $V_i = V_i + \alpha\eta \frac{\partial \mathcal{S}(V)}{\partial V_i}$;
    $V_k = V_k + \alpha\eta \frac{\partial \mathcal{S}(V)}{\partial V_k}$;
    $V_j = V_j + \alpha\eta \frac{\partial \mathcal{S}(V)}{\partial V_j}$;
    for *each item* $c \in C_i$ **do**
      Sample $k \in C_i, k \neq c$;
      $V_c = V_c + \beta\eta \frac{\partial \mathcal{C}(V,V')}{\partial V_c}$;
      ${V'}_k = {V'}_k + \beta\eta \frac{\partial \mathcal{C}(V,V')}{\partial {V'}_k}$;
      Sample h negative items$\{k' \sim P_{nc_c}\}$;
      for *each* k' **do**
        $V_c = V_c + \beta\eta \frac{\partial \mathcal{C}(V,V')}{\partial V_c}$;
        ${V'}_{k'} = {V'}_{k'} + \beta\eta \frac{\partial \mathcal{C}(V,V')}{\partial {V'}_{k'}}$;
      end
    end
  end
end

Algorithm 1: Model Learning.

log probability as follows:

$$\max_{V} \frac{1}{|C_i|} \sum_{j \in C_i} \sum_{k \in C_i, k \neq j} \ln p(k|j). \tag{3}$$

Following [20], we use a softmax function to define the probabilities $p(k|j)$:

$$p(k|j) = \frac{\exp({V'}_k^T V_j)}{\sum_{k' \in I} \exp({V'}_{k'}^T V_j)}. \tag{4}$$

By using a negative sampling technique for efficient learning and summarization of all items, we can obtain the objective function to be maximized,

$$\begin{aligned} \max_{V,V'} \mathcal{C}(V, V') = \max_{V,V'} \sum_{i \in I} \big(\sum_{j \in C_i} \sum_{k \in C_i, k \neq j} \big(\ln \sigma({V'}_k^T V_j) \\ + \sum_{k'} \mathbb{E}_{k' \sim P_{nc_i}} \ln \sigma(-{V'}_{k'}^T V_j) \big) \big), \end{aligned} \tag{5}$$

where h is the number of negative samples, P_{nc_i} denotes the distribution of items not in C_i, $V_j \in V$ is the latent representation of item j, and $\sigma(\cdot)$ is the sigmoid function. Here we adopt the unigram distribution raised to the power $\frac{3}{4}$ to construct P_{nc_i}, following the setting in [21].

C. A Unified Framework for Rating Estimation

Based on our models of substitute and complement networks, now we propose a unified framework (RSC) that leverages the substitute network (*S*) and complement network (*C*) to improve rating estimation (*R*) accuracy. Explicit user feedback on items comes in the form of star ratings. Here we use Matrix Factorization (MF) as a basic model for rating estimation. In MF, users and items are represented by K-dimensional latent-factor vectors, $U \in \mathbb{R}^{K \times M}$ and $V \in \mathbb{R}^{K \times N}$. The goal is to learn the latent vectors U and V such that the rating prediction error is minimized, i.e.,

$$\max_{U,V} \mathcal{R}(U, V) = \max_{U,V} - \sum_{R_{p,i} \neq 0} (R_{p,i} - \widehat{R}_{p,i})^2 - \lambda \|U\|_F^2 - \lambda \|V\|_F^2, \tag{6}$$

where $\widehat{R}_{p,i} = U_p^T V_i$ is the estimated rating, capturing the 'compatibility' between users and items in terms of the inner product of their corresponding latent representations. $\|U\|_F^2$ and $\|V\|_F^2$ are regularization terms used to avoid overfitting (and λ is a tradeoff). As explained in [26], Eq (6) has the probabilistic interpretation that the observed ratings are $U_p^T V_i$ subject to Gaussian noise.

By aligning the latent representations of items (V_i) between rating estimation and the modeling of substitute and complement networks, we obtain a unified objective function to be maximized:

$$\begin{aligned} \max_{U,V,V'} \ & \mathcal{L}(U, V, V') \\ & = \mathcal{R}(U, V) + \alpha \mathcal{S}(V) + \beta \mathcal{C}(V, V') \\ & = - \sum_{R_{p,i} \neq 0} (R_{p,i} - \widehat{R}_{p,i})^2 - \lambda \|U\|_F^2 - \lambda \|V\|_F^2 \\ & + \alpha \sum_{i \in I} \sum_{k \in S_i} \sum_{j \notin S_i} \ln \sigma(V_i^T (V_k - V_j)) \\ & + \beta \sum_{i \in I} \big(\sum_{j \in C_i} \sum_{k \in C_i, k \neq j} \big(\ln \sigma({V'}_k^T V_j) \\ & + \sum_{k'} \mathbb{E}_{k' \sim P_{nc_i}} \ln \sigma(-{V'}_{k'}^T V_j) \big) \big). \end{aligned} \tag{7}$$

Since rating estimation is our main goal, we introduce two hyperparameters α and β to control the tradeoff between predicting the ratings accurately and modeling the networks. Our algorithm for optimizing this objective function is shown in Algorithm 1. We use an alternating iterative update procedure and employ stochastic gradient descent to optimize the objective function. Specifically, for each training instance sampled in SGD, we calculate the derivative and update the corresponding parameters in $\Theta = \{U, V, V'\}$ by walking along the ascending gradient direction,

$$\Theta^{t+1} = \Theta^t + \eta \times \frac{\partial \mathcal{L}(\Theta)}{\partial \Theta}. \tag{8}$$

D. Parallelizability and Scalability Analysis

As real recommender systems need to deal with millions of users and items, it is important for the learning algorithm to scale well with large-scale datasets. A natural solution is to optimize its parameters based on multi-threading and parallelization. To parallelize the learning of our proposed method, we fist split items into several disjoint subsets so that each thread can simultaneously process one subset by optimizing the corresponding items' latent factor vectors and the users' latent vectors based on Eq. 7 and 8. In particular, we employ a lock-free asynchronous version of stochastic gradient descent (ASGD) to update the parameters of Eq. 7. As shown in [23], a lock-free approach to parallelize stochastic gradient descent can achieve a nearly optimal rate of convergence when the optimization problem is sparse. Taking the rating density (only around 10^{-6}) shown in Table I into consideration, we can safely update the parameters using lock-free ASGD based on multi-threading. We report scalability experiments in the following section and the results show that there is no loss of predictive performance when we increase the number of threads to speed up the training process. We will share all code and data used in this paper at publication time.

V. Experiments

In this section, we conduct experiments on Amazon datasets to evaluate the effectiveness of the proposed methods.

TABLE III
EXPERIMENTAL COMPARISONS (MAE) ON *Amazon* DATA.

Dataset	K=10						K=20					
	UserMean	ItemMean	PMF	SVD++	TrustSVD	RSC	UserMean	ItemMean	PMF	SVD++	TrustSVD	RSC
Baby	1.0079	0.9281	0.9939	0.9902	0.9713	**0.8807**	1.0079	0.9281	0.9905	0.9834	0.9693	**0.8922**
Beauty	0.9889	0.9739	1.0763	1.0092	0.8942	**0.8173**	0.9889	0.9739	1.0678	0.9920	0.9879	**0.9673**
Electronics	1.0817	1.0246	0.9432	0.9110	0.8940	**0.8832**	1.0817	1.0246	0.9483	0.9164	1.0548	**0.9061**
Home and Kitchen	1.0093	0.9626	1.1006	0.9711	0.9541	**0.9123**	1.0093	0.9626	0.9433	0.9362	0.9127	**0.8446**
Movies and TV	0.8550	0.8362	0.9156	0.8760	0.8064	**0.7872**	0.8550	0.8362	0.8842	0.8773	0.8111	**0.7384**
Toys and Games	0.9439	0.9243	1.0411	0.9444	1.0535	**0.9084**	0.9439	0.9243	1.0339	0.9445	0.8964	**0.8948**
Tools and Home Impr.	0.9811	0.9275	1.0601	0.9818	0.9503	**0.8886**	0.9811	0.9275	1.0512	0.9792	0.9526	**0.8829**

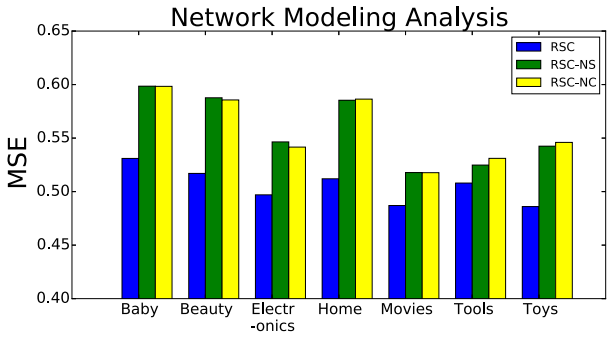


Fig. 1. Network modeling analysis. This figure shows the impact of removing substitute (NS) and complement (NC) networks from the model.

A. Experimental Setup

In order to demonstrate the performance of our approach, we use seven datasets and several metrics to evaluate all compared algorithms. α, β, λ and η are empirically set to 0.01, 0.01, 0.005 and 0.001 respectively. The number of negative samples is set to 5. Our experiments are intended to address the following questions:

1) How does our approach compare with related rating estimation methods for rating prediction?
2) How do substitute and complement networks contribute to rating estimation accuracy?
3) Can the proposed methods be applied to large-scale datasets?

Evaluation Metrics. In our experiments, for each user, we randomly select 80% of their observed ratings as a training set and leave the remainder for testing. We evaluate the prediction quality of all methods in terms of the metrics, Mean Absolute Error (MAE) and the Mean Squared Error (MSE).

Comparison methods. In order to demonstrate the benefits of our approach, we compare our model with the following methods for rating prediction, most of which can be found in LibRec[2] and MyMediaLite [3].

- **UserMean:** This method performs rating prediction based on the mean rating value of the user.
- **ItemMean:** This method performs rating prediction based on the mean rating value of the item.
- **PMF:** This method [26] assumes users' and items' latent vectors are generated from a gaussian distribution and performs matrix factorization based on the rating matrix.
- **SVD++:** This method [10] is a state-of-the-art rating estimation method amongst approaches that operate on raw rating data.
- **TrustSVD:** This method [6] exploits social relations between users to improve rating estimation performance. Here we employ this method by interchanging users and items in order to apply the method to an item (rather than a user) network. This is a state-of-the-art rating estimation method with relational knowledge.
- **RSC:** This is our proposed method that combines rating estimation with items' substitute and complement networks. The learning algorithm is shown in Algorithm 1.

[2]http://www.librec.net/

B. Performance Comparison

Tables III detail the experimental results when the number of latent factors are set as 10 and 20. From the results, we can see that the proposed framework significantly outperforms all the baseline methods at predicting unknown ratings. Specifically, we find that due to the extreme sparsity of observed ratings, PMF, which models users' and items' latent factors via a Gaussian assumption, cannot accurately solve the rating estimation problem considered here. Instead, BiasedMF and SVD++ show better performance than PMF by making use of user and item biases. Moreover, we also notice that by using network information among items, TrustSVD can perform better than other model- or memory-based methods, which also verifies the contribution of incorporating relational information into recommender systems. Compared with TrustSVD, our proposed RSC framework shows impressive results by incorporating two models which are specifically hand-crafted for modeling substitute and complement relationships among items. This confirms the necessity of considering a fine-grained recommendation framework to differentiate between multiple types of item relationships.

Network Modeling Analysis To demonstrate the contribution of modeling substitute and complement networks, we conduct ablation experiments by removing one of the two networks from the model and evaluating the deterioration in the model's

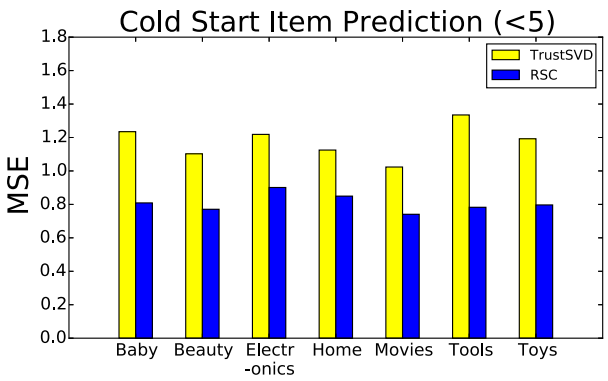
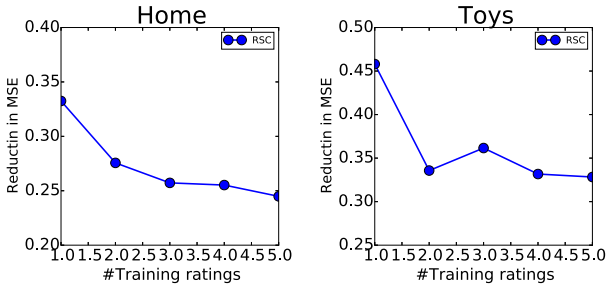
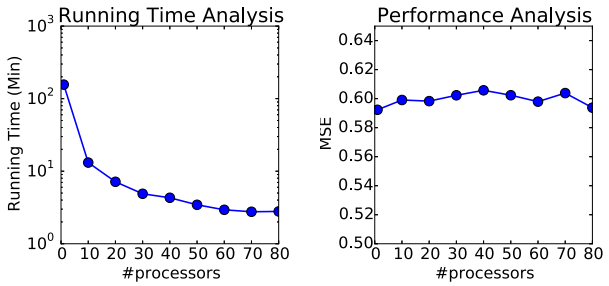


Fig. 2. Cold-start rating prediction.



(a) Home (b) Toys

Fig. 3. Improvement analysis on cold-start rating prediction. The Y-axis represents the MSE improvement of RSC compared with TrustSVD, as a function of the number of ratings available at training time.



(a) Running Time Analysis (b) Rating Prediction Accuracy

Fig. 4. Scalability analysis on the 'Baby' dataset.

predictive performance. Here we initialize both α and β as 0.01, and remove one of the two networks by setting its corresponding parameter to 0. Figure 1 shows the results on all datasets. *RSC-NS* represents the framework without substitute network modeling and *RSC-NC* represents the framework without complement network modeling. We can see that only by modeling both networks in concert can the model achieve satisfactory performance. Moreover, we find that network modeling has a different impact in each of the different data collections.

Cold Start Recommendation Analysis In this subsection, we perform experiments to demonstrate the recommendation performance of our proposed framework for cold start items. We consider items with fewer than 5 ratings as cold-start items (similar to [9, 36]). Figure 2 demonstrates the results. We compare *RSC*'s performance with TrustSVD and find that on all datasets, the proposed *RSC* framework outperforms TrustSVD in rating prediction for cold start items. Figure 3 further shows the improvement of *RSC* compared with TrustSVD on our two largest datasets when different amounts of training ratings are available for particular items. Notice that *RSC* achieves significant improvements when training ratings are particularly scarce. These results suggest that modeling both substitutable and complementary relationships significantly improves cold-start recommendation.

Scalability Analysis To verify the scalability of our proposed method, we implemented our method using a multi threaded architecture and demonstrate that it is scalable to large, real-world datasets. Figure 4 shows the scalability of the proposed method on the 'Baby' dataset. Here the speedup in model learning scales with the number of workers, from 1 to 80. We see that the proposed framework takes only about 4 minutes to complete training. Moreover, we see that there is no loss of predictive performance by using ASGD for model learning.

VI. Conclusion

In this paper, we studied the problem of improving recommendation accuracy via modeling networks of substitutable and complementary products. In particular, we proposed a rating estimation framework, *RSC*, by exploiting a pairwise ranking method for substitute network modeling and an unsupervised representation learning method for complement network modeling. Experiments on several real-world datasets showed that the proposed framework significantly improves the accuracy of rating estimation and can be efficiently applied to large-scale datasets by parallelization.

For future work, we are interested in investigating the efficacy of the proposed model when recurrent neural networks, such as long short term memory (LSTM) models, are used to model co-browsing and co-purchasing sequences.

References

[1] N. Djuric, H. Wu, V. Radosavljevic, M. Grbovic, and N. Bhamidipati. Hierarchical neural language models for joint representation of streaming documents and their content. *In WWW*, 2015.

[2] H. Fang, Y. Bao, and J. Zhang. Leveraging decomposed trust in probabilistic matrix factorization for effective recommendation. *In AAAI*, 2014.

[3] Z. Gantner, S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. MyMediaLite: A free recommender system library. *In RecSys.*, 2011.

[4] M. Grbovic, N. Djuric, V. Radosavljevic, F. Silvestri, and N. Bhamidipati. Context- and content-aware embeddings for query rewriting in sponsored search. *In SIGIR*, 2015.

[5] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, and D. Sharp. E-commerce in your inbox: Cross-domain product recommendations at scale. *In KDD*, 2015.

[6] G. Guo, J. Zhang, and N. YorkeSmith. Trustsvd: Collaborative filtering with both the explicit and implicit influence of user trust and of item ratings. *In AAAI*, 2015.

[7] G.-N. Hu, X.-Y. Dai, Y. Song, S.-J. Huang, and J.-J. Chen. A synthetic approach for recommendation: Combining ratings, social relations, and reviews. *In IJCAI*, 2015.

[8] L. Hu, A. Sun, and Y. Liu. Your neighbors affect your ratings: On geographical neighborhood influence to rating prediction. *In SIGIR*, 2014.

[9] M. Jamali and M. Ester. Trustwalker: a random walk model for combining trust-based and item-based recommendation. *In KDD*, 2009.

[10] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. *In KDD.*, 2008.

[11] M. A. Krishna and E. Charles. Link prediction via matrix factorization. *In ECML/PKDD*, 2011.

[12] Q. Le and T. Mikolov. Distributed representations of sentences and documents. *In ICML*, 2014.

[13] C. Li, Y. Lu, Q. Mei, D. Wang, and S. Pandey. Click-through prediction for advertising in twitter timeline. *In KDD*, 2015.

[14] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 2003.

[15] H. Ma, H. Yang, M. R.Lyu, and I. King. Sorec: Social recommendation using probabilistic matrix factorization. *In CIKM*, 2008.

[16] H. Ma, D. Zhou, C. Liu, M. R.Lyu, and I. King. Recommender systems with social regularization. *In WSDM*, 2011.

[17] J. J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. 2013.

[18] J. J. McAuley, R. Pandey, and J. Leskovec. Inferring networks of substitutable and complementary products. In *KDD*, 2015.

[19] J. J. McAuley, C. Targett, Q. Shi, and A. van den Hengel. Image-based recommendations on styles and substitutes. 2015.

[20] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *In ICLR.*, 2013.

[21] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *In NIPS.*, 2013.

[22] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. *In KDD*, 2014.

[23] B. Recht, C. Re, S. Wright, and F. Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. *In NIPS.*, 2011.

[24] S. Rendle, C. Freuden-thaler, Z. Gantner, and L. Schmidt-Thieme. Bpr:bayesian personalized ranking from implicit feedback. *In UAI*, 2009.

[25] R.Salakhutdinov and A.Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. *In ICML*, 2008.

[26] R.Salakhutdinov and A.Mnih. Probabilistic matrix factorization. *In NIPS*, 2008.

[27] Y. Shen and R. Jin. Learning personal + social latent factor model for social recommendation. *In SIGKDD*, 2012.

[28] F. Sun, J. Guo, Y. Lan, J. Xu, and X. Cheng. Learning word representations by jointly modeling syntagmatic and paradigmatic relations. *In ACL*, 2015.

[29] D. Tang, B. Qin, and T. Liu. Learning semantic representations of users and products for document level sentiment classification. *In ACL.*

[30] D. Tang, B. Qin, T. Liu, and Y. Yang. User modeling with neural network for review rating prediction. *In IJCAI*, 2015.

[31] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. Line: Large-scale information network embedding. *In WWW*, 2015.

[32] H. Wang, N. Wang, and D.-Y. Yeung. Collaborative deep learning for recommender systems. In KDD, 2015.

[33] L. Xu, K. Liu, and J. Zhao. Joint opinion relation detection using one-class deep neural network. *In COLING*, 2014.

[34] W. Yao, J. He, G. Huang, and Y. Zhang. Modeling dual role preferences for trust-aware recommendation. *In SIGIR*, 2014.

[35] M. Ye, X. Liu, and W.-C. Lee. Exploring social influence for recommendation: a generative model approach. *In SIGIR*, 2012.

[36] T. Zhao, J. J. McAuley, and I. King. Leveraging social connections to improve personalized ranking for collaborative filtering. In *In CIKM*, 2014.

[37] T. Zhao, V. Zhao, and I. King. Exploiting game theoretic analysis for link recommendation in social networks. *In CIKM*, 2015.