Mining Product Adopter Information from Online Reviews for Improving Product Recommendation

WAYNE XIN ZHAO, Renmin University of China
JINPENG WANG, Peking University
YULAN HE, Aston University
JI-RONG WEN, Renmin University of China
EDWARD Y. CHANG, HTC Research & Innovation
XIAOMING LI, Peking University

We present in this paper an automated framework which extracts product adopter information from online reviews and incorporates the extracted information into feature-based matrix factorization for more effective product recommendation. In specific, we propose a bootstrapping approach for the extraction of product adopters from review text and categorize them into a number of different demographic categories. The aggregated demographic information of many product adopters can be used to characterize both products and users in the form of distributions over different demographic categories. We further propose a graph-based method to iteratively update user- and product-related distributions more reliably in a heterogeneous user-product graph and incorporate them as features into the matrix factorization approach for product recommendation. Our experimental results on a large dataset crawled from JINGDONG, the largest B2C e-commerce website in China, show that our proposed framework outperforms a number of competitive baselines for product recommendation.

CCS Concepts: •**Information systems** → **Information systems applications;**

General Terms: Algorithms, Performance

Additional Key Words and Phrases: Online review, product adopter, product recommendation, matrix factorisation

1. INTRODUCTION

With the rapid growth of online e-commerce services, online reviews have become an important information resource to extract users' feedback and opinions towards products and services they purchased [Pang and Lee 2008]. Many studies have been devoted to uncover hidden knowledge from online reviews to help improving e-commerce services, including opinion summary generation [Hu and Liu 2004], product sale prediction [Liu et al. 2007] and product recommendation [Korfiatis and Poulos 2013]. However, an important type of information in online reviews has been seldom considered, i.e., product adopters. Consider the following review sentence about "iPhone 6":

> "I bought my son an iPhone 6, and he was very glad with the new phone."

Authors' addresses: W.X. Zhao (corresponding author) and J.-R. Wen, School of Information & Beijing Key Laboratory of Big Data Management and Analysis Methods, Renmin University of China; J. Wang and X. Li, Computer Science Department, Peking University; Y. He, Aston University; E.Y. Chang, HTC Research & Innovation.

DOI: 0000001.0000001

Here, we call the phrase "my son" as the *product adopter mention*. In this example, product purchaser and adopter are not the same person. Although the demographic information of the product purchaser is unknown, the explicit mention of the product adopter in the review tells us that the actual user of "iPhone 6" is most probably a young male. Mining product adopter mentions from online reviews of a particular product could thus enable the inference of the target audience of the product, which in turn, allows better product recommendation. For example, if another user expressed a purchase intent, "I want to buy a smart phone for my son. Any suggestions?" In such a case, "iPhone 6" might be a good candidate for recommendation.

Based on a large data set consisting of 139 million reviews crawled from the largest Chinese B2C e-commerce Website JINGDONG[1], we have found that more than 10% reviews (or 15 million reviews) contain at least one product adopter mention. This shows that it is indeed not uncommon that users bought products for others. Moreover, the prevalence of product adopter information available in online reviews makes it feasible to extract implicit demographic information of the target audience of products, which can be further leveraged for a better product recommendation. To the best of our knowledge, dealing with the mismatch of product purchasers and adopters for automatic inference of demographic information from text and subsequently incorporate it for product recommendation has been seldom studied before.

There are three major challenges we need to tackle in order to leverage product adopter information for recommendation. First, how to reliably extract product adopter mentions from noisy review text at a large-scale? Second, how to map product adopter mentions into a demographic feature space to form both product and user demographics? Third, how to effectively incorporate both user and product demographic information into a recommendation model?

To address these challenges, we develop an unsupervised bootstrapping method to automatically derive patterns to extract product adopter mentions, which are grouped into six categories with the idea of demographic segmentation in marketing research. Product demographic is then represented by a distribution over six such categories, which is called *product adopter distribution*. Similarly, each user is represented by its purchase preference patterns (whom she bought a product for) over the same six categories, called *user preference distribution*. To learn these two types of distributions, we construct a heterogeneous user-product graph and propose a graph-based method to iteratively update both user- and product-related distributions more reliably. We then further develop a matrix factorization approach to incorporate both user and product demographic information for product recommendation. Our experimental results on nearly $100K$ users and $60K$ products show that our proposed framework outperforms a number of competitive baselines consistently.

Our contributions can be summarised as follows:

— We propose a simple yet practical bootstrapping method to extract product adopter mentions from online reviews. Our analysis shows that over 15 million reviews contain at least one product adopter mention, which indicates the feasibility of learning implicit demographic information from online reviews for product recommendation.
— We construct six categories over the product adopter mentions with the idea of demographic segmentation. We formalise demographic characteristics with product adopter distributions and user preference distributions over the adopter categories. Furthermore, we propose a graph-based method to learn both types of distributions from a heterogeneous user-product graph.

[1] http://www.jd.com/

— We integrate the learnt product adopter distributions and user preference distributions into the matrix factorization approach for product recommendation. Experimental results on a large evaluation dataset show the feasibility and effectiveness of our proposed method.

2. DATA PREPARATION

We used a large e-commerce dataset used in [Wang et al. 2015], which contains 138.9 million transaction records from 12 million users on 0.2 million products. Each transaction record consists of a user ID, a product ID and the purchase timestamp. Most importantly, each transaction record is associated with a product review published by the user. Analysing and leveraging these review data will be our focus in this paper. We first group transaction records by user IDs and then obtain a list of purchased products for each user. This allows us to build a test set to evaluate our product recommendation method which will become clear in the experiments section. The statistics of our dataset are shown in Table I. Although our research was conducted on the dataset constructed from JINGDONG, the methods proposed here are equally applicable to the reviews collected from any other websites. We use the toolkit Jieba[2] to segment Chinese character streams into words.

Table I: Statistics of our entire JINGDONG dataset.

#reviews	#users	#products
138,905,740	12,127,267	246,447

3. EXTRACTION AND CATEGORIZATION OF PRODUCT ADOPTER MENTIONS

We first introduce two terminologies used in the paper.

Definition 3.1 (*Product adopter*). Given a product, the *product adopter* refers to the person who has actually used the product.

Definition 3.2 (*Product adopter mention*). The *product adopter mention* is a phrase in review text which describes the product adopter.

For example, in the following sentence:

> "I bought my son an iPhone 6, and he was very glad with the new phone."

the son of the purchaser (the review writer) was the actual *product adopter*, and "my son" is a *product adopter mention*. We will also use *adopter mention* or *mention* interchangeably with *product adopter mention* in the rest of the paper.

3.1. A Bootstrapping Extraction Method

We notice that some product adopter mentions could be described by the same linguistic pattern. For example, in a sentence "I bought my son this phone", the phrase "my son" is the adopter of the phone. If we can learn the pattern "buy somebody something", then we can extract the corresponding product adopter mention. We propose a bootstrapping approach in Algorithm 1 to iteratively learn the patterns and extract adopter mentions [Wang et al. 2015].

The approach starts with some seed patterns such as "buy somebody something" and "a gift to somebody". In each iteration, existing patterns are used to extract adopter

[2]https://github.com/fxsjy/jieba

mention phrases using the function ExtractAdopterMentionPhrases$(\cdot,\cdot)$, and new patterns are learned from the extracted phrases using the functions GeneratePatterns$(\cdot,\cdot)$ and ExtractTopFrequentPatterns$(\cdot)$. Specifically, for each extracted adopter mention phrase, the GeneratePatterns$(\cdot,\cdot)$ function combines the proceeding n_1 tokens and the following n_2 tokens as a candidate pattern. Moreover, since using single-token patterns extracts many false positive adopter mentions and using patterns with more than two tokens yields little improvement when applied on short review text as in our experiments, only two-token patterns are considered here, i.e., $n_1 + n_2 = 2$, s.t. $0 \leq n_1 \leq 2$ and $0 \leq n_2 \leq 2$. A pattern filtering step (Lines $14 - 17$) is used to further reduce spurious patterns that may lead to a large number of incorrect product adopter mentions. This is based on the speculation that the adopter mention phrases identified by a good pattern should not deviate from the previously identified phrases too much. Here the Jaccard coefficient is used to measure the similarity among the extracted phrases and the threshold δ is empirically set to 0.3. Finally, 45 frequent patterns are derived for the extraction of product adopter mentions.

ALGORITHM 1: Bootstrapping algorithm for extracting product adopter mentions from online reviews.

Input: review sentence corpus $\mathcal{R}$, seed extraction patterns $\mathcal{P}^{(seed)}$
Output: an set of learned extraction patterns $\mathcal{P}$ and a set of extracted adopter mentions $\mathcal{R}$;
$\mathcal{P} \leftarrow \mathcal{P}^{(seed)}$, $\mathcal{P}' \leftarrow \mathcal{P}^{(seed)}$;
$\mathcal{R}' \leftarrow \emptyset$, $\mathcal{R} \leftarrow \emptyset$;
repeat
  $\mathcal{R}' \leftarrow \emptyset$;
  for *each pattern* $p \in \mathcal{P}'$ **do**
    $\mathcal{R}_p \leftarrow \emptyset$;
    for *each sentence* $s \in \mathcal{R}$ **do**
      if *p exists in s* **then**
        $\mathcal{R}_p \leftarrow \mathcal{R}_p \cup$ ExtractAdopterMentionPhrases (p,s);
      end
    end
    if *Jaccard*($\mathcal{R}_p$, $\mathcal{R}$) $\leq \delta$ *and* $p \notin \mathcal{P}^{(seed)}$ **then**
      $\mathcal{R}_p \leftarrow \emptyset$;
      Remove p from $\mathcal{P}'$;
    end
    $\mathcal{R}' \leftarrow \mathcal{R}_p \cup \mathcal{R}'$;
  end
  $\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{P}'$, $\mathcal{R} \leftarrow \mathcal{R} \cup \mathcal{R}'$;
  $\mathcal{R}' \leftarrow \emptyset$, $\mathcal{P}' \leftarrow \emptyset$;
  for *each sentence* $s \in \mathcal{R}$ **do**
    for *each demographic phrase* $m \in \mathcal{R}'$ **do**
      $\mathcal{P}' \leftarrow \mathcal{P}' \cup$ GeneratePatterns(s, m);
    end
  end
  $\mathcal{P}' \leftarrow$ ExtractTopFrequentPatterns$(\mathcal{P}')$;
until *No new pattern is identified*;
return *An set of learned extraction patterns* $\mathcal{P}$ *and a set of extracted adopter mentions* $\mathcal{R}$*;*

As it is infeasible to check all the extracted mentions, we only consider those occurred more than 30 times and are left with 410 mentions. A total of 363 mentions are judged to be correct, which gives a precision of 88.5%.[3] We have conducted error analysis on the extraction results and found the following most common error types:

[3]We do not report the recall performance of the extraction algorithm here, since it is difficult to identify all the possible adopter mentions. With some efforts of manual checking, we have found that the top frequent adopter mentions have covered the majority of the reviews that contain adopter mentions.

— The product adopter is not a human being. For example, "I bought my dog this biscuit".
— The extraction pattern identified an adopter mention errorneously. For example, using an extraction pattern "to+nouns" would mistakenly take "JingDong" as an adopter mention from the review "Give five star to JingDong".

Using our proposed bootstrapping approach for the extraction of product adapter mentions, we found that 10.8% of a total of 130 million reviews (equivalent to 15 million reviews) contain at least one adopter mention. Since it is impossible to enumerate all the possible linguistic patterns describing adopter mentions, this value can be considered as a lower bound of the proportion of review documents containing product adopter mentions. We have found that more than 48.5% products (receiving the reviews) and 25.4% users (posting the reviews) have at least 10 reviews containing adopter mentions. The above results show that the product adopter information indeed prevails in review data.

Our extraction task is closely related to information extraction [Lafferty et al. 2001; Chang et al. 2006], in which many existing methods can be explored for the extraction of product adopter mentions. However, most of them are supervised learning methods requiring a considerable amount of labeled data. As future work, we could consider a semi-supervised learning model which can be trained from "pseudo" labeled data derived from our pattern matching method.

3.2. Categorization of Adopter Mentions

In online reviews, the same entity can be referred to in many different ways. For example, "mum, mom, mother" all refer to the same entity "mother". In addition, some different adopter mentions may share similar demographic features. For example, "grandpa, father-in-law" are all males and possibly over age 55. As such, it makes sense to group product adopter mentions into different categories where each category shares similar demographic information. To do this, we first remove ambiguous mentions such as "others" and "people" and only keep the mentions with a relatively clear demographic profile. Following the idea in *market segmentation*[4], we mainly consider two types of demographic characteristics, age and sex. We have invited two senior e-commerce officers in charge of market promotion from JINGDONG to give us advices on categorizing product adopter mentions. At the end, we have identified five major categories relating to the mention of relatives based on age and sex: *Children*, *Young female*, *Old female*, *Young male* and *Old male*. Apart from these five categories relating to relatives, we have further identified a category relating to *Colleagues*. Although it is difficult to identify clear demographic features from this category, it is observed that some office products such as printers and fax machines are more closely related to *Colleagues* rather than other categories. As such, the *Colleagues* category is also taken into account and we have a total of six user categories. Table II shows some example adopter mentions and their corresponding categories. The full categorization information of product adopter mentions can be found in http://goo.gl/avne1y.

4. PRODUCT AND USER CHARACTERISTICS LEARNING

With the aforementioned six categories of product adopters and let $\mathcal{C}$ be the set of categories of adopter mentions, we can characterize products and users in the following ways:

[4]http://en.wikipedia.org/wiki/Market_segmentation
#Demographic_Segmentation

Table II: Example product adopter mentions and their corresponding categories.

Category	Example product adopter mentions
Children	小孩(kids) 她小孩(her kid) 新生儿(new-horn baby) 婴儿(baby) 小侄女(little niece) 小外甥(little nephew)
Young female	妹妹(younger sister) 表妹(cousin) 孕妇(pregnant woman) 女朋友(girlfriend) 媳妇(daughter-in-law) 姐姐(elder sister)
Old female	妈妈(mother) 老妈(mother) 阿姨(aunt) 姑姑(aunt) 岳母(mother-in-law) 丈母娘(mother-in-law)
Young male	弟弟(younger brother) 哥哥(elder brother) 表弟(cousin) 哥们(buddy) 兄弟(brothers) 男朋友(boyfriend)
Old male	爸爸(father) 老爸(father) 叔叔(uncle) 爷爷(grandfather) 岳父(father-in-law) 公公(father-in-law) 舅舅(uncle)
Colleagues	公司(company) 办公室(office) 员工(employee) 分公司(branch) 厂里(factory) 工人(worker)

Definition 4.1 (*Product adopter distribution*). Given a product p, the *product adopter distribution* is a vector $\mathbf{f}_p$ with six elements where each element $f_{p,c}$ is a proportion of the adopters in category $c \in \mathcal{C}$ for product e. This distribution essentially characterizes the demographics of product p by the users who have actually used the product.

Definition 4.2 (*User preference distribution*). Similar to the definition of product adopter distribution, *user preference distribution* characterizes a user's purchase preference pattern (whom she bought a product for) over the six adopter categories. Given a user u, the *user preference distribution* is also a six-element vector $\mathbf{f}_u$ with each of its element $f_{u,c}$ denotes the probability of user u who tends to buy products for people in category c.

Product adopter distribution and *user preference distribution* characterize the preference over the six adopter categories from the perspective of products and users respectively. *Product adopter distribution* captures product demographics which are represented by a collection of the characteristics of the people who have adopted that product; while *user preference distribution* measures the preference level of a user over people in different adopter categories, that is, whom the user is more likely to buy products for.

To estimate these two kinds of distributions, a straightforward method is to apply the maximum likelihood (ML) method to infer the distribution values based on the number of explicit adopter mentions in the reviews. We can have

$$\begin{aligned} f_{p,c}^{(ML)} &= \frac{\#N(p,c)+\gamma}{\#N(p,\cdot)+|\mathcal{C}|\gamma}, \\ f_{u,c}^{(ML)} &= \frac{\#N(u,c)+\gamma}{\#N(u,\cdot)+|\mathcal{C}|\gamma}, \end{aligned} \tag{1}$$

where $\#N(p,c)$ is the number of reviews about product p which contain adopter mentions in category c, and $\#N(u,c)$ is the number of reviews written by u which contain adopter mentions in category c. We use Laplace smoothing to avoid the zero probability which is caused by data sparsity.

4.1. A Graph-Based Method for Estimating Distributions of Users and Products

The aforementioned ML method for the estimation of both *product adopter distribution* and *user preference distribution* relies on the explicit mention of product adopters in online review. As has been previously discussed, only 10.8% out of a total of 130 million reviews contain product adopter mentions. Therefore, for users or products whose associated online reviews seldom contain product adopter mentions, their respective distributions cannot be reliably estimated using ML. Another important issue is that the above ML method can only model the purchase preference for others but not the user herself, since only explicit adopter mentions are used for deriving these estimations. To address these problems, we build a heterogeneous user-product graph and propose a graph-based method to iteratively update user- or product-related distributions, in which three types of relation links are considered and more reliable estimations can be obtained.

Definition 4.3 (*User-product graph*). A *user-product graph* $\mathcal{G}$ consists of a vertex set $\mathcal{V}$ and an edge set $\mathcal{E}$. Given a set of users $\mathcal{U}$ and products $\mathcal{P}$, $\mathcal{V}$ is defined to be a union set of users and products, i.e. $\mathcal{V} = \mathcal{U} \cup \mathcal{P}$. Thus, $\mathcal{E}$ contains three categories of edges, i.e., edges between user-product pairs $\mathcal{E}^{(UP)}$, user-user pairs $\mathcal{E}^{(UU)}$ and product-product paris $\mathcal{E}^{(PP)}$.

On the user-product graph, each vertex (either a user or a product) is attached with a distribution over the six adopter categories. Next, we describe how to learn $\mathbf{f}_u$ for each user $u \in \mathcal{U}$ and $\mathbf{f}_p$ for each product $p \in \mathcal{P}$.

Evidence propagation through the user-product edges. Given a user u and a product p purchased by u, if u has explicitly mention a product adopter in her review, then both user- and product-related distributions can be estimated by ML. If this is not the case, we build an undirected edge between u and p and propagate the preference evidence from user u to product p by gathering the preference distribution of user u on other products. Similarly, we collect the existing adopter information of p contributed by other users, and then propagate the distribution evidence from product p to user u. When a user or product receives the distribution evidence from its incoming links, we take the average of these values, which can be formally modelled as

$$\mathbf{f}_u^{(UP)} \leftarrow \frac{1}{|\mathcal{E}_u^{(UP)}|} \sum_{p \in \mathcal{E}_u^{(UP)}} \mathbf{f}_p^{(old)}, \tag{2}$$

$$\mathbf{f}_p^{(UP)} \leftarrow \frac{1}{|\mathcal{E}_p^{(UP)}|} \sum_{u \in \mathcal{E}_p^{(UP)}} \mathbf{f}_u^{(old)}, \tag{3}$$

where $\mathcal{E}_u^{(UP)}$ and $\mathcal{E}_p^{(UP)}$ denote the set of product vertices linking to user u and the set of user vertices linking to product p.

Evidence propagation through the user-user edges. We now consider the preference propagation through the user-user edges. Our intuition is that users who have similar purchase records should also have similar preference distributions over the adopters categories. We first represent each user as a vector over all the products, and each entry of the vector indicates whether the user has purchased a product or not. Then we apply the cosine similarity to find the top K nearest neighbors for each user based on the constructed user vectors. A directed edge is built from the target user to these K neighbors. We used $\mathcal{E}_u^{(UU)}$ to denote the set of the top K most similar user vertices for user u. Having built the links to the top-K neighbors, we propagate the preference evi-

dence from the neighbors to the target user and then take an average of these evidence scores

$$\mathbf{f}_u^{(UU)} \leftarrow \frac{1}{|\mathcal{E}_u^{(UU)}|} \sum_{u' \in \mathcal{E}_u^{(UU)}} \mathbf{f}_{u'}^{(old)}. \tag{4}$$

Evidence propagation through the product-product edges. Similar to user-user edges, our intuition is that products purchased by similar users should also receive similar adopter distributions. We first represent each product as a vector over all the users, and each entry of the vector indicates whether the product has been purchased by a user or not. Then we apply the cosine similarity to find the top K nearest neighbors for each product based on the constructed purchase vectors. A directed edge is built between the target product and these K neighbours. We used $\mathcal{E}_p^{(PP)}$ to denote the set of the top K most similar product vertices of product p. We propagate the distribution evidence from the neighbors to the target product and then take an average of these evidence scores

$$\mathbf{f}_p^{(PP)} \leftarrow \frac{1}{|\mathcal{E}_p^{(PP)}|} \sum_{p' \in \mathcal{N}_p^{(PP)}} \mathbf{f}_{p'}^{(old)}. \tag{5}$$

An iterative update formula for distribution estimation. Once we have learned $\mathbf{f}_u^{(UP)}$, $\mathbf{f}_p^{(UP)}$, $\mathbf{f}_u^{(UU)}$ and $\mathbf{f}_p^{(PP)}$, we can integrate them with the ML estimation (Eq. 1) into a unified iterative formula.

We update $\mathbf{f}_u$ and $\mathbf{f}_p$ alternatively as follows

$$\begin{aligned} \mathbf{f}_u^{(new)} &\leftarrow \lambda_u \mathbf{f}_u^{(ML)} + (1-\lambda_u)\big(\mu_u \mathbf{f}_u^{(UP)} + (1-\mu_u)\mathbf{f}_u^{(UU)}\big), \\ \mathbf{f}_p^{(new)} &\leftarrow \lambda_p \mathbf{f}_p^{(ML)} + (1-\lambda_p)\big(\mu_p \mathbf{f}_p^{(UP)} + (1-\mu_p)\mathbf{f}_p^{(PP)}\big), \end{aligned} \tag{6}$$

where λ_u, λ_p, μ_u and μ_p are the tuning parameters. In our implementation, we first fix $\mathbf{f}_u^{(ML)}$ and $\mathbf{f}_p^{(ML)}$, and then randomly initialize $\mathbf{f}_u^{(UP)}$, $\mathbf{f}_p^{(UP)}$, $\mathbf{f}_u^{(UU)}$ and $\mathbf{f}_p^{(PP)}$. In each iteration, we first update $\mathbf{f}_u^{(UP)}$, $\mathbf{f}_p^{(UP)}$, $\mathbf{f}_u^{(UU)}$ and $\mathbf{f}_p^{(PP)}$, and then learn $\mathbf{f}_u^{(new)}$ and $\mathbf{f}_p^{(new)}$. At the end of each iteration, we perform the following update operations: $\mathbf{f}_u^{(old)} \leftarrow \mathbf{f}_u^{(new)}$ and $\mathbf{f}_p^{(old)} \leftarrow \mathbf{f}_p^{(new)}$.

The number of the most similar neighbours, i.e., K, is empirically set to 30 in this paper. When the K-nearest neighbourhood has been built for users and products, the above iterative algorithm in Eq. 6 has $\mathcal{O}(nK|\mathcal{U}||\mathcal{C}| + n\bar{L}^U|\mathcal{U}||\mathcal{C}| + nK|\mathcal{P}||\mathcal{C}| + n\bar{L}^P|\mathcal{P}||\mathcal{C}|)$, where n is the iteration number, $\bar{L}^U$ is the average number of the purchased products for a user and $\bar{L}^P$ is the average number of the purchasers for a product. Typically, the values of $\bar{L}^P$ and $\bar{L}^U$ are relatively small, thus the algorithm can be very efficient in practice. In our experiments, 10 iterations are sufficient to generate good results.

Note that the learnt distributions over product adopter categories are very useful in e-commerce services. Since for any business organisation, knowing the people who will purchase its products or services (term as *product demographics*) is a key to success. The product adopter distributions can be directly used to help infer *product demographics*. Unlike the simple ML method (Eq. 1), the learnt user preference distribution can model the purchase preference of both "buy for others" and "buy for the user herself", since it takes into account the evidence from all the purchased products and "neighbouring users". In this paper, our focus is to exploit the learnt distributions for

improving product recommendation. Our graph-based estimation method essentially follows the main principle of the widely used label propagation algorithm [Zhu and Goldberg 2009], where it is assumed that vertices closely connected in a graph are similar to each other. Our contribution lies in the construction of the heterogeneous user-product graph for the propagation of distribution evidence.

4.2. Evaluation of the Proposed Method

We conduct experiments to evaluate the proposed method. The question is how to build a gold standard to measure the accuracy of the estimated user- or product-related distributions. Our hypothesis is that for popular products which received a large number of reviews containing product adopter mentions, their product adopter distributions estimated by ML should be fairly accurate and can be used as a gold standard. Similarly, for users who have often mentioned product adopters in their reviews, the ML estimation of their preference distributions can also be used as a gold standard.

We select the top 2,000 products and 1,000 users with the most adopter mentions and collect their associated reviews. For these selected products and users, the ML estimation of their respective product adopter distributions and user preference distributions with all their associated reviews is used as the gold standard. Then, we order the reviews from the selected users and products by their publishing timestamps and use the first $x\%$ as our training set. When x is small, i.e., there are very few training data, we would like to find out whether our proposed graph-based method can overcome data sparsity and generate the user- or product-related distributions matched closely to the actual distributions calculated by ML on the full dataset. The root mean squared error (RMSE) is used as the evaluation metrics

$$RMSE = \sqrt{\frac{1}{|\mathcal{T}|} \frac{\sum_{e \in \mathcal{T}} \sum_{c \in \mathcal{C}} \left(f_{e,c}^{(gold)} - \hat{f}_{e,c}\right)^2}{|\mathcal{C}|}}, \tag{7}$$

where e denotes either a product or a user, and $|\mathcal{T}|$ is the number of instances in our test collection.

We have four sets of parameters to set in our graph-based method, namely $\{\lambda_u\}$, $\{\lambda_p\}$, $\{\mu_u\}$ and $\{\mu_p\}$. We find that the performance is less sensitive to the value of μ. As such, we simply set μs for all the users and products to 0.5, which indicates an equal weight for user-user (or product-product) edges and user-product edges. For λs, an intuition is that when a user or product has more training data (reviews containing adopter mentions), its ML estimation should be more reliable and hence should have a larger weight. We propose to use a sigmoid-like function to adaptively set λ

$$\lambda(x) = \frac{c}{1 + e^{-ax}} + (1 - c), \tag{8}$$

where x denotes the number of adopter mentions, a and c are the scale parameters. The value of λ monotonically increases with the value x, and it falls in the interval $(0, 1]$. We use 10% data in the gold standard as a development set to tune the parameters with grid search, and the rest 90% data as the test set. The optimal values obtained are: $a = 0.05$ and $c = 1.45$. We present the results in Table III which are averaged over product adopter distributions and user preference distributions. It can be observed that our graph-based method significantly improves over the simple ML method. The improvement is more prominent when the size of the training data is small.

Table III: Performance comparison of RMSE results for the estimation of user- and product-related distributions. Smaller is better.

$x\%$	ML	Graph
10	0.097 ($\uparrow 6.9\%$)	0.090
5	0.145 ($\uparrow 13.6\%$)	0.125
1	0.311 ($\uparrow 18.3\%$)	0.254

5. INCORPORATING PRODUCT ADOPTER INFORMATION INTO MATRIX FACTORIZATION FOR PRODUCT RECOMMENDATION

In recommender systems, traditional matrix factorization approaches can only deal with the dyadic matrices, which is not able to make use of auxiliary information. To address this, several studies have been proposed to incorporate auxiliary information into matrix factorization [Koren et al. 2009; Chen et al. 2012; Rendle 2012]. In this section, we study how to model the adopter-related information within the commonly used matrix factorization framework. We first discuss the standard matrix factorization and then introduce how to leverage product adopter information for product recommendation based on matrix factorization.

Given a user u and a product p, let $r_{u,p}$ be a binary value indicating whether u has purchased product p. The basic idea of matrix factorization is to learn a low-dimension latent representation for both users and products in the same space, then reconstruct the original purchase record matrix. A typical formulation [Koren et al. 2009] can be given as follows

$$\hat{r}_{u,p} = \mu + b_u + b_p + \mathbf{x}_u^\top \mathbf{y}_p, \tag{9}$$

where μ is the global bias, b_u and b_p are the bias parameters reflecting the mean preference value of user u and product p respectively, $\mathbf{x}_u$ and $\mathbf{y}_p$ are the L-dimension latent vectors for user u and product p respectively. The above matrix factorization approach has been widely used in practice and shown to be robust in many tasks. In our work here, we have obtained three types of adopter-related information apart from users' purchase records: 1) the adopter mentions in the reviews; 2) the user preference distributions; and 3) product adopter distributions. Next, we study how to incorporate such information into the standard matrix factorization framework [Koren et al. 2009].

5.1. Modelling Latent User Representation

When a user would like to make a purchase, we consider two key aspects influencing her final choice on the products besides her own purchase preference.

— Adopter category: It summarizes the common characteristics of a group of adopters with similar demographics as shown in Table II. When a user would like to buy a product, she is likely to be influenced by such group-based demographics. For example, it is important to know what products are suitable for *old female* when selecting a gift for "grandmother".
— Adopter (mention): The adopter mention directly impacts on the product selection. Instead of focusing on a single purchase, we can capture collective preference patterns when a large number of consumers buy products for a specific adopter (mention). For example, if *GNC Women's Ultra Mega Bone Density*[5] is highly selected as gifts to grandmothers by a large number of consumers, then it can be considered as a good recommendation when buying a product for "grandmother".

[5]http://www.gnc.com/GNC-Womens-Ultra-Mega-Bone-Density/product.jsp?productId=4033460

These two factors reflect multi-grain useful characteristics from the adopter information. Adopter categories can be used to characterize the common demographic patterns given a specific group of adopters, while adopter mentions further increase the discriminative power within the adopter category. Next we give detailed formulations for latent user representation for these two aspects.

Characterizing adopter categories: For an adopter category c in $\mathcal{C}$, we model it by using L-dimension latent vectors $\mathbf{x}_c$ and in total we can obtain $|\mathcal{C}|$ such latent vectors shared by all the users. So the problem becomes how a user selects over these categories. Recall that we have estimated the user preference distribution over the adopter categories in Section 4, which can be used as weights of these latent factors for a user. Formally, the categorical preference can be modelled as $\sum_{c=1}^{|\mathcal{C}|} f_{u,c}\mathbf{x}_c$ for user u.

Characterizing adopters: For an adopter mention m in $\mathcal{M}$, we model it by using L-dimension latent vectors $\mathbf{x}_m$, and in total we can obtain $|\mathcal{M}|$ such latent vectors shared by all the users. We can use a vector of binary indicators to use these adopter-specific latent vectors. In a purchase, only the latent vector(s) corresponding to the final adopter(s) will be active.

With the above two kinds of latent factors, the new user representation can be formulated as below

$$\tilde{\mathbf{x}}_u = \mathbf{x}_u + \sum_{c=1}^{|\mathcal{C}|} f_{u,c}\mathbf{x}_c + \sum_{m\in\mathcal{M}} \mathbb{I}[m]\mathbf{x}_m, \tag{10}$$

where $\mathbf{x}_u$ encodes the user-specific purchase characteristics as that in Eq. 9, and $\mathbb{I}[m]$ is indicator function only returning 1 when the adopter mention m is active (or specified) in a purchase. As we can see, $\mathbf{x}_u$, $\{\mathbf{x}_c\}$ and $\{\mathbf{x}_m\}$ contribute to the final purchase interests for users.

5.2. Modelling Latent Product Representation

As a major extension, our product representation considers the incorporation of product demographics.

— Product demographic: The product demographic[6], sometimes called the target audience, of a product or service is a collection of the characteristics of the people who buy that product or service. Knowing information such as the income status, age and tastes of the demographic can help better route the product recommendation to the more suitable users.

Since we set up six adopter categories in Table II, they give a categorisation of the adopter mentions based on demographics. For an adopter category c in $\mathcal{C}$, we model it by using L-dimension latent vectors $\mathbf{y}_c$ and in total we can obtain $|\mathcal{C}|$ latent vectors shared by all the products. To characterize the product demographics, we model it as a linear combination of $\{\mathbf{y}_c\}$, thus we have $\sum_{c=1}^{|\mathcal{C}|} f_{p,c}\mathbf{y}_c$ for product p, where the coefficients are set to the probability values in product adopter distribution.

Formally, we can have the new product representation as follows

$$\tilde{\mathbf{y}}_p = \mathbf{y}_p + \sum_{c=1}^{|\mathcal{C}|} f_{p,c}\mathbf{y}_c. \tag{11}$$

[6]http://www.ehow.com/info_10015346_product-demographic.html

5.3. The Recommendation Formula

With the above latent representations for users and products, we are ready to give the final recommendation formula. For a user-product pair (u, p), we have the following factorization formula

$$\begin{aligned} & \hat{r}_{u,p} \\ = \ & bias + \tilde{\mathbf{x}}_u^\top \tilde{\mathbf{y}}_p \\ = \ & \mu + b_{u,p} + b_u + b_p + \left(\mathbf{x}_u + \sum_{c=1}^{|\mathcal{C}|} f_{u,c}\mathbf{x}_c + \sum_{m\in\mathcal{M}} \mathbb{I}[u,p,m]\mathbf{x}_m\right)^\top \left(\mathbf{y}_p + \sum_{c=1}^{|\mathcal{C}|} f_{p,c}\mathbf{y}_c\right), \end{aligned} \tag{12}$$

where $b_{u,p}$, b_u and b_p are the bias parameters for user-product pair (u, p), user u and product p respectively. We incorporate a dyadic bias parameter $b_{u,p}$ to denote the match degree between a user and a product in terms of the distribution over the adopter categories. The idea is that a user is more likely to buy a product with a more similar adopter-category distribution, therefore the corresponding entry should receive a larger global bias value, which can be set as follows

$$b_{u,p} = \text{sim}(\mathbf{f}_u, \mathbf{f}_p),$$

here we use the cosine similarity to implement the function $\text{sim}(\cdot, \cdot)$. We use Θ to denote the parameters to learn, $\{\mu, b_{(\cdot)}, \mathbf{x}_{(\cdot)}, \mathbf{y}_{(\cdot)}\}$.

Detailed model analysis. Based on the matrix factorization approach, the incorporation of $\{\mathbf{x}_c\}$ and $\{\mathbf{y}_c\}$ enables the capability to capture the effect of adopter categories on recommendation. Two different sets of latent factors (i.e., $\{\mathbf{x}_c\}$ and $\{\mathbf{y}_c\}$) to model adopter categories makes it more flexible to characterize different demographic patterns in both sides of users and products. Furthermore, the corresponding sets of coefficients $\{\mathbf{f}_u\}$ and $\{\mathbf{f}_p\}$ give the user- and product-specific preference over these categories. In Eq. 12, we can obtain a term $f_{u,c}f_{p,c}\mathbf{x}_c^\top \mathbf{y}_c$. Intuitively, with the other terms fixed, $\mathbf{x}_c$ would be forced to be similar to $\mathbf{y}_c$ if $f_{u,c} \times f_{p,c}$ is large, which indicates that both user u and product p are strongly associated with category c. The use of $\{\mathbf{x}_m\}$ gives more discriminative power to select over products for different adopter mentions in the same category.

Parameter learning. To learn the parameters, we adopt the pairwise ranking model. Given a user u, we generate the positive-negative pairs of products (p, p') in which u has purchased p (called *positive*) but not p' (called *negative*). The pairwise ranking model assumes that the fitted value for the purchased product is larger than the one that has not been purchased by a user, i.e., $Pr(\hat{r}_{u,p} > \hat{r}_{u,p'})$. Furthermore, we adopt the sigmoid function as the loss function

$$Pr(\hat{r}_{u,p} > \hat{r}_{u,p'}) = \frac{1}{1 + e^{-(\hat{r}_{u,p} - \hat{r}_{u,p'})}}.$$

Note that for pairwise ranking, we do not need to learn the user bias parameters $\{b_u\}$. With the above partial-order rank probability function, the overall regularized ranking loss function can be written as follows

$$\mathcal{L} \ = \ -\sum_{u\in\mathcal{U}} \sum_{(p,p')\in\mathcal{D}_u} \log \frac{1}{1 + e^{-(\hat{r}_{u,p} - \hat{r}_{u,p'})}} + \sum_j \lambda_1 \parallel \mathbf{x}_j \parallel_2^2 + \sum_j \lambda_2 \parallel \mathbf{y}_j \parallel_2^2 + \lambda_3 \sum_p \parallel b_p \parallel_2^2,$$

where $\mathcal{D}_u$ denotes the positive-negative pairs for user u, and λs are the coefficients for ridge regularization. By minizing the loss function $\mathcal{L}$, we use the stochastic gradient descent method (SGD) to learn the model parameters. Given a training instance consisting of a user u and a positive-negative pair (p, p'), the derivatives at this instance for updating the model parameters are presented as follows

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \mathbf{x}_u} &= -e^u_{p>p'}\left\{\Delta \mathbf{y}_{p,p'} + \sum_{c\in\mathcal{C}} \mathbf{y}_c \Delta f_{p,p',c}\right\} + 2\lambda_1 \mathbf{x}_u,\\
\frac{\partial \mathcal{L}}{\partial \mathbf{x}_c} &= -f_{u,c} e^u_{p>p'}\left\{\Delta \mathbf{y}_{p,p'} + \sum_{c\in\mathcal{C}} \mathbf{y}_c \Delta f_{p,p',c}\right\} + 2\lambda_1 \mathbf{x}_c,\\
\frac{\partial \mathcal{L}}{\partial \mathbf{x}_m} &= -e^u_{p>p'}\left(\mathbb{I}[u,p,m]\bar{\mathbf{y}}^p - \mathbb{I}[u,p',m]\bar{\mathbf{y}}^{p'}\right) + 2\lambda_1 \mathbf{x}_m,\\
\frac{\partial \mathcal{L}}{\partial \mathbf{y}_p} &= -e^u_{p>p'}\bar{\mathbf{x}}^{u,p} + 2\lambda_2 \mathbf{y}_p,\\
\frac{\partial \mathcal{L}}{\partial \mathbf{y}_{p'}} &= e^u_{p>p'}\bar{\mathbf{x}}^{u,p'} + 2\lambda_2 \mathbf{y}_{p'},\\
\frac{\partial \mathcal{L}}{\partial \mathbf{y}_c} &= -e^u_{p>p'}\left(f_{p,c}\bar{\mathbf{x}}^{u,p} - f_{p',c}\bar{\mathbf{x}}^{u,p'}\right) + 2\lambda_2 \mathbf{y}_c,\\
\frac{\partial \mathcal{L}}{\partial b_p} &= -e^u_{p>p'}\left(\beta^{(p)}_j - \beta^{(p')}_j\right) + 2\lambda_3 b^{(P)}_j,
\end{aligned}$$

where $\Delta \mathbf{y}_{p,p'} = \mathbf{y}_p - \mathbf{y}_{p'}$, $\Delta f_{p,p',c} = f_{p,c} - f_{p',c}$, $e^u_{p>p'} = 1 - Pr(\hat{r}_{u,p} > \hat{r}_{u,p'})$, $\bar{\mathbf{x}}^{u,p} = \mathbf{x}_u + \sum_{c=1}^{|\mathcal{C}|} f_{u,c}\mathbf{x}_c + \sum_{m\in\mathcal{M}} \mathbb{I}[u,p,m]\mathbf{x}_m$ and $\bar{\mathbf{y}}^p = \mathbf{y}_p + \sum_{c=1}^{|\mathcal{C}|} f_{p,c}\mathbf{y}_c$.

The SGD method to train our model has the computational complexity of $\mathcal{O}(nL\bar{F}|\mathcal{D}|)$, where n is the iteration number, L is the number of latent factors, $\bar{F}$ is the average number of non-zero features for a training instance and $|\mathcal{D}|$ is the training data size. In practice, we have found SGD has very fast convergence speed, and usually $30 - 50$ iterations over our training set are sufficient for convergence.

5.4. Applications in Product Recommendation

With the learnt models, we consider two application scenarios as follows:

Overall product recommendation: for each user, we generate a list of candidate products that she is likely to purchase. The candidate products are ranked with the fitted values according to Eq. 12.

Adopter-oriented product recommendation: it aims to return a list of products to the product purchaser for a specific adopter. For example, if a mother wants to buy a new phone for her son, she publish a short message on a microblogging platform:

> "I would like to buy my son a new smart phone. Any suggestions?"

An efficient product recommendation system should target the actual user of the smart phone, in this case, "my son" instead of the writer of the message. This is what we called *adopter-oriented product recommendation*. To the best of our knowledge, there is little work focusing on adopter-oriented product recommendation. We rank products as follows

$$\hat{r}_{u,p} = bias + \Big(\mathbf{x}_u + f_{u,c_m}\mathbf{x}_{c_m} + \mathbf{x}_m\Big)^\top \Big(\mathbf{y}_p + f_{p,c_m}\mathbf{y}_{c_m}\Big), \tag{13}$$

where c_m is the adopter category for the mention m and *bias* is used to incorporate all the bias values for simplicity. Besides the user's own preference, we also consider the adopter mention and the corresponding adopter category. These three types of information are modelled as latent factors in the same space with the matrix factorization approach.

Since the mention latent vectors $\{\mathbf{x}_m\}_{m\in\mathcal{M}}$ and the user's adopter-category latent vectors $\{\mathbf{x}_c\}_{c\in\mathcal{C}}$ are shared by all the users, they can be leveraged to alleviate the "cold start" problem where new users come to the system without purchase records. Formally, a purchase need can be modelled as a pair of an mention and the corresponding category (m, c_m). We rank products as follows

$$\hat{r}_{u,p} = bias + \Big(\mathbf{x}_{c_m} + \mathbf{x}_m\Big)^\top \Big(\mathbf{y}_p + f_{p,c_m}\mathbf{y}_{c_m}\Big). \tag{14}$$

The major difference between Eq. 13 and 14 is that there is no user latent vector in Eq. 14, thus it does not rely on the purchase record of a user, which naturally alleviate the cold-start problem. In order to have a better understanding of the proposed approach, we give an illustrative example for the cold-start recommendation case in which a purchase intent is expressed as "`I would like to buy my son a new smart phone`". In this case, our ranking formula in Eq. 14 can be detailed as follows:

$$\hat{r}_{u,p} = bias + \Big(\mathbf{x}_{\text{“son”}} + \mathbf{x}_{\text{“young-male”}}\Big)^\top \Big(\mathbf{y}_p + f_{p,\text{“young-male”}} \cdot \mathbf{y}_{\text{“young-male”}}\Big),$$

where $\mathbf{x}_{\text{“son”}}$ and $\mathbf{x}_{\text{“young-male”}}$ are the latent vectors corresponding to the adopter mention of "son" and the adopter category of "Young male". These adopter-related information is absent in traditional matrix factorization for recommendation. On the contrary, we leverage such information to capture the preference over the adopters.

6. EXPERIMENTS

We present the evaluation results on both overall product recommendation and adopter-oriented product recommendation.

6.1. Overall Product Recommendation

Overall product recommendation aims to return a list of recommended products for a user.

Construction of the Evaluation Set. As shown in Table I, we have collected a total of 130 million transaction records from 12 million users on 0.2 million products. We first group them by userID and obtain a list of purchased products for each user. We remove the users who have purchased fewer than 20 products and products with fewer than 10 purchases. After the above filtering step, we build a user-product matrix and set an entry to one if the product is purchased by the corresponding user. We further iteratively remove the users and products with at least ten non-zero entries. [7] Finally, we obtained a collection of 99,997 users and 57,243 products with a total number of

[7]Since we would split purchase records by users into training and test parts, it would be more reliable to evaluate the comparison methods using the users with relatively enough purchase information.

5,379,724 purchase records. Its sparsity degree is about 99.91%. In order to simulate the real scenario of online product recommendation, we split the data collection into training and test datasets by timestamps. For each user, we take the first $\delta\%$ of her purchase records as the training data, and the remaining $(100-\delta)\%$ as the test data. To examine the performance with varying amount of training data, we set δ to 50, 33 and 25, which correspond to the $\frac{\#training}{\#test}$ ratios of 1/1, 1/2 and 1/4 respectively. For fairness of comparison, we do not filter users and products with very few reviews containing product adopter mentions.

Methods to Compare. We consider the following methods for performance comparisons:

— **Matrix Factorization (MF)**: the standard MF method as in [Koren et al. 2009]. We use the square loss error function to learn the model parameters.
— **Matrix factorization with Bayesian personalized ranking (BPR-MF)**: it applies the Bayesian personalised ranking to learn the models with pairwise ranking for matrix factorization [Rendle et al. 2009].
— **AMF**$_U$: our proposed Adopter-based Matrix Factorization (AMF) approach with only user preference distributions incorporated as features.
— **AMF**$_{U+P}$: our proposed AMF approach with both user preference distributions and product adopter distribution used as features.
— **AMF**$_{all}$: our proposed AMF approach with user preference distributions, product adopter distribution and adopter mentions incorporated as features.

We set the regularization coefficient to a small value, i.e., 0.004, the iteration number to 50 and the factor number to 20 for all the methods. In our data sets, we treat the un-purchased products by a user as the negative items. To generate the training set, for each user, we sample the negative products with a ratio of 1:1, i.e., we have the same number of negative and positive products. To generate the test set, we randomly sample the negative products with a ratio of 1:50, i.e., each positive product would be paired with 50 negative products[8]. All the negative products come from the corresponding category of the positive product. In the test set, for each user, we shuffle her list of candidate products. The evaluation task aims to examine whether a method is able to rank positive products over negative products. For our proposed methods, we use the proposed graph-based estimation method ($a = 0.05, c = 1.45$ in Eq. 8 and $\mu = 0.5$ in Eq. 6) to estimate both the user preference distributions and product adopter distributions based on the training data.

Evaluation Metrics. We adopt three widely used metrics for the evaluation of product recommendation results. The first metrics is $Precision@k$, which is a ratio of the purchased products in the top k recommendations. We do not consider $Recall@k$ since for the current task, it would be positively correlated with $Precision@k$. Apart from $Precision@k$ which reflects the performance of top recommendations, we also use the Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR), which reflect the performance of overall ranking and the first rank respectively. For the evaluation of model efficiency, we use *running time* (RT) as a metric. We developed all the methods using the open source toolkit SVDFeature[9], in which an accelerated algorithm tailored

[8]Given a user, we also tried using all the un-purchased products as the negative products. The finding is similar to what have been obtained in the following experiments. However, the computational cost for evaluation is extremely high when $|\mathcal{U}|$ and $|\mathcal{P}|$ are very large, roughly as $|\mathcal{U}| \times \big(|\mathcal{P}| \times L + |\mathcal{P}| \times \log|\mathcal{P}|\big)$. Thus, we adopt the negative sampling method for evaluation, which is commonly used in previous studies on item ranking evaluation [Rendle and Freudenthaler 2014; Yin et al. 2013].
[9]http://svdfeature.apexlab.org/wiki/Main_Page

to matrix factorization has already been implemented. We run all the experiments on a single machine with 64G memory, 16 AMD Opteron 8380 CPU cores at 2.5GHZ.

Results and Analysis. We present the results of different methods for overall product recommendation in Table IV. Three $\frac{\#training}{\#test}$ ratios are considered here: 1:1, 1:2 and 1:4. We have the following observations.

— BPR-MF performs better than the simple baseline MF. The major difference between MF and BPR-MF lies in the model learning method. MF uses a regression form to train the parameters with the squared loss functions, while BPR-MF consider the partial order between item pairs and uses a Bayesian learning method. As shown in [Rendle et al. 2009], BPR-MF is indeed a competitive method in the recommender task and serves as a strong baseline in our experiments.
— Our proposed AMF methods outperform both MF and BPR-MF with the best results obtained using AMF_{all}. Interestingly, the improvement of AMF_{all} over BPR-MR becomes more significant with less training data. It indicates that the adopter-related features are effective to improve product recommendation.
— The performance order of our proposed methods is as follows: $\text{AMF}_{all(U+P+M)} > \text{AMF}_{U+P} > \text{AMF}_{U}$. Recall we have three types of adopter-related features, namely user preference distributions, product adopter distributions and adopter mentions. The above observation shows that all these three types of features are essential for the recommendation task.
— For matrix factorization algorithms, training takes up significantly more time compared to testing. Thus, we only report the training time here. It can be observed that simpler methods take less training time. MF is most efficient in terms of traing, while AMF_{all} requires the longest training time. Nevertheless, our method is still very efficient, which can be trained in less than three minutes on all the three different training sets.

6.2. Adopter-Oriented Product Recommendation

Construction of the Evaluation Set. We use the same evaluation dataset in Section 6.1 to learn different methods for adopter-oriented product recommendation. The only difference is the way we construct the test set. The idea is that some users have already included adopter mentions in their product reviews, each of which in fact corresponds to a triplet (user, product, adopter). We treat each triplet as an individual case instead of grouping them by users. In total, we have obtained 246,032 cases. We still consider the evaluation sets with three $\frac{\#training}{\#test}$ ratios: 1/1, 1/2 and 1/4.

Given a triplet, the purchased product is treated as the only positive product. To generate the negative products, we first search the product category tree provided by JINGDONG and identify the leaf-node category (i.e., the most specific category). For example, a path over the category tree for "`smart phone`" could be "root $\rightarrow$ electronic product $\rightarrow$ communication tool $\rightarrow$ smart phone". Thus, "`smart phone`" is identified as the leaf-node category. Then we take all products except the purchased product in the identified category as negative products. Some categories may have a large number of products. In this case, we only keep at most the top 600 negative products ordered by their sales volume. Finally, for each triplet, the only positive product and the negative products form a list of candidate products. We further shuffle the list.

Methods to Compare. We still take MF and BPR-MF as baselines. Since these two methods are not designed for adopter-oriented product recommendation, the training and testing procedures follow the same way as in Section 6.1. We also consider another strong baseline for comparison.

Table IV: Performance comparisons of different methods on overall product recommendation. $***$ indicates that the improvement of of AMF$_{all}$ over BPR-MF is significant at the confidence levels of 0.001. "RT" indicates the training time.

$\frac{\#training}{\#test}$	Methods	P@10	MAP	MRR	RT (in sec.)
1:1	MF	0.101	0.086	0.237	38
	BPR-MF	0.154 ($\uparrow 30.9\%$)	0.113 ($\uparrow 34.0\%$)	0.319 ($\uparrow 13.7\%$)	64
	AMF$_U$	0.171	0.130	0.312	104
	AMF$_{U+P}$	0.186	0.140	0.317	147
	AMF$_{all}$	**0.202**$***$	**0.151**$***$	**0.363**$***$	161
1:2	MF	0.092	0.081	0.233	37
	BPR-MF	0.127 ($\uparrow 41.9\%$)	0.083 ($\uparrow 38.6\%$)	0.288 ($\uparrow 16.6\%$)	55
	AMF$_U$	0.145	0.096	0.280	80
	AMF$_{U+P}$	0.155	0.102	0.297	111
	AMF$_{all}$	**0.179**$***$	**0.116**$***$	**0.336**$***$	114
1:4	MF	0.081	0.052	0.201	35
	BPR-MF	0.092 ($\uparrow 64.6\%$)	0.062 ($\uparrow 37.6\%$)	0.213 ($\uparrow 40.0\%$)	45
	AMF$_U$	0.115	0.071	0.244	61
	AMF$_{U+P}$	0.123	0.074	0.254	78
	AMF$_{all}$	**0.151**$***$	**0.086**$***$	**0.298**$***$	78

PIR-MF: it is first proposed by Rendle and Schmidt-Thieme in the general framework of factorization machine [Rendle and Schmidt-Thieme 2010]. It is used to model the triplets (user, item, tag), whereas in our setting, an mention can be treated as a tag. We modify it to fit our purpose in which "tag" can be absent.

We use AMF$_{all}$ here and follow a similar training procedure as for overall product recommendation. In the ranking stage, we use the ranking formula in Eq. 13. We also consider another variant in Eq. 14 which is proposed specifically to tackle the "cold-start" problem, and in this case we denote our method as AMF_{cold}.

Results and Analysis. We use $Recall@k$ and *Mean Reciprocal Rank* (MRR) as evaluation metrics. We present the results of different methods for product recommendation in Table V. It can be observed that the Bayesian personalized ranking BPR-MF performs better than the simple baseline MF. However, PIR-MF is much better than both BPR-MF and MF. One main reason is that neither BPR-MF nor MF takes the adopter information into consideration, while PIR-MF makes use of the mention information for product recommendation. Our proposed AMF$_{all}$ outperforms both PIR-MF and BPR-MF substantially, especially compared with BPR-MF. PIR-MF is a very strong baseline in our current task, however, its performance largely decreases when the $\frac{\#train}{\#test}$ ratio decreases, i.e., with less training data (from 0.267 to 0.232 in terms of $R@10$). While our method is relatively stable and yields 17.6% improvement (with an absolute increase gain of 0.042)[10] over PIR-MF in terms of $R@10$ when $\frac{\#train}{\#test} = 1/4$.

We also notice that the variant of our methods AMF$_{cold}$ performs very competitively. It makes use of the mention latent vector and the categorical latent vectors.

[10]Note that we compute $R@10(R@10 = \frac{\#hit@10}{\#test\_cases})$ by averaging over all the test cases, an absolute increase gain of 0.042 indicates that our method can find the correct product for 4.2% more test cases in top ten ranks, which is indeed significant when the number of test cases is large as in real e-commerce Website.

Table V: Performance comparisons of different methods on adopter-oriented product recommendation. $**$ indicates that the improvement of AMF$_{all}$ over PIR-MF is significant at the confidence level of 0.01.

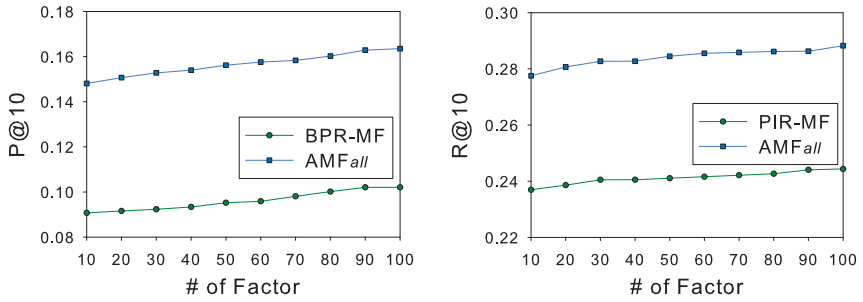
$\frac{\#training}{\#test}$	Methods	MRR	R@1	R@10
1:1	MF	0.098	0.028	0.229
	BPR-MF	0.109	0.040	0.232
	PIR-MF	0.122 ($\uparrow 7.4\%$)	0.046 ($\uparrow 6.1\%$)	0.267 ($\uparrow 10.9\%$)
	AMF$_{cold}$	0.122	0.045	0.265
	AMF$_{all}$	**0.131****	**0.049****	**0.296****
1:2	MF	0.100	0.027	0.2338
	BPR-MF	0.105	0.037	0.232
	PIR-MF	0.114 ($\uparrow 10.0\%$)	0.040 ($\uparrow 10.3\%$)	0.254 ($\uparrow 13.5\%$)
	AMF$_{cold}$	0.117	0.041	0.260
	AMF$_{all}$	**0.126****	**0.044****	**0.288****
1:4	MF	0.097	0.028	0.222
	BPR-MF	0.098	0.030	0.227
	PIR-MF	0.106 ($\uparrow 11.1\%$)	0.036 ($\uparrow 7.1\%$)	0.239 ($\uparrow 17.6\%$)
	AMF$_{cold}$	0.109	0.037	0.257
	AMF$_{all}$	**0.117****	**0.038****	**0.281****

This shows that AMF$_{cold}$ is effective when faced with the "cold-start" problem. The above results confirm that it is important to take into account adopter-related information for adopter-oriented recommendation, which can not be handled by traditional recommendation methods.

6.3. Parameter Analysis

For matrix factorization methods, an important parameter to set is the number of latent factors. In this subsection, we aim to find out how this parameter affects the performance of our method AMF$_{all}$. We use BPR-MF as a comparison. We vary the number of latent factors from 10 to 100 with a gap of 10. The results with $\frac{\#training}{\#test} = 1/4$ are shown in Fig. 1. For both overall recommendation and adopter-oriented recommendation, it can be observed that the performance improves with the increased number of latent factors, and AMF$_{all}$ performs consistently better than BPR-MF. Since the performance gain is not significant with the number of latent factors larger than 20 for both models, using 20 latent factors would be sufficient for our recommendation tasks considering the trade-off between performance and computational complexity.

In previous experiments, we use the proposed graph-based methods (Section 4) for learning user preference distributions and product adopter distributions. When $\lambda = 0$ and $\mu = 0$, the graph-based method reduces to the simple ML method. We compare the performance of AMF$_{all}$ with the distributions estimated by graph-based and the ML method in Table VI. We can see that using graph-based estimation gives better results compared to using ML for both recommendation tasks. It further verifies that the proposed graph-based method is more effective than ML for learning the distributions over the adopter categories.



(a) Overall recommendation ($Precision@10$) (b) Adopter-oriented recommendation ($Recall@10$)

Fig. 1: Results by varying the number of latent factors.

Table VI: Performance comparisons of AMF_{all} with the distributions estimated by the ML and the graph-based methods. $**$ indicates that the improvement is significant at the confidence levels of 0.01.

$\frac{\#training}{\#test}$	Methods	Overal recommendation		Adopter-oriented recommendation	
		MAP	P@10	MRR	R@10
1:1	AMF_{all}+ML	0.147	0.196	0.125	0.277
	AMF_{all}+Graph	**0.151****	**0.202****	**0.131****	**0.296****
1:2	AMF_{all}+ML	0.108	0.166	0.119	0.265
	AMF_{all}+Graph	**0.116****	**0.179****	**0.126****	**0.288****
1:4	AMF_{all}+ML	0.075	0.127	0.110	0.255
	AMF_{all}+Graph	**0.086****	**0.151****	**0.117****	**0.281****

6.4. Qualitative Analysis of Top Products for Adopter Categories and Mentions

Our model can produce the latent factors for adopter categories and mentions, namely $\{\mathbf{x}_c\}$ and $\{\mathbf{x}_m\}$. First, by using the cosine similarity (or dot product), we can obtain top products for adopter categories with the ranking criterion $\mathbf{x}_c^\top \tilde{\mathbf{y}}_p$. Since we have already observed that the incorporation of these latent factors is useful to the recommendation performance, we continue to give a qualitative analysis of how the top products are like and whether they are meaningful for these adopter categories.

The sample popular products by categories are given in Table VII. We can observe that these sample products have strong categorical characteristics. For example, the *Children* category is almost exclusively related to products made for baby and kids, and the *Colleagues* category mainly contains office supplies. As a comparison, shavers are most popular in *Old male* category, while a relatively diverse range of products are preferred in the *Young male* category; both the *Young female* and *Old male* categories like skin care products with the *Old male* clearly showing a preference on anti-aging products.

Given the above differences across different categories, we further analyse the variations for different mentions within an adopter category. It is observed that the mention-specific latent factor (i.e., $\mathbf{x}_m$) is able to further discriminate products related to a specific adopter category. For example, in Table VII, although *Children milk powder* and *Baby diaper* are top popular products for the *Children* category, the former is mainly purchased with the mentions of "son" or "daughter" while the later is almost

Table VII: Examples for top ranked products for each adopter category. Brand names are shown in brackets.

Category	Examples of top popular products
Children	[Huggies] Baby diaper, [Bobo] Baby training cup, [Pampers] Baby diaper, [Baby banana] Baby teething toothbrush, [Nestle] Children milk powder
Young female	[Seishido] Deep cleansing oil, [Olay] Eye transforming cream, [Ginkgo] Cleansing cream, [Levi's] Women short [Abbott] Milk Powder for Pregnant Women
Old female	[Nature's Bounty] Melatonin, [Xinle] Massage pillow, [Olay] Anti Aging Cream [Olay] Whitening cream, [Laorentou] Handbags
Young male	[NIVEA] Face wash, [Clear] Shampoo, [Gillette] Shavers [LAORENTOU] Message bag, [Swisswin] Backpack
Old male	[Gillette] Shavers, [Panasonic] Shavers, [Nokia] Phone, [Motorola] Phone, [ACCU-CHEK] Self-monitoring of blood glucose
Colleagues	[HP] Printers, [Sandi] Flash disks, [Toshiba] Mobile Hard disk [TP-LINK] Routers and Switches, [EPSON] Projector

always purchased with the mention of "baby". More interestingly, we have found that different products might indicate different closeness degrees of relationship between a buyer and the adopter. For example, in the *Young male* category, *Face wash*s are usually bought with the mention "my husband" while *Backpack*s are usually bought with the mention "my brother". The incorporation of mention-specific latent factors enhances the discriminative power in product recommendation.

With the above analysis, we can see an important merit of our approach which improves the explainability of the recommendation results within the traditional matrix factorization framework.

7. RELATED WORK

Our work is related to the following lines of studies.

Recommender systems. Early work on recommender systems typically uses collaborative filtering (CF) to make recommendations based on matching users with similar "tastes" or interests [Sarwar et al. 2001; Adomavicius and Tuzhilin 2005; Linden et al. 2003; Symeonidis et al. 2011]. In recent years, the matrix fatortization approach [Koren et al. 2009; Shi et al. 2012] has been widely applied and received much research interests. With the increasing volume of Web data, many studies focus on incorporating auxiliary information [Hong et al. 2013; Wang and Zhang 2013; Massa and Avesani 2007; Jamali and Ester 2009; Tang et al. 2012; Ma et al. 2011] into the matrix factorization approach. Two typical frameworks of such studies are the SVDFeature [Chen et al. 2012] and Factorization Machine [Rendle 2012]. Both are general models and do not focus on adopter-oriented recommendation.

Online review mining. With the rapid growth of online e-commerce services, online review mining has become a hot research topic [Pang and Lee 2008]. In particular, it has been shown that online review are useful to improve the results of product ranking or recommendation. Liu et al. ([2007]) proposed to use a sentiment model to predict sales performance; while in [McGlohon et al. 2010], composite rating scores were derived from aggregated reviews collected from multiple websites using different statistic- and heuristic-based methods and were subsequently used to rank products and merchants. Ganu et al. ([2013]) derived text-based ratings of item aspects from review text and then grouped similar users together based on the topics and sentiments that appear in the reviews. Zhang et al. ([2014a; 2014b]) extracted explicit product features (i.e. aspects) and user opinions by phrase-level sentiment analysis on user reviews for product recommendation.

Demographic-based recommendation. Demographic information has been important for recommender systems [Adomavicius and Tuzhilin 2005]. Typically, many existing studies utilize the demographic information obtained directly from user websites [Pazzani 1999; Giering 2008] or questionnaires [Qiu and Benbasat 2010]. Besides the users' registered profile, Seroussi et al. ([2011]) proposed to extract topics from user-generated text using the Latent Dirichlet Allocation (LDA) model, termed as text-based user attributes. Both types of attributes were then integrated into a matrix factorization model for rating prediction. Korfiatisa and Poulos ([2013]) proposed to build a demographic recommender system by extracting service quality indicators (star ratings) and consumer types from hotel reviews. They defined different demographic groups by consumer types based on the assumption that different types of travelers assess each quality indicator differently.

In this paper, we propose to infer users' demographic information from product adopter mentions in review text, which can be subsequently effectively utilized to achieve better recommendation performance. We do not require the demographic profile to be explicitly supplied by users. In fact, in many e-commerce websites such as JINGDONG, users are not required to fill in their demographic attributes which makes it impossible to obtain such information directly. Furthermore, we focus on the situation where a product buyer is different from a product adopter, i.e., a user bought a product for others. In this case, even with the availability of product buyers' registered demographic information, it is not useful for the recommendation task. We need to infer the demographic information of the actual product adopters and incorporate the adopter information into recommendation models to achieve better recommendation results. To the best of our knowledge, there is scarce work which deals with the mismatch between product buyers and adopters for product recommendation.

Our work is built on the previous study [Wang et al. 2015], where online review data was first used for product adopter extraction. We have made two major improvements: (1) We proposed a graph-based method from a heterogeneous user-product graph for learning product adopter distributions and user preference distributions over the adopter categories. Compared to the simple frequency based estimation [Wang et al. 2015], it can alleviate the data sparsity problem where a user or a product is associated with very few adopter mentions. Furthermore, by taking the evidence from all the purchased products and neighbouring users, our current method can yield meaningful estimations for the overall purchase preference "buy for others and the user herself"; while in [Wang et al. 2015], it only models the preference "buy for others". (2) Current work gives a more principle way to use the adopter-related information for product recommendation. Especially, we develop a approach which can make use of the information from the adopters and the corresponding adopter categories based on the matrix factorization framework.

8. CONCLUSION AND FUTURE WORK

In this paper, we take the initiative to mine product adopter information from reviews and use it for product recommendation in a large dataset. Our data analysis has revealed that more than 10% of the reviews contain at least one adopter mention, which shows the feasibility and importance to consider adopter-related information in review data. It also shows that it is not uncommon that a product buyer is different from a product adopter, i.e., a user bought a product for others. We first construct six adopter categories using the idea of demographic segmentation in marketing research, and then model the demographic characteristics with distributions over the adopter categories. We integrate the extracted adopter mentions and the learnt adopter-category distributions into the matrix factorization approach. Extensive experiments on a large evaluation data shows that our proposed methods are very effective.

Currently, we manually group adopter mentions into predefined categories for demographic segmentation. In the future, we will study how to automatically cluster the adopter mentions into meaningful categories. In the identification of the product adopters, we rely on an explicit mention of the actual adopter of a product in reviews. It will be interesting to explore an automated method which infers the actual adopter given a purchase transaction. Also, external events which might trigger large product sales are ignored in our current work. For example, we might expect to see a large number of the adopter mention of "mother" in product reviews near to Mother's Day. Such external knowledge could be potentially useful for more accurate extraction of product adopter mentions. Furthermore, we speculate that users would have different purchase preferences depending on whether they buy products for themselves or for others. If we can distinguish different roles each user plays during the purchase process, we could build a better recommendation model by taking user roles into account. Finally, we will apply our approach to other e-commerce websites, e.g., Amazon and Taobao, and investigate methods to improve the performance of adopter-oriented product recommendation, which we believe will be of great value for e-commerce services.

ACKNOWLEDGMENTS

The authors thank the anonymous reviewers for their valuable and constructive comments. The work was partially supported by National Natural Science Foundation of China under Grant No. 61502502, the National Key Basic Research Program (973 Program) of China under Grant No. 2014CB340403, the Open Fund of Beijing Key Laboratory on Integration and Analysis of Large-scale Stream Data, North China University of Technology, and Innovate UK under the grant number 101779.

REFERENCES

Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE TKDE* 17, 6 (June 2005).

Chia-Hui Chang, Mohammed Kayed, Moheb R. Girgis, and Khaled F. Shaalan. 2006. A Survey of Web Information Extraction Systems. *IEEE Trans. Knowl. Data Eng.* 18, 10 (2006), 1411–1428.

Tianqi Chen, Weinan Zhang, Qiuxia Lu, Kailong Chen, Zhao Zheng, and Yong Yu. 2012. SVDFeature: A Toolkit for Feature-based Collaborative Filtering. *Journal of Machine Learning Research* 13 (2012).

Gayatree Ganu, Yogesh Kakodkar, and AméLie Marian. 2013. Improving the Quality of Predictions Using Textual Information in Online User Reviews. *Inf. Syst.* 38, 1 (2013).

Michael Giering. 2008. Retail Sales Prediction and Item Recommendations Using Customer Demographics at Store Level. *SIGKDD Explor. Newsl.* 10, 2 (Dec. 2008).

Liangjie Hong, Aziz S. Doumith, and Brian D. Davison. 2013. Co-factorization Machines: Modeling User Interests and Predicting Individual Decisions in Twitter. In *WSDM*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *SIGKDD*.

Mohsen Jamali and Martin Ester. 2009. TrustWalker: a random walk model for combining trust-based and item-based recommendation. In *SIGKDD*.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37. DOI:http://dx.doi.org/10.1109/MC.2009.263

Nikolaos Korfiatis and Marios Poulos. 2013. Using online consumer reviews as a source for demographic recommendations: A case study using online travel reviews. *Expert Syst. Appl.* 40, 14 (2013).

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*. 282–289.

Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon.Com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing* 7, 1 (Jan. 2003).

Yang Liu, Jimmy Huang, Aijun An, and Xiaohui Yu. 2007. ARSA: A Sentiment-Aware Model for Predicting Sales Performance Using Blogs. In *SIGIR*.

Hao Ma, Tom Chao Zhou, Michael R. Lyu, and Irwin King. 2011. Improving Recommender Systems by Incorporating Social Contextual Information. *ACM Trans. Inf. Syst.* 29, 2 (2011).

Paolo Massa and Paolo Avesani. 2007. Trust-aware recommender systems. In *ACM RecSys*.

Mary McGlohon, Natalie S. Glance, and Zach Reiter. 2010. Star Quality: Aggregating Reviews to Rank Products and Merchants.. In *ICWSM*.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval* 2, 1-2 (2008), 1–135.

Michael J Pazzani. 1999. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review* 13, 5-6 (1999).

Lingyun Qiu and Izak Benbasat. 2010. A Study of Demographic Embodiments of Product Recommendation Agents in Electronic Commerce. *Int. J. Hum.-Comput. Stud.* 68, 10 (Oct. 2010), 669–688.

Steffen Rendle. 2012. Factorization Machines with libFM. *ACM Trans. Intell. Syst. Technol.* 3, 3 (May 2012).

Steffen Rendle and Christoph Freudenthaler. 2014. Improving pairwise learning for item recommendation from implicit feedback. In *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*. 273–282.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*.

Steffen Rendle and Lars Schmidt-Thieme. 2010. Pairwise Interaction Tensor Factorization for Personalized Tag Recommendation. In *WSDM*.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based Collaborative Filtering Recommendation Algorithms. In *WWW*.

Yanir Seroussi, Fabian Bohnert, and Ingrid Zukerman. 2011. Personalised Rating Prediction for New Users Using Latent Factor Models. In *ACM HH*.

Yue Shi, Xiaoxue Zhao, Jun Wang, Martha Larson, and Alan Hanjalic. 2012. Adaptive Diversification of Recommendation Results via Latent Factor Portfolio. In *SIGIR '12*.

Panagiotis Symeonidis, Eleftherios Tiakas, and Yannis Manolopoulos. 2011. Product Recommendation and Rating Prediction Based on Multi-modal Social Networks. In *ACM RecSys*.

Jiliang Tang, Huiji Gao, Huan Liu, and Atish Das Sarma. 2012. eTrust: Understanding Trust Evolution in an Online World. In *SIGKDD*.

Jian Wang and Yi Zhang. 2013. Opportunity Model for e-Commerce Recommendation: Right Product; Right Time. In *SIGIR*.

Jinpeng Wang, Wayne Xin Zhao, Yulan He, and Xiaoming Li. 2015. Leveraging Product Adopter Information from Online Reviews for Product Recommendation. In *ICWSM*.

Hongzhi Yin, Yizhou Sun, Bin Cui, Zhiting Hu, and Ling Chen. 2013. LCARS: A Location-content-aware Recommender System. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '13)*. 221–229.

Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014a. Explicit Factor Models for Explainable Recommendation Based on Phrase-level Sentiment Analysis. In *SIGIR*.

Yongfeng Zhang, Haochen Zhang, Min Zhang, Yiqun Liu, and Shaoping Ma. 2014b. Do Users Rate or Review?: Boost Phrase-level Sentiment Labeling with Review-level Sentiment Classification. In *SIGIR*.

Xiaojin Zhu and Andrew B. Goldberg. 2009. *Introduction to Semi-Supervised Learning*. Morgan & Claypool Publishers.