We Know What You Want to Buy: A Demographic-based System for Product Recommendation On Microblogs

Wayne Xin Zhao[1], Yanwei Guo[2], Yulan He[3], Han Jiang[2], Yuexin Wu[2] and Xiaoming Li[2]
[1]School of Information,Renmin University of China, China
[2]School of Electronic Engineering and Computer Science,Peking University, China
[3]School of Engineering & Applied Science, Aston University, UK
batmanfly@gmail.com, pkuguoyw@gmail.com, y.he@cantab.net,
jianghan08@gmail.com, wuyuexin@gmail.com, lxm@pku.edu.cn

ABSTRACT

Product recommender systems are often deployed by e-commerce websites to improve user experience and increase sales. However, recommendation is limited by the product information hosted in those e-commerce sites and is only triggered when users are performing e-commerce activities. In this paper, we develop a novel product recommender system called METIS, a MErchanT Intelligence recommender System, which detects users' purchase intents from their microblogs in near real-time and makes product recommendation based on matching the users' demographic information extracted from their public profiles with product demographics learned from microblogs and online reviews. METIS distinguishes itself from traditional product recommender systems in the following aspects: 1) METIS was developed based on a microblogging service platform. As such, it is not limited by the information available in any specific e-commerce website. In addition, METIS is able to track users' purchase intents in near real-time and make recommendations accordingly. 2) In METIS, product recommendation is framed as a learning to rank problem. Users' characteristics extracted from their public profiles in microblogs and products' demographics learned from both online product reviews and microblogs are fed into learning to rank algorithms for product recommendation. We have evaluated our system in a large dataset crawled from Sina Weibo. The experimental results have verified the feasibility and effectiveness of our system. We have also made a demo version of our system publicly available and have implemented a live system which allows registered users to receive recommendations in real time.

Categories and Subject Descriptors

I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text analysis*; H.3.1 [**Information Storage and Retrieval**]: text mining

KDD'14, August 24–27, 2014, New York, NY, USA.

http://dx.doi.org/10.1145/2623330.2623351.

Keywords

E-commerce; product recommender; product demographic; microblog

1. INTRODUCTION

Recent years have witnessed a great success of e-commerce websites such as Amazon and eBay as they transcend geophysical barriers and allow individuals or business to form transactions anywhere and anytime. A technique widely adopted by e-commerce companies is to exploit *product recommender systems* to improve user experience and increase sales. Research has found that product recommendation largely influences consumers' purchase decisions and is likely to boost sales [23, 26, 14].

Generally speaking, there are two major challenges in the design of product recommender systems. Firstly, it is difficult to find out users' demographic information, which is essential for making right recommendations to right persons. Existing recommender systems relying on collaborative filtering explore techniques for matching users with similar interests and make recommendations on this basis. Nevertheless, it is well-known that collaborative filtering suffers from the "cold-start" problem when a recommender system knows little about a new user. Secondly, existing recommender systems embedded in e-commerce websites can only make recommendations when users are performing e-commerce activities in those websites, which cannot capture users' instantaneous purchase intents outside those websites.

Although there has been much research work on online product recommendation [24, 10, 14], most studies only focus on constructing solutions for certain e-commerce websites and are thus constrained by the information available there. In our work here, rather than relying on limited information available in any specific e-commerce website, we aim to develop a generic online product recommender system by exploring vast amount of information available externally such as that on social media platforms.

Online social media has already become the new arena of our lives and involved different aspects of our social presence from day to day. Through online activities such as chatting with friends and posting short status updates, online social networks (OSNs) have become important platforms where users discuss their needs and desires [11], and even disclose their personal information [3]. Thus, in this paper, we propose to capture users' purchase intents from OSNs and develop a generic product recommender system not limited to any specific e-commerce website. In particular, we develop our recommender system based on a microblogging service

platform due to the following reasons: 1) Microblogs contain abundant data from which users' purchase intents can be easily discovered; 2) Microblogs are continuously updated which implies that users' purchase intents are also updated in real time. This is especially attractive for developing an effective product recommender system; 3) Microblogs contain public profiles of users, including age, sex and/or professions, from which we can extract users' demographic characteristics.

In this paper, we present the development of a novel product recommender system called *METIS*, a MErchanT Intelligence recommender System. The main characteristics of METIS that distinguishes itself from traditional product recommender systems are as follows: 1) METIS was developed based on a microblogging service platform which naturally addresses the aforementioned two challenges. First, users' public profiles on microblogs can be used to extract their demographic characteristics. Second, users' purchase intents can be instantaneously discovered from their tweets and the constantly updated purchase intents allow the development of a more effective recommender system not constrained by the limited information available in any specific e-commerce website. 2) The core to the METIS system is a novel demographic based recommendation algorithm which learns product demographics by leveraging knowledge from online social media. Product demographic, sometimes called the target audience, of a product or service is a collection of the characteristics of the people who buy that product. The features derived from both product demographics and the extracted users' characteristics can be fed into any learning to rank algorithm for real-time product recommendation. 3) We also propose an effective method for semi-automatic acquisition of training data for the learning to rank algorithms. This makes our system more practical since it does not rely on actual transaction records from any e-commerce website.

2. OVERVIEW OF THE SYSTEM

Our system, METIS, aims to link users' purchase intents with the real-world products in an online fashion. In specific, when a user published a tweet describing her purchase intent, METIS will first identify her purchase intent, then extract her demographic characteristics, and finally return a list of recommended products based on some similarity measurement between user's demographic and products' demographic information. The architecture of METIS is shown in Figure 1 which consists of three major components:

- *Purchase intent detection.* This component aims to detect users' purchase intents in near real-time. In order to reduce noise, irrelevant tweets are first filtered using a manually constructed keyword list. Then a classification-based method [11] is employed to identify tweets containing purchase intents. While only textual features were used to learn classifiers in [11], we propose to consider users' demographic information in addition to the lexical and syntactic information derived from tweets.
- *Demographic information extraction.* This component is divided into two parts: user demographics extraction and product demographics extraction. In the user side, we extract users' demographic information from their public profiles in a microblogging site; while in the product side, we propose two ways to leverage information from social media by extracting online product reviews on e-commerce websites and the follow-

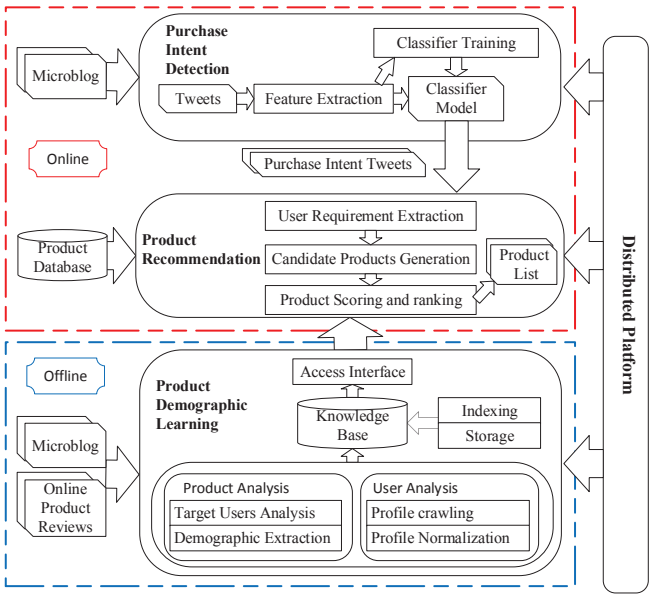

Figure 1: Architecture of METIS.

ings/mentions information on microblogs. For both users and products, we map their demographic information into the same demographic attribute feature space.

- *Product recommendation.* This is the core component of the system which returns a list of recommended products to a user. We propose a novel demographic-based recommendation algorithm where similarity measurement is performed between a user and products based on features derived from their demographic information which are subsequently combined in a learning to rank framework for making product recommendation with high accuracy.

For the development of our system, we choose a microblogging platform, Sina Weibo[1], which is the largest Chinese microblogging site. We have crawled all the information for a set of 5 million active users on Weibo via an authenticated API, including their tweets, following relationships and public profiles. We have retrieved a total of 1.7 billion tweets from these 5 million active users within a half-year time span from January 2013 to June 2013. We also choose a e-commerce website, Jingdong[2], the largest B2C e-commerce website, for the selection of products for making recommendations. It is worth mentioning that our system can be easily extended to cover multiple e-commerce websites. For our experiments, we choose three popular product types in Jingdong: laptop, camera and phone. In total, these three product categories contain 3,155 products and 1.13 million user reviews. To deal with such big data, we build our backend system based on distributed indexing which allows the retrieval of information of users or products in a quick way.

Preliminary concepts

We present some preliminary concepts before delving into the details of our system.

[1] http://weibo.com
[2] http://jd.com

Purchase-intent tweet. By following the definition in [11], a tweet is defined as a *purchase-intent tweet* if it explicitly expresses a desire or interest of buying some product. Here we only consider explicit expressions of buying desires and ignore implicit purchase intents since the latter is more difficult to detect. Note that we do not assume that the author of a tweet is the one who wants to buy a potential product. In an example tweet below:

> Please recommend! *My son* wants to buy a *Samsung phone* less than $200.

The potential customer of *Samsung phone* is *My son*, not the author of the tweet.

Product demographics. The product demographics [3], 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. A demographic profile (often shortened as "a demographic") provides enough information about the typical member of this product to create a mental picture of this hypothetical aggregate. For example, a product demographic might refer to the single, female, middle-class, age 18 to 24, college educated.

Knowing information such as the income status, age and tastes of users can help companies sell more, branch out to other groups and create more products that appeal to target buyers. Instead of identifying these useful demographic attributes manually, we start with the attributes in users' public profile on Weibo. With reference to the marketing studies [20, 28], we have identified six major demographic attributes: gender, age, marital status, education, career and interests. To quantitatively measure these attributes, we have further discretized them into different bins. We summarize the details of the demographic attributes in Table 1.

We use probabilities to characterize the demographics of a product. Formally, let $\mathcal{A}$ denote the set of attributes and $\mathcal{V}_a$ denote the set of values for an attribute a. For a product e, its demographic distribution of an attribute a is $\theta^{(e,a)} = \{\theta_v^{(e,a)}\}_{v \in \mathcal{V}_a}$, where $\theta_v^{(e,a)}$ is the proportion of the target consumers with the value v for attribute a. Since $\theta^{(e,a)}$ is a distribution, we have $\sum_{v \in \mathcal{V}_a} \theta_v^{(e,a)} = 1$. Furthermore, we use the set of attribute distributions to represent the product demographics, i.e. $\{\theta^{(e,a)}\}_{a \in \mathcal{A}}$.

Table 1: List of demographic attributes.

Attribute	Values
Gender	male, female
Age	1-11, 12-17, 18-30, 31-45, 46-59, 60+
Marital Status	single, engaged, loving secretly, married, relationship seeking, bereft of one's spouse, separated, divorced, ambiguous, loving
Education	literature, natural science, engineering, social sciences, medical science, art, others
Career	internet technology, designing, media, service industry, manufacturing, medicine, scientific research, management, others
Interests (Weibo tags)	travel, photographing, music and movie, computer games, Internet surfing, other

User profile. *User profile*, which is also called *user demographics*, is similar to *product demographics*. Formally, given a user u and the considered attribute set $\mathcal{A}$, we use $\mathcal{D}_u = \{v_a^{(u)}\}_{a \in \mathcal{A}}$ to denote the user's profile, where $v_a^{(u)}$ is the value of attribute a. On Sina Weibo, a user is required to fill in some attributes in her public profile during registration, which can be modified later.

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

3. PURCHASE INTENT DETECTION

The task of purchase intent detection on microblogs was first studied in [11] where a classification model is learned to classify a tweet automatically into one of the two categories (containing or not containing purchase intents). While only textual features were used in [11] for classifier training, we additionally explore users' demographic information in learning classifiers for purchase intent detection. Furthermore, only a small set of 1,335 tweets were tested in [11]. Instead, we have explored several ways to improve the efficiency of purchase intent detection in order to achieve the near real-time performance when dealing with large amount of social stream data.

3.1 Fast Generation of Candidate Purchase-Intent Tweets

Tweets are produced at a dramatic rate. There are on average 100 million tweets generated within a day on Sina Weibo. It is not feasible to directly apply purchase intent classification on such a huge amount of data. Thus, a keyword filtering method is first used to only keep tweets that are likely to contain purchase intents. An annotator major in commerce service is invited to compile a list of *purchase indicator keywords* which is shown in Table 2. We process the data stream in an online mode by distributing the data into multiple servers. We use a fast string search algorithm to keep tweets that contain at least one purchase indicator keyword in our list for subsequent processing.

Table 2: List of purchase indicator keywords and their English translations.

buy (买), recommend (推荐), change (换), which is better (哪个更好), cheap (便宜), cost (价值), auction (拍), on sale (降价), price (价格), need (需要), shopping (购物)

3.2 Intent Classification

We train an intent classifier based on the following features:

Textual features. Hollerit et al. [11] used highly discriminative n-gram features which consist of consecutive words (e.g. "buy cheap") or Part-of-Speech (POS) tags (e.g. "VB JJ") ranked by the chi-squared statistic for training intent classifiers. Intuitively, a word together with its POS tag in a tweet can represent more complete semantics than a single word or POS tag. Thus, we consider a word together with its POS tag as a feature unit. For example, for a phrase "buy VB cheap JJ", instead of generating two separate features "buy cheap" and "VB JJ", we only generate a single feature "buy-VB cheap-JJ" and call it *lexical-POS* feature.

Demographic features. The second type of features we consider is users' demographic information which is incorporated as additional context to supplement the textual evidence. For example, given two tweets respectively from a female and a boy, both tweets have mentioned "transformer model". Intuitively, the boy's tweet is more likely to contain a purchase intent. As such, we also extract users' demographic features containing the six attributes presented in Table 1 for intent classifier training.

4. PRODUCT DEMOGRAPHICS LEARNING

It is relatively easy to obtain users' demographic information from their public profiles in Sina Weibo. In this section,

we focus on extracting product demographic information, also called *target audience* or *target consumer*, from social media. Previously, product demographics are derived from either user surveys [1] or commercial purchase records [10]. As have been discussed earlier, we focused on the six demographic attributes shown in Table 1. We propose to leverage demographic related knowledge from online product reviews and the followings/mentions information on microblogs.

4.1 Demographics Extraction from Online Product Reviews

The first resource we consider is *online product reviews*. We have found that users might explicitly mention demographic related information in their reviews in addition to their opinions. For example, in a sentence "I *bought my son* this phone" from a product review, the phrase "buy my son" indicates that the current product is suitable for the author's son, who is a potential target consumer. In this example, we can see that "*buy somebody something*" is an important pattern for deriving the demographic knowledge which is embedded in a phrase *my son* which we call a *demographic phrase*. We want to collect statistics of such demographic related phrases in order to reliably derive patterns expressing demographic information.

We first propose to use a bootstrap algorithm to iteratively extract frequent patterns and demographic phrases. Then we further study how to convert these patterns into demographic statistics. The bootstrapping algorithm for the extraction of demographic phrases is presented in Algorithm 1, in which we iteratively extract new patterns and identify demographic phrases with the extracted patterns. The function ExtractDemographicPhrase (p,s) aims to identify the demographic phases in a review sentence s by using the pattern p. Here we only consider extracting nouns, noun phrases, adjectives, and numbers. Note that this algorithm is very flexible to deal with data in an online mode by setting the set of identified patterns $\mathcal{P}$ as the seed set for new data.

Given a product e, we obtain a set of its related demographic phrases $\mathcal{R}_e$ which have at least 10 occurrences in our data. We then map these phrases into six demographic dimensions as has been previously described in Table 1 using a list of pre-defined rules. For example, given a phrase "my little son", we can map it into two dimensions "male"(sex) and "12∼17"(age)". We maintain a counter $\#(a, v)$ which counts the number of times phrases being mapped to value v of the attribute a. Finally, we use Laplace smoothing (a.k.a add-one smoothing) to estimate the demographic distribution of a product e for an attribute a as follows

$$\theta_{a,v}^{(e)} = \frac{\#(a,v)+1}{\sum_{v' \in \mathcal{V}_a} \#(a,v') + |\mathcal{V}_a|}, \quad (1)$$

where $\mathcal{V}_a$ is the set of all possible values of attribute a. $\theta_{a,v}^{(e)}$ is a valid probability since $\sum_{v \in \mathcal{V}_a} \theta_{a,v}^{(e)} = 1$. Thus, for each attribute of a product, we can model the aggregated statistics as a distribution over a set of all possible attribute values, and a higher probability value indicates a more prominent characteristic of the product for that attribute.

4.2 Demographics Extraction from Microblogs

The second type of resources we consider is the information collected from *microblogs*. The rationale behind is that given a product, we could capture the "like" evidence, i.e. positive opinions, of users on the product. Such users can be treated as potential target audience for that particular product. As a result, the profiles of these users can be aggregated as the product demographics. We detect users' endorsements on a certain product in microblogs by considering the following two most common user behaviors:

- *following*: a brand or a product usually has its official account on microblogs. We take the followers of these official accounts as the potential target audience.
- *mentioning*: if a user is interested in a product, she may publish tweets to explicitly express her positive opinions on the product. Given a product, we retrieve the tweets containing the product name through simple keyword matching. We then use the machine learning based approach proposed in [17] for the detection of polarity $\{positive, negative\}$ of the tweet. We used review sentences with users' five-star ratings as training data: positive (> 3 stars) and negative (< 3 stars). Only tweets with *positive* polarity are treated as the supporting evidence, and their authors are considered as the product's target audience.

Given a product e, let $\mathcal{U}_e$ denote the set of its target users who are selected from either the *following* or the *mentioning* categories mentioned above. It is worth noting that in both *following* and *mentioning* categories, users may not be directly related to a specific product (e.g., "iPhone 5S"), instead they may be related to a brand (e.g., "iPhone") or a company (e.g., "Apple"). We introduce b_e to denote the brand or company of a product e. We then aggregate the target audience of b_e by following the similar approach, and denote it as $\mathcal{U}_{b_e}$. Also, for users in $\mathcal{U}_e$ and $\mathcal{U}_{b_e}$, we further perform spam user filtering by considering their followings/followers, tweet contents and interactions with other users.

The demographic distribution of a product e for attribute a taking the value v is calculated by combing the estimation

Algorithm 1: Bootstrapping algorithm for extracting demographic phrases from online reviews.

Input: review sentence corpus $\mathcal{C}$, seed purchase-intent patterns
Output: an set of identified demographic patterns $\mathcal{P}$ and a set of identified demographic phrases $\mathcal{R}$;
$\mathcal{P}' \leftarrow$ seed demographic patterns;
$\mathcal{P} \leftarrow$ seed demographic patterns;
$\mathcal{R}' \leftarrow \emptyset$;
$\mathcal{R} \leftarrow \emptyset$;
repeat
  $\mathcal{R}' \leftarrow \emptyset$;
  for *each pattern* $p \in \mathcal{P}'$ **do**
    for *each sentence* $s \in \mathcal{C}$ **do**
      if *p exists in s* **then**
        $\mathcal{R}' \leftarrow \mathcal{R}' \cup$ ExtractDemographicPhrase (p,s);
      end
    end
  end
  $\mathcal{P}' \leftarrow \emptyset$;
  for *each sentence* $s \in \mathcal{C}$ **do**
    for *each demographic phrase* $i \in \mathcal{R}'$ **do**
      $\mathcal{P}' \leftarrow \mathcal{P}' \cup$ GeneratePatterns(s, i);
    end
  end
  $\mathcal{P}' \leftarrow$ ExtractTopFrequentPatterns$(\mathcal{P}')$;
  $\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{P}'$;
  $\mathcal{R} \leftarrow \mathcal{R} \cup \mathcal{R}'$;
until *No new pattern is identified*;
return *An set of identified demographic patterns* $\mathcal{P}$ *and a set of identified demographic phrases* $\mathcal{R}$;

computed from $\mathcal{U}_e$ and $\mathcal{U}_{b_e}$ through the Jelinek-Mercer (JM) smoothing method:

$$\theta_{a,v}^{(e)} = (1-\lambda)\times\frac{\sum_{u\in\mathcal{U}_e}\mathbf{1}[u.a=v]}{\sum_{v'}\sum_{u\in\mathcal{U}_e}\mathbf{1}[u.a=v']}+\lambda\times\frac{\sum_{u\in\mathcal{U}_{b_e}}\mathbf{1}[u.a=v]}{\sum_{v'}\sum_{u\in\mathcal{U}_{b_e}}\mathbf{1}[u.a=v']}, \quad (2)$$

where $\mathbf{1}[\cdot]$ is the indicator function which returns 1 only when the condition is true and λ is the interpolation coefficient, which is empirically set to be 0.3 in our experiments. The target audience of a product is constructed based on both the information related to itself and that related to its corresponding brand or company.

Summary

In this section, we have presented two methods to leverage information from social media for product demographics learning. As such, we have two types of feature representations for product demographics. We do not attempt to combine them into a unified representation, instead, we keep them separate. As will be discussed later, the learning to rank algorithms can automatically assign optimal weights to these two feature types.

5. LEARN TO RANK FOR PRODUCT RECOMMENDATION

Now we discuss how to devise the product recommendation algorithm. A good recommendation algorithm should satisfy two main criteria [2]: 1) it is flexible to incorporate various types of useful information. 2) the algorithm is easy to understand for future extension and the ranking results are intuitive to explain. For the first criterion, we develop a learning to rank framework for product recommendation, which can be used on any types of features. For the second criteria, we represent products and users in the same demographic feature space, which makes our results easy to explain. Our proposed framework is generic and can be deployed with any ranking algorithms.

Our algorithm consists of two main steps: 1) candidate product generation; 2) learning to rank.

5.1 Candidate product generation

Recall that we have identified the purchase-intent tweets in *Purchase Intent Detection*. The authors of these tweets are treated as potential consumers, and the tweets reveal purchasing needs from users. In this section, we study how to generate candidate products based on users' profiles and their purchase-intent tweets.

Requirement identification. A user's requirement of a certain product can be derived from the identified purchase-intent tweets. We consider an example here:

> Please recommend! I want to buy a *Samsung phone* less than $200.

In this example, we need to detect the user's requirement of *price* < $200 for a brand, *Samsung phone*. To build a product knowledge base for products, we have indexed all the product related information such as brand, price, and screen size. Regular expressions have been compiled for each information field and users' requirements are identified by using the regular expression patterns. We generate a list of candidate products which fulfil the identified requirements.

Complicated query analysis. By default, the author of a purchase-intent tweet will be the potential customer of its related product. However, we have noticed that there are a considerable number of purchase-intent tweets, in which the potential consumer is not the author of the tweets. See the following example:

> Please recommend! I want to buy my son a *Samsung phone*.

The author of this tweet is a female, and if simply taking her as the potential customer of the phone, the recommender system would generate inappropriate recommendations. Although it is very challenging to solve this problem, our current approach naturally address this issue. Recall that we have extracted a set of demographic phrases such as "my son". We can create a set of virtual users with the profiles of this specific identity ("my son"). Once we have found that a tweet contains such an identify indicator, we will take its corresponding profile for product recommendation.

Candidate product list pruning. Although we only select the products which fulfil the identified requirements, the number of candidate products is still large which is not suitable for the learning to rank algorithms. For this reason, we select at most 30 best-sale products among the candidate products as the input to the learning to rank algorithms in the next step.

5.2 Learning to Rank

Having identified the purchase requirements of a potential consumer, we now discuss how to recommend products with the demographic information of both users and products. We formulate the task as a ranking problem and adopt the learning-to-rank algorithm as our solution. We first introduce the learning to rank framework briefly.

A brief introduction to learning to rank. In learning (training), a number of queries and their corresponding retrieved documents are given. Furthermore, the relevance levels of the documents with respect to the queries are also provided. The relevance levels are represented as ranks. The objective of learning is to construct a ranking function (model) which achieves the best results in ranking of the training data by minimizing a loss function. Ideally the loss function is defined on the basis of the performance measure used in testing. In retrieval (testing), given a query, the system returns a ranked list of documents in a descending order of the relevance scores which are calculated using the ranking function.

Ranking for product recommendation. Learning to rank was originally proposed for information retrieval. We first make a connection between product recommendation and information retrieval. In our task, a purchase-intent tweet can be understood as a query and an adopted product can be understood as a relevant document. For convenience, we use the term "*query*" to denote a purchase-intent tweet.

To formulate our product recommendation problem as a ranking task, we assume that there are a set of purchase-intent tweets (i.e. queries) $\mathcal{Q} = \{q^{(1)}, q^{(2)}, ..., q^{(m)}\}$ during training. A purchase-intent tweet (query) $q^{(i)}$ is associated with a set of $n^{(i)}$ candidate products $\{p_1^{(i)}, ..., p_{n^{(i)}}^{(i)}\}$. For each candidate product, let $y_j^{(i)}$ denote the judgment on product $p_j^{(i)}$ with respect to query $q^{(i)}$. The value of $y_j^{(i)}$

can be either discrete or continuous, and a higher value indicates a better recommendation for the query $q^{(i)}$. A feature vector $\boldsymbol{x}_j^{(i)}$ can be constructed for each query-product pair $(q^{(i)}, p_j^{(i)})$. The aim of the learning task is to derive a ranking function f such that, for each feature vector $\boldsymbol{x}_j^{(i)}$ (corresponding to $q^{(i)}$ and $p_j^{(i)}$), it outputs a recommendation score $f(\boldsymbol{x}_j^{(i)})$ for product ranking.

To learn the ranking function f, there are three general approaches [15]:

- Pointwise approach. It reduces ranking to regression or classification on single documents. That is, we build the ranking function f by approximating $f(\boldsymbol{x}_j^{(i)})$ to the corresponding relevance score $y_j^{(i)}$.
- Pairwise approach. It no longer assumes absolute relevance, which reduces ranking to classification on product pairs w.r.t. the same query. That is, given a query $q^{(i)}$, we only focus on the relative preference order between a pair of candidate products $p_j^{(i)}$ and $p_k^{(i)}$.
- Listwise approach. Instead of reducing ranking to regression or classification, it performs learning directly on product lists, and an entire ranked list is treated as a learning instance. That is, for query $q^{(i)}$, the ranking function focuses on learning a list of relevance scores $(y_1^{(i)}, ..., y_{n^{(i)}}^{(i)})$ given the list of feature vectors $(\boldsymbol{x}_1^{(i)}, ..., \boldsymbol{x}_{n^{(i)}}^{(i)})$.

We define our task in a generic way such that all the above approaches can be used. Another important problem is how to construct the feature vector for each instance, which is usually the basis of good performance for learning to rank algorithms. We mainly consider two types of features:

Query-independent product features. This type of features are independent of the specific query. Intuitively, the product sale follows "rich-gets-richer", i.e. a user is more likely to buy a product with more successful sales history and more positive comments. Based on this intuition, we use the following features: the sales history of a product (*sale*) and the overall rating score of a product (*rating*). We also incorporate the overall polarity score of a product (*polarity*) as the third feature. In our dataset, we have found that popular products usually have similar high rating scores. Thus, we consider looking into the reviews and derive the polarity scores based on the opinionated user reviews. Here we follow the unsupervised method in [22] with an open Chinese opinion lexicon to transform review content into polarity scores within the interval of $[-1, 1]$. For new products, we can smooth these features with the generalized information at the brand level.

Query-dependent product features. Query-dependent product features are extracted based on a potential user and a candidate product. We quantify the match degree between a user and a product based on their six-dimensional demographic attribute values (see Table 1). Formally, given a user u and a candidate product e, we set the feature value of attribute a as the probability in the demographic distribution of e:

$$x_a^{u,e} = \theta^{(e)}_{a, v_a^{(u)}}, \tag{3}$$

where $v_a^{(u)}$ denotes the value of attribute a for user u and $\theta^{(e)}_{a,\cdot}$ is the demographic distribution defined in Equation 1 and 2. Intuitively, if a user closely matches a product, the user will be a representative of the target consumers of the product. Note that we have learnt the demographic distributions from both online reviews and microblogs, and for each attribute, we can obtain two different feature values. Instead of simply combining both values, we keep them separate and let the ranking function learn the weights automatically. The weights of the demographic features learned by a ranking function provide an intuitive way to explain the results. Although we have only evaluated our proposed recommender algorithm on the aforementioned features, the algorithm can be easily extended to incorporate other types of features.

Semi-automatic acquisition of training data. The performance of learning to rank algorithms highly relies on the amount of training data. Previous studies tend to use transaction records (users' actual purchase decisions) from online e-commerce websites as supervised information [24]. However, such data are not always available as e-commerce companies could choose not to reveal their transaction data. Even with the availability of transaction data, it is still difficult to link users across microblogs and e-commerce websites, which makes it infeasible for the evaluation of our current task. Here, we propose a novel approach to acquire training data semi-automatically which is motivated by the following two example tweets from the same user:

> *Oct 1, 2012* Please recommend! I want to buy my son a Samsung phone.
> *Oct 4, 2012* Done! I *have bought* Galaxy II for my lovely son!

In the first tweet, a user expressed her desire to buy a Samsung phone for her son. The user subsequently reported that she has purchased a product. The above example forms a query-decision pair, which can be naturally converted into a training instance.

To extract such query-decision pairs, we first identify purchase-intent tweets. Then for each tweet, a set of tweets from the same author which were subsequently published in a week time have also been retrieved. We compile some cue phrases which are indicative of reporting purchase decisions, e.g., "*have bought* something". With such cue phrases, we can then identify those tweets potentially reporting purchasing decisions. We subsequently invite a human judge to manually examine whether these tweets actually report buying decisions. The query-decision pairs can then be used as training instances for the learning to rank algorithms.

6. EXPERIMENTS

In our work here, we chose Sina Weibo, the largest Chinese microblogging service, as the microblogging platform, and Jingdong, the largest Chinese B2C e-commerce company, as the e-commerce product database. Our recommender system aims to recommend products on Jingdong to users who have expressed explicit purchase intents on microblogs. We evaluate our recommender system in two aspects: purchase intent detection and product recommendation. **All the data used in our experiments will be made available.**

6.1 Evaluation on Purchase Intent Detection

Construction of the dataset. To construct the dataset for the evaluation of purchase intent detection, we first randomly select 10,000 tweets which contain at least one keyword expressing purchase intent as listed in Table 2. Then two annotators are asked to label the tweets as containing

purchase intents or not by following the guidelines introduced in [11]. The Cohen's Kappa agreement coefficient between annotators is 0.85, which indicates a high-level agreement. By only keeping the tweets which receive the same labels from both annotators, we end up with 4,434 purchase intent tweets and 5,352 non-purchase intent tweets.

Methods to compare. We evaluate the following methods for purchase intent detection:

- $Baseline_T$. Use textual features as proposed in [11].
- $Baseline_{T+D}$. Textual features + demographic features.
- $Ours_T$. Lexical-POS features.
- $Ours_{T+D}$. Lexical-POS features + demographic features.

We train the Support Vector Machines (SVMs) with two different kernels, linear and RBF, on the feature sets mentioned above.

Results and analysis. It can be observed from the results presented in Table 3 that using our proposed lexical-POS features ($Ours_T$) outperforms the baseline model using textual features only ($Baseline_T$). In particular, the improvement is more significant with the RBF kernel where over 13% improvement is observed. Adding demographic features to textual features ($Baseline_{T+D}$) improves upon $Baseline_T$ by 1% with the linear kernel and 4.7% with the RBF kernel. However, additionally incorporating demographic features into our proposed lexical-POS features ($Ours_{T+D}$) only gives marginal improvements compared to $Ours_T$. Overall, the best performance of 81.8% in F-measure is achieved using a combination of lexical-POS and demographic features trained on SVMs with the RBF kernel.

Table 3: Performance comparison for purchase intent detection with SVM.

Kernel	Methods	Precision	Recall	F-value
Linear	$Baseline_T$	0.738	0.771	0.754
	$Baseline_{T+D}$	0.743	0.788	0.765
	$Ours_T$	0.762	0.802	0.782
	$Ours_{T+D}$	0.77	0.806	0.788
RBF	$Baseline_T$	0.81	0.582	0.678
	$Baseline_{T+D}$	0.725	0.726	0.725
	$Ours_T$	0.796	0.829	0.812
	$Ours_{T+D}$	0.805	0.832	0.818

6.2 Evaluation on Product Recommendation

Construction of the dataset. We choose three popular product types, phone, camera and laptop, for the construction of our dataset for the evaluation of product recommendation. We follow the method of semi-automatic acquisition of query-decision pairs described in Section 5.2 to build our dataset. The statistics of these three product types is summarized in Table 4.

Table 4: Statistics of the dataset for product recommendation.

Types	#brands	#models	#query-decision pairs
phone	57	1,584	170
camera	25	724	496
laptop	25	829	437

Methods to compare. We evaluate our proposed recommender system using two representative learning to rank algorithms from each of the categories, pointwise, pairwise and listwise:

- *Pointwise*: MART [9], RandomForest (RF) [5];
- *Pairwise*: Ranksvm [12], RankBoost [8];
- *Listwise*: Listnet [6], AdaRank [27].

These learning to rank algorithms are trained on two types of features, query-independent features and query-dependent features. We use product sales (*sale*), product polarity scores (*polarity*), and product rating scores (*rating*) as query-independent features. The query-dependent features, or demographic based features, are derived from online reviews in the Jingdong website and microblogs from Sina Weibo, which are denoted as *JD* and *Weibo* respectively.

The open source toolkit, RankLib[4] is used for the implementation of these learning to rank algorithms. For each algorithm, five-fold cross-validation is performed and the results are averaged over five such runs.

In our task, the purchase history on e-commerce websites is not available. Thus, existing approaches relying on purchase history [24, 14] are not suitable to be used as baselines. For comparison, we built three baselines with each of them using one of the three query-independent features to rank products from our generated candidate product list.

Evaluation metrics. The evaluation metrics adopted here are precision at k ($p@k$) and Normalized Discounted Cumulative Gain (NDCG), which are commonly used in information retrieval. For $p@k$, we consider the relevance at the product model level, i.e. the specific product actually purchased as described in a tweet is treated as the only relevant product. To compute $p@k$, we calcuate the proportion of queries for which we have made the correct recommendations in the top k positions. NDCG takes into account both the ranking positions and the quality grade of products, and is defined as follows:

$$NDCG(\mathcal{Q},k) = \frac{1}{|\mathcal{Q}|}\sum_{j=1}^{|\mathcal{Q}|} Z_{k,j} \sum_{m=1}^{k} \frac{2^{R(j,m)}-1}{\log_2(1+m)}, \qquad (4)$$

where $\mathcal{Q}$ is the set of queries, $Z_{k,j}$ is the normalizer for the jth query which denotes the ideal ranking score for the top k positions, and $R(j,m)$ is the relevance level of the mth product for the jth query. For $R(j,m)$, we mainly consider three relevance levels: relevant (2), partially-relevant (1) and non-relevant (0). For each query, there is only one relevant product which matches a user's final purchase decision. If a product model does not match the final product purchased but has the same brand, it is treated as partially-relevant. For all the other cases, products are treated as non-relevant.

Overall performance comparison. It can be observed from Table 5 that the learning to rank methods are much more effective than the three simple baselines. Furthermore, the performance of the pointwise approach is much better than that of pairwise and listwise approaches. The listwise approaches perform the worst due to the difficulty in obtaining the full product ordered list for training since there is usually only one correct product decision and the ordering of irrelevant products is not available. In the pointwise category, *MART* works very well and nearly achieves the best performance for both $p@k$ and $NDCG@k$ in all the three categories, except that it is slightly worse than *RandomForest* for $p@5$ in the "Laptop" category. Overall, for all methods, the performance in the "Phone" category is much

[4] https://sourceforge.net/p/lemur/wiki/RankLib/
RankLib might assign an equal score to some items in the returned ranked list. In such cases, we order items with an equal score by their sales performance.

Table 5: Performance comparison of product recommendation on three datasets for $p@k$ and $NDCG@k$.

Types	Metrics	Baselines			Pointwise		Pairwise		Listwise	
		sale	polarity	rating	MART	RF	RankSVM	RankBoost	Listnet	AdaRank
PHONE	p@1	0.006	0.006	0.006	**0.106**	0.041	0.018	0.012	0.006	0.006
	p@5	0.041	0.018	0.029	**0.282**	0.253	0.141	0.112	0.124	0.047
	NDCG@1	0.094	0.094	0.015	**0.226**	0.126	0.085	0.097	0.129	0.165
	NDCG@5	0.170	0.093	0.099	**0.256**	0.239	0.168	0.165	0.176	0.176
CAMERA	p@1	0.000	0.000	0.004	**0.606**	0.466	0.041	0.062	0.033	0.004
	p@5	0.238	0.004	0.018	**0.688**	**0.688**	0.183	0.310	0.152	0.244
	NDCG@1	0.128	0.128	0.008	**0.641**	0.515	0.092	0.140	0.073	0.127
	NDCG@5	0.254	0.094	0.061	**0.495**	0.466	0.146	0.232	0.115	0.253
LAPTOP	p@1	0.019	0.000	0.000	**0.608**	0.599	0.140	0.055	0.133	0.017
	p@5	0.026	0.014	0.012	0.653	**0.665**	0.439	0.102	0.302	0.026
	NDCG@1	0.239	0.002	0.002	**0.695**	0.686	0.145	0.228	0.138	0.220
	NDCG@5	0.212	0.073	0.059	**0.573**	0.559	0.223	0.219	0.172	0.202

worse than that of "Camera" and "Laptop". This is because there are more brands and models but much fewer training instances in the "Phone" category compared to the other two categories as shown in Table 4.

Feature selection and analysis. As have been previously mentioned, we have a set of three query-independent features, *sale*, *polarity* and *rating*, denoted as *spr*. We also have two types of query-dependent features *Weibo* generated from microblogs in Sina Weibo, and *JD* derived from online reviews in Jingdong. We build our recommender systems using the best learning to rank algorithm, *MART*, trained from different types of the features. The results are presented in Table 6. We can see that 1) the combination of all the features achieves the best performance; and 2) demographic features derived from online reviews in Jingdong and those extracted from microblogs are effective to improve the recommendation performance. Combining these two types of demographic features (Weibo + JD) yields a competitive performance compared to that of using all the features.

Table 6: Performance comparison with different type of features for *MART*. *Both* denotes *Weibo* + *JD*.

Types	Metrics	spr	Weibo	JD	Both	all
p@5	PHONE	0.194	**0.282**	0.224	0.259	**0.282**
	CAMERA	0.680	0.485	0.476	0.676	**0.688**
	LAPTOP	0.606	0.245	0.520	0.537	**0.653**
NDCG@5	PHONE	0.157	0.247	0.199	0.228	**0.256**
	CAMERA	0.471	0.346	0.331	**0.495**	**0.495**
	LAPTOP	0.525	0.258	0.343	0.452	**0.573**

Table 7: Samples of the learnt product demographics based on online reviews. Real numbers denote the learned weights for the corresponding attribute values.

Galaxy S4 (White)	$(SEX, [\text{“}male\text{”}, 0.271], [\text{“}female\text{”}, 0.729])$
Galaxy S4 (Blue)	$(SEX, [\text{“}male\text{”}, 0.688], [\text{“}female\text{”}, 0.312])$
Galaxy S4 (Black)	$(SEX, [\text{“}male\text{”}, 0.852], [\text{“}female\text{”}, 0.148])$
Galaxy S4 (White)	$(Age, [\text{“} < 45\text{”}, 0.931], [\text{“} \geq 45\text{”}, 0.069])$
Galaxy S4 (Blue)	$(Age, [\text{“} < 45\text{”}, 0.755], [\text{“} \geq 45\text{”}, 0.245])$
Galaxy S4 (Black)	$(Age, [\text{“} < 45\text{”}, 0.650], [\text{“} \geq 45\text{”}, 0.350])$

Qualitative analysis of demographic features. Table 6 shows that demographic features are very effective in product recommendation. Here we present in Table 7 some learned demographic features for a phone product, Samsung Galaxy S4. In specific we only show the associated demographic features for "color". It can be observed that the demographic distributions indeed varies with different colors. Young females prefer *white* phones while young males like *black* phones more.

Table 7 shows some learned product demographic features of two different phone brands, Apple and Samsung. We have a couple of interesting observations, 1) Samsung has a more balanced sex distribution; 2) Apple products are more preferred by the consumers in the IT field.

Table 8: Samples of the learnt product demographic based on microblogs.

Apple	$(SEX, [\text{“}male\text{”}, 0.593], [\text{“}female\text{”}, 0.407])$
	(CAREER, [“IT”, 0.28], [“management”, 0.219] , [“media”, 0.172], [“industry”, 0.139])
	(TAG, [“music&movie”, 0.316], [“travel”, 0.249] , [“Internet surfing”, 0.163], [“computer games”, 0.161])
Samsung	$(SEX, [\text{“}male\text{”}, 0.503], [\text{“}female\text{”}, 0.497])$
	(CAREER, [“management”, 0.252], [“IT”, 0.223] , [“industry”, 0.252], [“media”, 0.223])
	(TAG, [“computer games”, 0.281], [“travel”, 0.27] , [“music&movie”, 0.209], [“Internet surfing”, 0.188])

Impact of the completeness of user profiles. In Table 1, we have presented the six demographic attributes. It is worth noting that not all users will fill in all the attributes in their public profiles. We want to check how the system performance is affected by the completeness of user profiles. We first analyze the dataset of three product categories in Table 4, where a query-decision pair corresponds to a unique Weibo user. The average number of attributes a user has filled in for each product category is as follows: 2.73 ± 1.10 (laptop), 2.66 ± 1.12 (camera) and 2.81 ± 1.07 (phone). Our evaluation results show that our system generally works well when a user has filled in three or more attributes, and *age*, *sex* and *interest tags* have been identified as the top three most important attributes by using the leave-one-out evaluation test. To examine the applicability of our system on a large Weibo user population, we compute the proportions of the user demographic attributes that can be found in the public profiles of the 5 million Weibo users: *sex* (100%), *age* (36.7%), marital status (4.6%), education (26.3%), career (12.9%) and *tags* (65.7%). It can be seen that apart from marital status, all the other attributes have considerable filled-in values, especially the identified top three most important attributes. Among the 5 million Weibo users, 44.1% of them have filled in at least three attributes and 26.5% of them have filled in all the identified top three most important attributes. These statistics provide a support evidence to our demographic based approach.

6.3 Efficiency analysis

Our dataset consists of 5 million users and 1.7 billion tweets, and about 113 tweets are published every second. As mentioned in Section 3, we use keyword filtering to reduce the number of tweets for subsequent processing. That is, tweets that do not contain any of the keywords express-

ing purchase intents are filtered. After keyword filtering, 3,380,894 tweets have been retained. By averaging this statistics into seconds, there would be 0.216 tweet generated per second. Our dataset covers roughly 10% of the entire Weibo data stream from active users. In this case, our purchase intent detection module will need to deal with about $0.216 \times 10 = 2.16$ tweets per second. We conduct the experiments on servers equipped with an Quad-core AMT opteron(tm) processor 8380 operating at 2.5 GHz running Red Hat Linux version 9.1 and equipped with 70 GB of physical memory. Table 9 shows the running time performance of various steps in our recommender system by running on a single server with a single thread. Here, we chose linear kernel for SVM for the trade-off between accuracy and speed for purchase intent detection. It can be observed from Table 9 that our approach has a very competitive throughout of tweets per second, which is much larger than 2.16.

Table 9: Running time performance of each step in our recommender system. "t/s" denotes "tweets per second".

Purchase intent keyword filtering	751,880 t/s
Feature extraction for intent detection	301.16 t/s
Purchase intent classification (SVM linear)	217.10 t/s
Requirement identification & Feature extraction for recommendation	233.48 t/s
Ranking with MART	204.29 t/s

7. DEPLOYMENT OF THE REAL SYSTEM

We have implemented two versions of the aforementioned product recommender system (in Chinese).

Demo system. We have implemented a demo system which simulates the Weibo stream by processing our canned data collected between January and June of 2013. The system processes each tweet in turn and when it identifies a purchase-intent tweet, it will make an alert in the user interface. If we put the mouse over the tweet, the system will display the product recommendation results. The demo system is available at `http://sewm.pku.edu.cn/metis`.

Weibo service. To allow our recommender system deal with live tweets, we have set up *an official account* of our system (`http://weibo.com/u/3926085440`) and *a system response* account (`http://weibo.com/u/3802166078`). If a user wants to receive the recommendation service from our system, she can follow our official Weibo account. Our system maintains a pool of followers, and constantly monitors their tweets. Once we have detected a purchase-intent tweet, our system will generate a tweet which contains the recommendation results and a short URL link pointing to the detailed product information via the response account. The tweet contains a mention to the author of the original purchase-intent tweet. Note that the our Weibo service is limited by Sina Weibo API, and it can only deal with a small amount of requests per day.

Architecture description. A distributed platform is constructed for parallel processing, which consists of a master server and five child servers with the same configuration described in Section 6.3. Each child server keeps a local copy of all the product information. The master node coordinates the activities of the child nodes for load balancing. On each child server, an incrementally distributed index system has been implemented for storing and retrieving tweets, users profiles and product information. Each child server runs a *purchase intent detection* module and a *product recommendation* module in a pipeline mode, i.e. the identified purchase intent tweets are directly forwarded into the *product recommendation* module on the same server as shown in Figure 1. Finally, the master server collects and returns the results. User privacy has also been considered in the system design: all the user profiles are encrypted in our system and are transparent to the developers.

8. RELATED WORK

Our work is mainly related to three lines of research:

Product recommendation. Early work on product recommendation mainly relies on collaborative filtering which makes recommendations based on matching users with similar interests [19, 2, 14]. Collaborative filtering suffers from the "cold start" problem. Recently, there have been some attempts to incorporate information from online social networks into recommendation [21]. In particular, demographic information has been shown to be useful to improve the recommendation performance [13, 18, 10]. Our work is related to the aforementioned research. Nevertheless, we presented the first study which identifies users with purchase needs on social networks and make product recommendations by jointly considering both users' and products' demographic information. Some design issues and suggested guidelines for the development of product recommender systems have been previously discussed in [23, 26]. We have followed the suggested guidelines but proposed different methodologies for the implementation of our system.

Learning to rank. Learning to rank was originally proposed in information retrieval, where it aims to incorporate various features in a formal way to improve the ranking performance of the retrieved results [15]. In this paper, we formulate the product recommendation task as a learning to rank problem and evaluate various learning to rank algorithms on both the query-dependent and query-independent features derived from social media data. The learning to rank algorithms we tested include pointwise approaches, MART [9] and RandomForest [5], pairwise approaches, RankSVM [12] and RankBoost [8], and listwise approaches, ListNet [6] and AdaRank [27].

Social networking mining. Commercial intent detection and analysis has been performed on the Web data [7] and social media data such as Twitter [11]. The correlation between external events and commercial intents on microblogs have been studied in [25]. We also consider automatically detecting purchase intents from microblogs but with different feature representation and with an additional incorporation of users' demographic features. Furthermore, we have explored efficient implementation of purchase intent detection in order to achieve the near real-time performance when dealing with large amount of social stream data.

In recent years, there has also been some work on identifying individual's demographic characteristics such as age, gender and interests from social media data [16, 4]. In this paper, we directly extract users' demographic information from their public profiles in Sina Weibo. We will explore automatic methods in inferring users' demographic attributes as future work.

9. CONCLUSIONS

In this paper we have presented a novel demographic based product recommender system which detects users' purchase intents from their microblogs in near real-time and makes product recommendations based on matching the users' de-

mographic information extracted from their public profiles with product demographics learned from microblogs and online reviews. Unlike traditional product recommender systems which are often designed for some specific e-commerce websites and can only make recommendations when users are performing e-commerce activities in those websites, our system is not constrained by any particular e-commerce website and can make instantaneous product recommendations to users who have expressed their purchase intents in micoblogs.

We have conducted extensive experiments on large-scale microblog data crawled from Sina Weibo. The experimental results have verified the feasibility and effectiveness of our proposed recommender system. We have also made a demo version of our system publicly available and have implemented a live system which allows registered users to get recommendations whenever they need. We believe our study will have profound influence on both research and industrial communities. In future work, apart from exploring methods to automatically infer users' demographic information from social media data, we will also investigate other potentially useful features to be incorporated into our recommender system.

ACKNOWLEDGEMENTS We thank the anoymous reviewers for his/her thorough review and highly appreciate the comments. We thank Prof. Ming Zhang for the help with the data collection. This work was partially supported by the National Key Basic Research Program (973 Program) of China under grant No. 2014CB340403, 2014CB340405, NSFC Grant 61272340. Yulan He was supported by the UK' s EPSRC grant EP/L010690/1. Xin Zhao was supported by MSRA PhD fellowship. Xin Zhao and Yanwei Guo contributed equally to this work and should be considered as joint first authors. Xin Zhao is the corresponding author.

10. REFERENCES

[1] Us demographic and business summary data. *Product guide*, 2012.

[2] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE TKDE*, 17(6), June 2005.

[3] Jin Yeong Bak, Suin Kim, and Alice Oh. Self-disclosure and relationship strength in twitter conversations. ACL '12, 2012.

[4] Bin Bi, Milad Shokouhi, Michal Kosinski, and Thore Graepel. Inferring the demographics of search users: Social data meets search queries. WWW '13, 2013.

[5] Leo Breiman. Random forests. *Mach. Learn.*, 45(1), October 2001.

[6] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: From pairwise approach to listwise approach. ICML '07, 2007.

[7] Honghua (Kathy) Dai, Lingzhi Zhao, Zaiqing Nie, Ji-Rong Wen, Lee Wang, and Ying Li. Detecting online commercial intention (oci). In *WWW '06*, 2006.

[8] Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.*, pages 933–969, 2003.

[9] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

[10] Michael Giering. Retail sales prediction and item recommendations using customer demographics at store level. *SIGKDD Explor. Newsl.*, 10(2), December 2008.

[11] Bernd Hollerit, Mark Kröll, and Markus Strohmaier. Towards linking buyers and sellers: Detecting commercial intent on twitter. WWW '13 Companion, 2013.

[12] Thorsten Joachims. Training linear svms in linear time. KDD '06, 2006.

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

[14] Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1), January 2003.

[15] Tie-Yan Liu. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3), March 2009.

[16] Alan Mislove, Bimal Viswanath, Krishna P. Gummadi, and Peter Druschel. You are who you know: Inferring user profiles in online social networks. WSDM '10, 2010.

[17] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. ACL '04, 2004.

[18] Lingyun Qiu and Izak Benbasat. A study of demographic embodiments of product recommendation agents in electronic commerce. *Int. J. Hum.-Comput. Stud.*, 68(10):669–688, October 2010.

[19] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. WWW '01, 2001.

[20] G. Sridhar. Consumer involvement in product choice - a demographic analysis. *XIMB Journal of Management*, 2007.

[21] Panagiotis Symeonidis, Eleftherios Tiakas, and Yannis Manolopoulos. Product recommendation and rating prediction based on multi-modal social networks. RecSys '11, 2011.

[22] Peter D. Turney. Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. ACL '02, 2002.

[23] Felix von Reischach, Florian Michahelles, and Albrecht Schmidt. The design space of ubiquitous product recommendation systems. MUM '09, 2009.

[24] Jian Wang and Yi Zhang. Opportunity model for e-commerce recommendation: Right product; right time. SIGIR '13, 2013.

[25] Jinpeng Wang, Wayne Xin Zhao, Haitian Wei, Hongfei Yan, and Xiaoming Li. Mining new business opportunities: Identifying trend related products by leveraging commercial intents from microblogs. In *EMNLP*, 2013.

[26] Bo Xiao and Izak Benbasat. E-commerce product recommendation agents: Use, characteristics, and impact. *MIS Quarterly*, 31:137–209, 2007.

[27] Jun Xu and Hang Li. Adarank: A boosting algorithm for information retrieval. SIGIR '07, 2007.

[28] Valarie A Zeithaml. The new demographics and market fragmentation. *Journal of Marketing*, 49:64–75, 1985.