Mobile App Retrieval for Social Media Users via Inference of Implicit Intent in Social Media Text

Dae Hoon Park*
Yahoo! Inc.
Sunnyvale, CA 94089, USA
daehoon@yahoo-inc.com

Yi Fang
Department of Computer Engineering
Santa Clara University
Santa Clara, CA 95053, USA
yfang@scu.edu

Mengwen Liu
College of Computing and Informatics
Drexel University
Philadelphia, PA 19104, USA
ml943@drexel.edu

ChengXiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA
czhai@cs.illinois.edu

ABSTRACT

People often implicitly or explicitly express their needs in social media in the form of "user status text". Such text can be very useful for service providers and product manufacturers to proactively provide relevant services or products that satisfy people's immediate needs. In this paper, we study how to infer a user's intent based on the user's "status text" and retrieve relevant mobile apps that may satisfy the user's needs. We address this problem by framing it as a new entity retrieval task where the query is a user's status text and the entities to be retrieved are mobile apps. We first propose a novel approach that generates a new representation for each query. Our key idea is to leverage social media to build parallel corpora that contain implicit intention text and the corresponding explicit intention text. Specifically, we model various user intentions in social media text using topic models, and we predict user intention in a query that contains implicit intention. Then, we retrieve relevant mobile apps with the predicted user intention. We evaluate the mobile app retrieval task using a new data set we create. Experiment results indicate that the proposed model is effective and outperforms the state-of-the-art retrieval models.

1. INTRODUCTION

With rapid development of Internet, people leave massive amount of their status messages on social media. Everyday, 500 million tweets are left on Twitter[1], 55 million status updates are made on Facebook[2], and 80 million photos (with text descriptions) are shared on Instagram[3]. A myriad of such user-generated text data provided researchers opportunities to analyze them. For example, real-time events detection [35], sentiment analysis in tweets [1], interestingness prediction for tweets [29], and stock market prediction based on tweets [43] have been studied.

More recently, researchers studied commercial intention of tweets (*e.g.*, [17, 13]). In social media, people discuss various topics such as their current status including what they do, how they feel, and where they are at the moment. Often, social media users express their intention to purchase a product in an implicit or explicit way. For example, a sentence "I will buy an iPhone" contains the user's explicit intention to buy the product. In another sentence "I lost my cellphone," the user does not explicitly express the intention to buy a cellphone, but we can infer that the user may want to purchase a cellphone (implicit intention). Most existing work focused on detection of such commercial intention, and only few work studied on product recommendation based on commercial intention in social media.

In this work, we study the task of mobile app recommendation for social media text that contains implicit intention. People often use mobile phones to update their status. Recommending mobile apps thus can be an immediate solution to users who implicitly express their needs using mobile devices. We formulate the problem as information retrieval problem where we retrieve mobile apps that satisfy a query, which is user status text with implicit intention. In order to match implicit intention text with mobile apps, (`i`) we first infer possible user intentions in the query and then (`ii`) rank apps based on their relevance to the inferred intentions. For example, given a user query "I am hungry", we infer possible intentions such as "find nearby restaurants" and "browse recipe books", and then we recommend apps that can find nearby restaurants or show recipes. Note that a query with implicit intention is different from traditional ad-hoc search queries, where users explicitly express their intentions. To learn possible explicit intentions for texts that contain implicit intention, we leverage social media. We collect tweets that contain information about what people truly desire when they implicitly express their needs. We then build parallel corpora of implicit and explicit intention texts, which we exploit to infer user needs.

Although we can recommend apps based on other kinds of data, including frequently used apps and smartphone sensor data, we do not employ them because they are not directly expressed by users. On the other hand, user status text provides more direct information such as a user's current thoughts and needs, which are difficult to obtain indirectly.

*This work was done while the author was at the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA.

[1]http://www.internetlivestats.com/twitter-statistics/

[2]https://blog.kissmetrics.com/facebook-statistics/

[3]https://www.instagram.com/press/

CIKM'16 , October 24 - 28, 2016, Indianapolis, IN, USA

ISBN 978-1-4503-4073-1/16/10...$15.00

DOI: http://dx.doi.org/10.1145/2983323.2983843

Moreover, the other types of data may not always be available. For example, users are often unwilling to share their smartphone's sensor data with recommendation systems due to privacy concerns. Therefore, we focus on inferring intention in user status text in this paper.

Inference of intention in user status text is an important problem. By analyzing intention in user status, product manufacturers or service providers can provide relevant items or targeted ads to the users even when the intention is implicitly expressed. Such recommendation can also benefit users since they can find products or services they need without searching for them. While users often update user status with explicit intention, they more frequently reveal their needs implicitly. For example, researchers have found that there are about twice as many tweets with implicit commercial intention than those with explicit commercial intention [17, 13]. Therefore, we focus on analyzing implicit intention and recommendation based on it although analyzing implicit intention is more challenging than analyzing explicit intention [13].

This work makes the following contributions:

1. We introduce and study a novel problem of recommending mobile apps for user status text containing implicit intention. To the best of our knowledge, there has been no research work that studied the same problem as ours.

2. We propose a novel approach to retrieve mobile apps that satisfy implicit intention of users. We first infer user intention using parallel corpora we build from social media. Then, we measure relevance of mobile apps to the inferred intention in order to rank them. To the best of our knowledge, no research work has leveraged parallel corpora for user intention analysis. Our model is general, so it can be applied to other product or service domains.

3. Since the task has never been performed in the literature, we create a new test data set for evaluating different models, and we make the data set public. We employ crowdsourcing to label data with query relevance. The test collection is available at http://timan.cs.uiuc.edu/downloads.html.

2. RELATED WORK

Understanding search intents behind queries has been a great challenge in information retrieval systems. Traditionally, Web search engines take user queries from a single text input box, which makes the systems analyze search intents from a short (about two terms per query [18]) list of keywords. In order to better understand queries, researchers classified queries into different types [8, 20, 26]. For example, Broder [8] classified queries into three classes: navigational, informational, and transactional. The author showed how search engines could evolve by supporting different search intents. Contextual search [27, 10], which uses a user's contextual information to better capture the user's search intent, is related to our work in that we exploit a user's status text in social media, which contains the user's implicit intention. However, we regard each user status text as a single query that contains a user's (implicit) intention, while contextual search requires contextual information in addition to the query. Our work is also related to content-based recommendation systems [33], but our work differs in that we recommend items based on a user's immediate needs in a query while they typically recommend items based on the user's general interests revealed in a user profile.

Query recommendation (or suggestion) has been widely studied to recommend alternative related queries given a query, in order to help users who would repeatedly rephrase their queries [4, 22]. Instead of suggesting alternative queries to users, query expansion tries to reformulate queries to resolve vocabulary gap problem [15], and it is surveyed well in [11]. Our work extends query expansion in the sense that we automatically adjust a given implicit query to an explicit query. However, our work differs from query expansion in that the target query is not supposed to be an "expansion" of the original query but "conversion" of the original query to predict hidden intention in it. Thus, the original query and the expanded query may carry completely different meaning in our work. Query translation [30] and cross-lingual information retrieval [6] are also related to our work in that our work can be seen as translating an implicit query to an explicit query. However, our work differs from them in that the same language is used for the original query and the translated query.

Besides general query intent, researchers studied commercial intention of users in online text, where commercial intention is defined in [12] as "a user has intention to purchase or participate in commercial services." Dai *et al.* [12] defined online commercial intention as commercial intention behind a user's online activities. They developed models to detect whether a query or Web pages a user visits lead to commercial activity or not. Ashkan *et al.* [3] and Guo and Agichtein [16] classified whether a query contains a commercial intent or not. Our work is related to commercial intent analysis in that we try to connect queries to products (mobile apps). However, unlike those existing studies, we do not classify queries into commercial or noncommercial. Instead, we assume a user needs something, and we recommend products that can satisfy the user's hidden intent.

Recently, commercial intention analysis on social media has attracted researchers' attention. Hollerit *et al.* [17] performed the first commercial intention detection on social media in order to link product buyers and sellers. They distinguished between explicit and implicit commercial intention and stated that implicit intention also has an economic value. Other researchers [40, 38] also studied intention classification in social media. Ding *et al.* [13] also exploited social media to identify whether a user text has a commercial intention or not. They found that 625 out of 1,000 commercial intention tweets contain implicit commercial intention, and they claimed that detecting implicit commercial intention is more challenging. Likewise, most of the studies about social media commercial intention analysis focused on intention detection. Our work is related to social media commercial intention analysis since we too analyze intention in social media. However, our goal is to further retrieve mobile apps that meet user intention, which we infer from the user text.

Previously, few researchers studied product retrieval problem based on commercial intention tweets. Duan *et al.* [14] mined intention-related products in online Q&A community data. They used a pattern-based method to extract candidate products from the answers. Then, they measured relevance between intention and products to mine intention-related products. Their work can be regarded as the most similar one to ours since they connected intention with the products. However, our work is different from their work. We focus on how implicit queries can be converted to explicit ones while they do not study such relationship. Also, they mine products in the Q&A data, so they cannot recommend products that are not present in the data. We do not require products to be present in the social media data, so the scope of recommended products is not limited.

3. PROBLEM DEFINITION

We study how to infer a user's intention in a user's status text segment, in order to recommend mobile apps that would satisfy the user's intention. This is a new entity retrieval task where a query q is a user's status text segment that implicitly expresses the user's needs, and the entities are mobile apps $\boldsymbol{A} = \{a_1, ..., a_M\}$. Each mobile app a has its

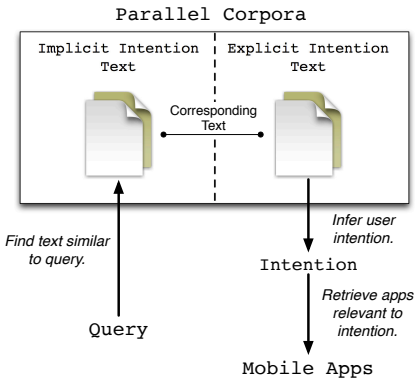

Figure 1: Overview of our approach.

text representation. We are also given social media text data $\boldsymbol{D}$, which we can leverage to infer the user intent. Thus, our goal is to retrieve a list of mobile apps for each q based on their text representations, where the apps are ordered according to their relevance to the user intent in q.

We distinguish between explicit queries and implicit queries. While explicit queries contain their intention clearly, implicit queries do not. For example, user status texts such as "I'm hungry", "I'm so tired", and "I have no one to talk to" do not reveal its intention explicitly while they certainly imply the needs for something, so we classify them into implicit queries. The corresponding explicit queries may be "I want food", "relaxing music", and "dating app", respectively, and these types of explicit queries have been input to traditional search engines. In this paper, we focus on entity retrieval based on implicit queries since such queries occur more often in social media text [17, 13].

To the best of our knowledge, retrieval of mobile apps given user status text with implicit intent has not been addressed in previous work. Park *et al.* [32] studied mobile app retrieval problem given explicit queries, but we focus on implicit queries. Baeza-Yates *et al.* [5] studied next mobile app prediction problem based on a user's spatio-temporal contexts. Their work is different from our work since they do not analyze the user's direct input such as a user status text.

Retrieving mobile apps given a user status text with implicit intention is challenging. First of all, social media text is notoriously noisy [21] and short. Twitter limits user status text to 140 characters at each time, so it is harder to automatically understand tweets than Web documents, which are usually much longer. Second, a user status text q may not have enough similar text in our parallel corpora, which we build to infer user intent. Third, even though there exist enough similar user status texts in the corpora, the hidden intent may be different depending on the users. For example, when a query is "i am hungry", some people might mean "i want a recipe book" while others might mean "i want to find nearby restaurants." Lastly, descriptions of apps are often written in the app developer's language while social media text is often written in more general and informal language, resulting in vocabulary gap.

4. METHODS

In order to retrieve mobile apps relevant to a user's intention, we build parallel corpora leveraging social media. The overview of our approach is depicted in Figure 1. When a user implicitly expresses user needs in a user status text q, we search for implicit intention text similar to q in the parallel corpora. Then, we infer user intention from the explicit intention text that corresponds to the implicit intention text. Finally, we retrieve mobile apps that are relevant to the inferred intention.

We follow language modeling approach to perform the mobile app retrieval task. In this paper, we focus on how to expand the original query language model to capture user intention in the user status text. Thus, our goal is to estimate the query language model $p(w|q)$, which is defined as

$$p(w|q) = (1-\gamma)p_{ml}(w|q) + \gamma p(w|I_q)$$
$$p_{ml}(w|q) = \frac{count(w,q)}{\sum_{w'} count(w',q)} \quad (1)$$

where w is a word, $p_{ml}(w|q)$ is a maximum likelihood (ML) estimation of the word w being in the query, $p(w|I_q)$ is the intention language model for q whose weight is γ, and $count(w,q)$ is the number of w's occurrences in q. Thus, the query language model can be regarded as a mixture model of the ML-estimated query language model and the intention language model. We define γ as the confidence on our intention language model. That is, if we are not confident on the estimated intention language model, we assign more weight on $p_{ml}(w|q)$. Although we build parallel corpora to "translate" implicit intention text into explicit intention text, the inferred intention may not be correct for various reasons, which are discussed in Section 3. In such situations, γ can help us adjust our confidence level on the intention language model.

Once we estimate the query language model $p(w|q)$, we retrieve mobile apps relevant to the language model with KL-divergence retrieval model with Dirichlet prior smoothing as presented in [42]. The model is believed as one of the state-of-the-art ad-hoc retrieval models and it can naturally adopt query language model, so we choose it to retrieve mobile apps. The score function to score an app a with respect to q is thus defined as

$$score(a,q) = \left[\sum_{w\in a} p(w|q)\log\frac{p_s(w|a)}{\delta_a p(w|\boldsymbol{A})}\right] + \log\delta_a \quad (2)$$

where a word w's smoothed probability $p_s(w|a)$, a background language model $p(w|\boldsymbol{A})$, and the coefficient δ_a are defined as

$$p_s(w|a) = \frac{|a|}{|a|+\tau}\cdot\frac{count(w,a)}{|a|} + \frac{\tau}{|a|+\tau}\cdot p(w|\boldsymbol{A})$$
$$p(w|\boldsymbol{A}) = \frac{count(w,\boldsymbol{A})}{\sum_{w'} count(w',\boldsymbol{A})} \quad (3)$$
$$\delta_a = \frac{\tau}{\sum_w count(w,a)+\tau}$$

where $|a|$ is the length of a's text representation, and τ is the Dirichlet prior smoothing parameter. By retaining only the highest probability words in the query language model and re-normalizing it, it can process a query very efficiently [41] with inverted index since only apps containing a query language model word are considered in the formula (2). We keep the top 50 words with highest probabilities in the query language model and re-normalize the probability distribution as in the literature [36]. In the rest of this section, we describe the steps of estimating the intention language model $p(w|I_q)$.

4.1 Building Parallel Corpora From Social Media

Measuring relevance directly between a query and mobile apps may not be ideal when the query does not explicitly reveal the user intent. We need to understand what users truly want by writing their status text. Therefore, we attempt to "translate" the user's implicit intent text into explicit intent

text. Our key idea is to leverage parallel corpora that contain texts with implicit intent and their corresponding texts with explicit intent, in order to infer user intention hidden in the status text. Thus, a user query with implicit intent is matched with similar text in the parallel corpora, which provide us their corresponding explicit intent texts.

To build such parallel corpora, we employ text data in social media. There exist other useful resources to build parallel corpora such as chat logs, movie scripts, and question and answering data. However, chat logs are generally not accessible from public, movie scripts are limited in their amounts, and question and answering data focus more on general knowledge instead of people's intention. On the other hand, a myriad of user status texts are updated at social media. Not all of them contain people's needs, but even a small portion of all user status texts are plentiful. In addition, due to the nature of social media such as Twitter, the user status texts are accessible from public. Moreover, because people often leave their "current status" through social media, their contents can be applied to the task well since we want to infer a user's immediate needs. Therefore, we choose to employ social media text data to build parallel corpora although they are often very noisy.

We employ a template-based approach to build our parallel corpora for two main reasons. First, there are much more social media texts that do not contain both implicit intention and explicit intention than those containing them, so we need to filter them out with templates. Second, if we do not split social media texts into implicit intention text and explicit intention text, the social media texts found for a query q will have their language model dominated by all the words in q. However, we want the language model to explain the hidden intent, which may have quite different vocabulary than q. The experiment results in Section 6.2 support that we need higher weight on the intention language model, which is derived from the explicit intention texts, than on the original query language model.

Thus, we aim at finding social media texts that contain both implicit and explicit intention texts with templates. Since finding texts with implicit intention is more difficult than finding those with explicit intention, we plan to find texts with explicit intention and then find texts containing both of them among the texts found. We observe that people often use words such as "want" and "need" to explicitly express their needs in social media, so we employ such words to find texts with explicit intention. In order to find texts containing both implicit and explicit intention texts, we pay attention to multi-clause sentences that can accommodate two different predicates. Among the subordinating conjunctions, which connect clauses in a sentence, we notice that those for cause and effect relationship can be quite useful for our task. A user's status and the user's explicit need caused by the status are closely related to our task because this kind of relationship is what we want to capture in order to translate a user's status text into the user's needs.

Therefore, we make templates such as "i want <EXP> because <IMP>" where <EXP> is explicit intention text and <IMP> is implicit intention text. For example, a user status text "i want to eat pizza because i am hungry" matches our template with the implicit intention text being "i am hungry" and the corresponding explicit intention text being "to eat pizza". We also use other words than "want" in the template, such as "need", "should", and "wanna". To ensure high precision, we force restrictions to the matching sentences. For example, we don't let punctuation comes between <EXP> and <IMP>, and we always require sentences starts with "i". Finally, the texts in <EXP> and <IMP> of each user status text become documents in $\boldsymbol{D}^{exp}$ and $\boldsymbol{D}^{imp}$, respectively, while their association is preserved.

In the experiments, we use the above relatively simple words and templates to build high-precision parallel corpora, since the focus of the paper is not on the template generation but on the whole pipeline of mobile app retrieval based on the user's status text. The experimental results in Section 6 demonstrate the effectiveness of the proposed approach despite the simplicity of templates. In future work, we will investigate the bootstrapping approach [2] to automatically generate more diverse patterns from basic templates. Since we simply match templates with social media text, building parallel corpora can be efficiently done and linearly scalable with respect to the size of social media text. In addition, it is a one-time effort and can be done offline.

4.2 Finding Similar Implicit Intention Text

We follow information retrieval approach to match user status text with its similar implicit intention texts in parallel corpora. We employ Query Likelihood retrieval model [34] with Dirichlet prior smoothing [42], which scores an implicit intent document d^{imp} with respect to q with formulas:

$$
\begin{aligned}
score(q, d^{imp}) &= \sum_{w \in d^{imp}} count(w, q) \log p(w|d^{imp}) \\
&\propto \sum_{w \in q \cap d^{imp}} count(w, q) \log \frac{p_s(w|d^{imp})}{\delta_{d^{imp}} p(w|\boldsymbol{D}^{imp})} + \log \delta_{d^{imp}} \\
p_s(w|d^{imp}) &= \frac{|d^{imp}|}{|d^{imp}| + \omega} p_{ml}(w|d^{imp}) + \frac{\omega}{|d^{imp}| + \omega} p(w|\boldsymbol{D}^{imp}) \\
p(w|\boldsymbol{D}^{imp}) &= \frac{count(w, \boldsymbol{D}^{imp})}{\sum_{w'} count(w', \boldsymbol{D}^{imp})}
\end{aligned}
\tag{4}
$$

We employ this model since it is one of the standard models, and it is relatively easy to tune the parameter. In addition, this model can process a query very efficiently (as efficient as vector space models) with the inverted index. From the retrieved documents, we keep only top F documents for both efficiency and effectiveness, yielding $\boldsymbol{D}_q^{imp}$. Then, the corresponding explicit intent documents $\boldsymbol{D}_q^{exp} = \{d_1^{exp}, ..., d_F^{exp}\}$ are used to infer user intent in the next step. Please note that this process is similar to pseudo relevance feedback approaches in the literature, but it differs in that it eventually uses the corresponding explicit intent documents instead of the retrieved documents.

4.3 Inference with Intention Topic Modeling

In order to infer the intention language model $p(w|I_q)$ from $\boldsymbol{D}_q^{exp}$, we propose to employ intention topic modeling. That is, we first model various user intentions in $\boldsymbol{D}^{exp}$ as a pre-processing step, and then, we infer the intentions in q using the intention topic models. Employing such intention topic modeling gives us several benefits. First, we can understand various intentions in a given query. The same query may have different intentions depending on user context, and the intention topics can help us understand such different intentions by inferring multiple intentions. Second, we can remove noisy topics, which impair our intention language model because social media text data are inevitably very noisy. We can exclude intention topics that do not occur enough in $\boldsymbol{D}_q^{exp}$, which are likely noisy topics. Third, the inferred user intentions can be understood by humans in a more straightforward way. If the resulting intention language model contains multiple intentions or much noise, humans may not understand well what the language model describes. However, by inferring intention topics and removing noise in $\boldsymbol{D}_q^{exp}$, intention topic modeling can provide a list of intentions in q, which humans can understand more easily.

We first pre-process $\boldsymbol{D}^{exp}$ to model general user intentions by topic modeling approach, Latent Dirichlet Allocation (LDA) [7], with parameters α, β, and K. The learned word probability distribution for each intention topic, $\hat{\boldsymbol{\phi}}_k$, is then given to our Intention LDA in order to infer intentions in $\boldsymbol{D}_q^{exp}$.

The generative story of $\boldsymbol{D}_q^{exp}$ in Intention LDA is as fol-

lows. For each word of explicit intention text d, a user first chooses an intention $t_{d,i}$ according to the *query-level* intention distribution $\boldsymbol{\theta}_q$, which is drawn from a Dirichlet distribution with a symmetric vector $\boldsymbol{\alpha'}$. Then, the user chooses a word $w_{d,i}$ according to a word distribution $\hat{\boldsymbol{\phi}}_{t_{d,i}}$, which is pre-estimated from $\boldsymbol{D}^{exp}$ by LDA. This process is repeated for all words in $\boldsymbol{D}_q^{exp}$ for I iterations.

The conditional posterior distribution for $t_{d,i}$ is thus defined by Bayes rule as

$$p(t_{d,i}|\hat{\boldsymbol{\Phi}}, \boldsymbol{T}_{\backslash d,i}, \boldsymbol{\alpha'}) \propto p(w_{d,i}|t_{d,i}, \hat{\boldsymbol{\Phi}}) \cdot p(t_{d,i}|\boldsymbol{T}_{\backslash d,i}, \boldsymbol{\alpha'}) \tag{5}$$

where $\boldsymbol{T}$ is a set of all intention assignments in $\boldsymbol{D}_q^{exp}$, and "$\backslash d, i$" means excluding d's ith data. With the pre-processed intention topics, $p(w_{d,i}|t_{d,i}, \hat{\boldsymbol{\Phi}}) = p(w_{d,i}|\hat{\boldsymbol{\Phi}}_{t_{d,i}})$. The prior distribution for $t_{d,i}$ is defined for collapsed gibbs sampling as

$$p(t_{d,i}|\boldsymbol{T}_{\backslash d,i}, \boldsymbol{\alpha'}) = \frac{N^{\backslash d,i}_{t_{d,i}|\boldsymbol{D}_q^{exp}} + \alpha'}{N_{\boldsymbol{D}_q^{exp}} - 1 + K\alpha'} \tag{6}$$

where $N^{\backslash d,i}_{t_{d,i}|\boldsymbol{D}_q^{exp}}$ is the number of words in $\boldsymbol{D}_q^{exp}$ assigned with the intention $t_{d,i}$ excluding the ith word in d, $N_{\boldsymbol{D}_q^{exp}}$ is the number of all words in $\boldsymbol{D}_q^{exp}$, and K is the number of intention topics.

This model is different from a regular topic modeling approach in that it takes a small subset of all intention topics when it builds the intention language model, and we also take a unique smoothing technique, which is discussed in the next paragraph. In addition, unlike regular topic models, we estimate the intention distribution $\boldsymbol{\theta_q}$ at query-level instead of document-level. This is because our data set is expected to be very sparse. Both implicit and explicit texts consist of a few words, so our documents are much shorter than typical documents. This results in much less word co-occurrences, so we cannot reliably estimate the topics. Since we can assume that the retrieved texts in $\boldsymbol{D}_q^{exp}$ for q have the similar topic distributions, we employ the query-level estimation for $\boldsymbol{\theta_q}$, which means that $\boldsymbol{\theta_q}$ generates topics of words in all texts retrieved for q.

Building Intention Language Model.

Now, we can build the intention language model $p(w|I_q)$ with the estimations from the intention topic model.

$$p(w|I_q) = \sum_{t \in \text{Intentions}(q)} p(w|t, \boldsymbol{D}_q^{exp}) \cdot p(t|\boldsymbol{D}_q^{exp}) \tag{7}$$

where Intentions(q) is a set of candidate intentions for q. In order to remove noisy topics as discussed earlier in this section, we do not utilize all possible intention topics. We instead keep only top X intentions according to $p(t|\boldsymbol{D}_q^{exp})$, which is the likelihood of the intention being in the retrieved explicit intention text. Intuitively, a few top intentions for q can satisfy general users' needs, so we exclude the other intentions that are likely to be noisy. Also, by considering only a small subset of all intentions, we can build the language model more efficiently since we do not need to iterate over all intentions. The probabilities $p(w|t, \boldsymbol{D}_q^{exp})$ and $p(t|\boldsymbol{D}_q^{exp})$ are defined as

$$\begin{aligned} p(w|t, \boldsymbol{D}_q^{exp}) =& \frac{\hat{N}_{t|\boldsymbol{D}_q^{exp}}}{\hat{N}_{t|\boldsymbol{D}_q^{exp}} + \mu} \cdot p_{ml}(w|t, \boldsymbol{D}_q^{exp}) \\ &+ \frac{\mu}{\hat{N}_{t|\boldsymbol{D}_q^{exp}} + \mu} \cdot p(w|\hat{\boldsymbol{\Phi}}_t) \\ p(t|\boldsymbol{D}_q^{exp}) =& \frac{\hat{N}_{t|\boldsymbol{D}_q^{exp}} + \alpha'}{\hat{N}_{\boldsymbol{D}_q^{exp}} + K\alpha'} \end{aligned} \tag{8}$$

where $\hat{N}$ and $\hat{\boldsymbol{\Phi}}_t$ are estimations from the intention topic model and the regular LDA, respectively, μ is the smoothing parameter. $p(t|\boldsymbol{D}_q^{exp})$ is normalized to have its sum equal to one if $\boldsymbol{T_q}$ doesn't contain all possible intentions. Here, we smooth $p(w|I_q)$ in a topic-level. That is, for each topic, $p(w|t, \boldsymbol{D}_q^{exp})$ is smoothed with its posterior estimation from LDA by Dirichlet prior smoothing. Thus, its ML estimation, $p_{ml}(w|t, \boldsymbol{D}_q^{exp}) = \frac{\hat{N}_{w,t|\boldsymbol{D}_q^{exp}}}{\hat{N}_{t|\boldsymbol{D}_q^{exp}}}$, gets higher weight if more words are assigned to t. With such topic-level smoothing, we can expect two benefits. Firstly, we can dynamically smooth the intention topics depending on the number of assigned words. Intuitively, if more words are assigned with a topic t, we can more trust its ML estimation since we have more evidence about t in the text $\boldsymbol{D}_q^{exp}$. Secondly, we can "customize" each intention topic model for q. When a user status text q contains intentions that are not captured well by existing intention topics, the per-topic smoothing can help transform existing intention topics into new intention topics with the evidence present in the retrieved text. Please note that, in order to obtain reliable probability distributions, we run multiple Markov chains, and take the average values of the topic assignments.

Computational Complexity.

The complexity of the pre-processing steps, which is modeling user intentions with collapsed Gibbs sampling from $\boldsymbol{D}^{exp}$, is $O(I \cdot W \cdot K)$ where I is the number of iterations, W is the number of all words in $\boldsymbol{D}^{exp}$, and K is the number of intention topics. Therefore, it is linearly scalable with respect to the size of the social media text data ($\boldsymbol{D}^{exp}$). This pre-processing can be done offline, thus a one-time effort.

On the other hand, user intention inference is done online. For collapsed Gibbs sampling, the computation complexity is $O(I' \cdot W_q \cdot K)$ where I' is the number of iterations for inference, and W_q is the number of all words in $\boldsymbol{D}_q^{exp}$. The complexity of building intention language model with the inferred intentions is $O(K \log K + V \cdot T)$ including choosing top T intentions ($O(K \log K)$) and computing word distribution for T intentions ($O(V \cdot T)$), where V is the vocabulary size. In general, a small constant for I' (*e.g.*, 100) is good enough. Also, W_q is very small (*e.g.*, 875 words on average, which is a size of a couple of general Web documents) since $\boldsymbol{D}_q^{exp}$ contains a small subset of explicit intention texts. Also, $T \leq K$, so inferring user intention and building intention language model has an average complexity of $O(K) + O(K \log K + V \cdot T) = O(K(\log K + V))$. In general, the vocabulary size V sublinearly increases as the size of text data increases, so this inference process is sublinearly scalable with respect to the size of text data. As discussed in multiple places of this section, other components of the system are also scalable and perform efficiently, so the whole process is scalable and performed efficiently.

5. EXPERIMENTAL SETUP

5.1 Data Set

In order to perform experiments for the task, we need data sets such as mobile apps, social media data for parallel corpora, and test queries and their relevance data. We use mobile app data in [32], which crawled most popular apps for each category from Google Play App Store[4]. We omitted apps in game categories, which takes about 38% of the whole data, since our goal is to recommend more practical solutions while games are for entertainment in general. The resulting data set contains 26,832 apps in total. Although we take a subset of all available apps in Google Play App Store, the data set should cover most of the downloads by

[4]https://play.google.com/store/apps

users.[5] Since user status text is often written in an informal way by people in general, we also use apps' user review texts, which are also often written in an informal way. On average, there are 31.4 reviews for each app. The experiment results in this work will be based on concatenated text of app descriptions and user reviews unless otherwise specified. We tokenized text into word tokens and lemmatized them using Stanford CoreNLP [28] version 1.3.5. We lowered all word tokens and removed punctuation, stopwords, and word tokens that appear in less than five app descriptions (and five user reviews if reviews are also used). More statistics of the resulting data are shown in Table 1. Note that about 30% of vocabulary is omitted when we add user reviews; this is because many words in app descriptions are indeed not used by users, and vocabulary used by app developers is different from that used by general users [32].

Table 1: Statistics of text data in apps without reviews and with reviews.

	without reviews ($\neg \boldsymbol{R}$)	with reviews ($\boldsymbol{R}$)
Avg. # of tokens	116.5	395.6
Total # of tokens	3,124,611	10,615,767
Vocabulary size	20,293	14,196

Since we study a new task that has not been studied before, there is no existing test collection or parallel corpora available to use. In the rest of this section, we describe how we collect such data in detail.

5.1.1 Collecting Tweets to Build Parallel Corpora

We already described how to build parallel corpora with social media text data in Section 4.1. Here, we describe how to collect tweets and the details of the data set. Among various social media, we decided to crawl the data from Twitter mainly because plenty of tweets are available to public. We searched for tweets containing our keywords using Twitter's search function. The query we used is, for example, *"i want" because until:YYYY-MM-DD*, which is supposed to search for tweets containing the word *because* and the exact phrase *i want* until the specified date. We crawled tweets dated between June 6, 2006 and November 4, 2015. Then, we removed tweets that do not satisfy the templates in Section 4.1, and cleaned the texts in the same way as the app text data are cleaned. We allowed only one (implicit text, explicit text) pair if there exist exactly the same pairs, to avoid noise. Finally, we obtained 1,609,894 (implicit text, explicit text) pairs from 1,115,948 unique Twitter users where implicit text contains 2.7 word tokens and explicit text contains 2.5 word tokens on average, and its vocabulary consists of 35,695 unique words.

5.1.2 Collecting Representative Queries

We need to obtain queries so as to evaluate whether the proposed model is indeed useful. It is, however, challenging to obtain "representative" user status text segments. We cannot simply rank tweet texts based on the number of tweets that contain exactly the same content or similar content in the whole tweet data set. People use different vocabulary to describe similar statuses. In addition, the resulting ranking may still contain many redundant themes. Thus, we employ topic model to first find main themes in tweets and choose a representative tweet from each theme. We crawled tweets containing a keyword "i" so that the retrieved tweets are likely to be about the user's own status. The crawled tweets are dated between June 20, 2015 and August 18, 2015, and after cleaning them, we obtained 960,874 tweets. With LDA, we modeled 50 topics and ranked tweets with KL-divergence scoring function in formula (2) where the query becomes each topic's language model. After removing nine topics that are spams, we asked a domain expert to extract the first user status segment that implicitly expresses user need, starting from the most relevant tweets, for each of the non-spam topics. We observed that implicitly expressed user needs are closely related to the user's mood, so we also added 12 mood words from [39]. To choose 12 mood words from them, we omitted the words that do not overlap with the existing queries and that occur less than 400 times in the tweets we crawled. Then, we made a query for those words with the template "i am <word>" or "i feel <word>", whichever occurs more frequently according to Google's exact phrase search. After removing nine queries that are discussed in the next section, we compiled 44 queries, each of which contains around 5.5 words on average.

The number of queries we collected is relatively small compared to hundreds of queries in large test collections such as TREC data sets. It is very expensive to create a large-scale data set for this new task. In this paper, we aim at conducting a pilot study to demonstrate the feasibility of app retrieval based on social status text. Despite a relatively small number of queries, they cover a wide variety of topics that are representative of the real-world user implicit intentions. In the future work, we will further validate the proposed approach by collecting and labeling more data over an extended period of time.

5.1.3 Collecting Query Relevance Data

Since labeling all the retrieved mobile apps from various models for each query is too expensive, we created a pool. In specific, for each query, we pooled together the top 20 retrieved apps from *each* of various retrieval models, including standard information retrieval models and our proposed model, with various parameter settings, in order to acquire enough apps that are most likely relevant. We then employed a crowd-sourcing service, CrowdFlower[6], to label the (query, app) pairs at affordable price. Each worker was asked to read a query and a corresponding app's name and description, and follow the link to the app store page if needed. Then, the worker was paid three cents to judge if the app satisfies the user need on three relevance levels (no satisfaction at all (0), makes sense with some context (1), and perfect satisfaction (2)).

To ensure the quality of judgments, we let each (query, app) pair be judged by three annotators, and we used "quiz" function[7] in CrowdFlower to remove users who do not score high enough on the quizzes we made. The three resulting judgments for each (query, app) pair were averaged to be used as a relevance score. We removed nine queries that retrieved less than five perfectly relevant apps (by majority vote) since they cannot reliably distinguish different methods. In total, 156 workers were employed through the crowdsourcing service, and each of them made 177.0 $\pm$ 117.5 judgments where the number after $\pm$ is standard deviation. After all, we obtained relevance data for 7,920 (query, app) pairs, where there are 180 $\pm$ 59.5 relevance data on average for each query. We measured the inter-annotator agreement by Fleiss' kappa, which was 0.31, where the value between 0.21 and 0.4 can be interpreted as fair agreement according to [23].

5.2 Evaluation Metrics

The aggregated relevance judgment ranges between 0 and 2. Thus, we employ Normalized Discounted Cumulative Gain (NDCG) [19] as the evaluation metric since it can handle multiple-level relevance data while metrics such as Mean

[5]The top 1% and 10% of most downloaded apps accounts for over 78% and 96% of the total downloads, respectively [37].

[6]http://www.crowdflower.com/

[7]A worker's judgments are stored only if the worker inputs 6 correct answers from the initial quiz with eight questions and maintains the accuracy above 75% afterwards with random quizzes.

Average Precision cannot. We measure NDCG at top 3, 5, 10, and 20 retrieved apps to reflect information needs of various users. Note that NDCG@3 is more important for this task than for traditional Web search since only a few mobile apps can fit a mobile phone screen well, and many social media users use their mobile phones to connect to social media. We ignore some possible unjudged apps as in [32]. We employ student's two-tailed paired t-test ($p<0.05$) to obtain statistical significance.

5.3 Baseline Methods

We have two types of baseline methods: models that do not leverage parallel corpora and models that do. The models not leveraging parallel corpora include Query Likelihood Language Model (**QL**) and **Relevance** model [25], which are standard language model-based information retrieval methods. We use Query Likelihood Language Model with Dirichlet prior smoothing as described in Section 4.2 where we score an app a instead of d^{imp}. Since our Intention model expands the original query with online processing, we employ Relevance model (Model 1) as a baseline to be fair, which is one of the state-of-the-art query expansion methods with online processing. Relevance model first retrieves apps with original query using QL. Then, it expands the original query with the descriptions of the top F_a retrieved apps.

The models that leverage parallel corpora are closely related to cross-lingual information retrieval models [6] because parallel corpora are usually exploited to translate between two different languages. We employ **Translation** (or **Trans.**) model and Cross-Lingual Relevance Model (**CL-Relevance** or **CL-Rel.**) [24]. Translation model employs IBM Model 1 [9] to estimate word translation probabilities from $\boldsymbol{D}^{imp}$ to $\boldsymbol{D}^{exp}$. Then, it builds the query language model by

$$p(w|q) = \sum_{w' \in V(\boldsymbol{D}^{imp})} tr(w|w')p(w'|q) \quad (9)$$

where $V(\boldsymbol{D}^{imp})$ is a vocabulary set in $\boldsymbol{D}^{imp}$, and $tr(w|w')$ is the translation probability from w' to w. Thus, the resulting query language model would consist of words in $\boldsymbol{D}^{exp}$ that are semantically associated with the original query. CL-Relevance is an extension of the Relevance model and is one of the state-of-the-art cross-lingual information retrieval models. Similar to our **Intention** model, CL-Relevance leverages the parallel corpora to retrieve relevant intention texts from $\boldsymbol{D}^{\boldsymbol{exp}}$ given a query as in Section 4.2, and then, it estimates the query language model from the top F intention texts.

Translation and CL-Relevance models are designed for cross-lingual information retrieval. However, our task involves only one language although we built the parallel corpora with $\boldsymbol{D}^{imp}$ and $\boldsymbol{D}^{exp}$, which means that we can incorporate the original query into the estimated query language model since they are made of the same language. We interpolate the original query language model with the estimated query language model with a fixed coefficient γ as in formula (1), where the estimated query language model from Translation or CL-Relevance becomes the intention language model. We call the resulting models for Translation and CL-Relevance models as **Translation+ORIG** (or **Trans.+O**) and **CL-Relevance+ORIG** (or **CL-Rel.+O**), respectively.

5.4 Parameter Setting

We tune the parameters as much as possible with a data set containing both app descriptions and user reviews, and we use the data set unless otherwise specified. The parameter values are shown in Table 2, and we use those values in this work unless otherwise specified. To model topics in $\boldsymbol{D}_q^{exp}$ with LDA for Intention model, we run 1,000 Gibbs sampling iterations. We use three Markov chains with 100 Gibbs sampling iterations each for Intention LDA. We use GIZA++ package [31] for Translation model, where we use IBM Model 1 with default parameter values.

Table 2: Parameter setting for QL (Q), Relevance (R), Trans. (T), Trans.+ORIG (TO), CL-Rel. (C), CL-Rel.+O (CO), and Intention (I) models.

	Value	Models
γ	0.9(TO), 0.8(CO, I)	TO, CO, I
τ	1,000	Q, R, T, TO, C, CO, I
ω	100	C, CO, I
F	350	C, CO, I
F_a	50	R
λ	0.8(R), 0.5(C, CO)	R, C, CO
α	0.01	I
β	0.01	I
K	300	I
α'	0.1	I
X	5	I
μ	5	I

6. RESULT ANALYSIS

6.1 Qualitative Analysis

Table 3: Top five intention topics by LDA. Each column contains top words for a topic.

room	show	eat	play	win
clean	watch	make	football	super
fridge	tv	chicken	team	seahawks
tidy	stop	cheese	player	bowl
house	netflix	soup	soccer	raven
living	reality	egg	basketball	bronco
mini	series	fries	game	ravens
closet	ellen	potato	baseball	49ers
lock	movie	bacon	sport	lose
bigger	program	mac	fantasy	simply

Table 3 shows top intention topics obtained from social media user's explicit intention text $\boldsymbol{D}^{exp}$. It seems that the intentions are closely related to people's everyday life such as cleaning room, watching TV shows, eating or making food, playing sports, and supporting sports teams. For each query, we infer the most related intentions, and the top intentions for several queries are shown in Table 4. One of the advantages of our Intention model is that it is easier to understand different user intentions for a user status text than other baseline models. Interestingly, for a query "i feel sleepy", Intention model infers diverse intentions such as "sleep" (first), "drink coffee" (second), "take a nap" (third), and "sleep earlier" (fourth). These intentions indeed make sense for the user status, and services related to the inferred intentions can be recommended appropriately. On the other hand, there also exist some queries that do not have various inferred intentions. For example, Intention model infers intentions that are very similar to each other for a query "i am hungry"; the intentions are mostly about eating food. Nonetheless, the inferred intentions make sense, and such intentions can be definitely applied to recommender systems to understand a user's hidden intention in a user status text. One important thing to note is that even though user intentions are inferred well, there may not exist mobile apps that satisfy the inferred intentions. For example, an intention "stop buying things" (second) is inferred for a query "i spend way too much money". However, we could not find relevant mobile apps using either the intention language model or the query words since mobile apps are barely able to directly stop people buying things. Perhaps, we may need another layer of translation from an intention language to

Table 4: Top five intentions inferred from our Intention model for each query. Each intention is represented by its top five words.

query	i'm getting sad	i feel sleepy	i feel lonely
top intentions	happy, make, feel, smile, sad	sleep, back, wake, day, home	friend, make, internet, talk, guy
	hug, give, kiss, big, nus	coffee, drink, energy, cup, caffeine	buddy, cuddle, friend, texting, text
	listen, stop, song, radio, station	nap, stop, home, school, work	watch, movie, stop, film, cry
	listen, music, stop, play, ipod	sleep, bed, start, earlier, early	follow, fan, account, friend, 5so
	life, people, thing, positive, stop	late, stay, stop, night, sleep	hug, give, kiss, big, nus
query	i got a bad feeling about today	i spend way too much money	i am hungry
top intentions	start, people, listen, advice, hang	shopping, store, grocery, shop, work	eat, start, breakfast, dinner, lunch
	god, pray, jesus, lord, prayer	stop, online, buy, thing, stuff	food, eat, stop, make, bring
	cry, tear, joy, happiness, happy	money, job, give, make, lot	eat, stop, food, hungry, bore
	medicine, sleep, doctor, med, pain	card, credit, gift, buy, give	eat, make, chicken, cheese, soup
	cry, ball, curl, bed, die	job, find, asap, pay, good	pizza, eat, order, work, food

an app function language so that the intention "stop buying things" is translated into an appropriate text such as "budget manager".

Table 5: Top query language model words and their probabilities estimated from each model for a query "i'm pretty tired after work today". MLE indicates maximum likelihood estimation of the query.

MLE		Trans.+O		CL-Rel.+O		Intention	
pretty	0.25	work	0.11	sleep	0.26	sleep	0.34
tire	0.25	pretty	0.08	bed	0.09	bed	0.13
work	0.25	day	0.07	work	0.08	nap	0.06
today	0.25	today	0.07	today	0.07	work	0.06
-	-	tire	0.06	pretty	0.07	today	0.06
-	-	tomorrow	0.05	tire	0.07	tire	0.05
-	-	sleep	0.04	stop	0.06	pretty	0.05
-	-	feel	0.04	play	0.05	plan	0.03
-	-	hour	0.02	friend	0.02	start	0.02
-	-	week	0.02	hour	0.01	back	0.02

Table 5 shows the query language models estimated by different methods for a query "i'm pretty tired after work today." While MLE gives the same probability to each keyword of the query, the other models expand the query language model with *our* parallel corpora. The language model estimated by Trans.+O model does not seem to highlight words that are important particularly for the query. As discussed in the next section, the intention language model estimated by Trans.+O model is in more general context. For example, words such as "pretty", "tomorrow", "hour", and "week" do not have to be emphasized much, and words such as "sleep" and "bed" need to be emphasized more for the query. CL-Rel.+O model assigns more weights to such words, and Intention model assigns even more weights to them, so they retrieve more relevant apps. Table 6 shows the top retrieved apps for the same query. QL, which assigns the same weight to each query word as MLE does, retrieves many irrelevant mobile apps such as "Wheel Tire Calculator", "Tires Plus", "Pregnancy Workout Today", and so on. Trans.+O model retrieves apps such as "Horoscope for Facebook", "Squats", "Fetch: Your Buying Assistant", and so on, which do not seem to satisfy the user intention in the query, and this can be expected from its poorly estimated query language model. Most apps retrieved by CL-Rel.+O and Intention models are satisfying since they could build the query language models well as shown in Table 5.

6.2 Quantitative Analysis

Table 7 compares Intention model with models that do not leverage our parallel corpora, and it also shows measures when reviews ($\boldsymbol{R}$) are used and not used. Relevance model indeed outperforms QL with its query expansion from the app data set. Intention model significantly outperforms QL and Relevance models because it can infer user intention with our parallel corpora. Intention model outperforms the other models especially for NDCG at 3 and 5, which is desirable for mobile environment. For the task, exploiting user reviews is shown to be beneficial, which is similar to the findings in [32] for general app search task. Also, when reviews are used, Intention model's percentage improvement over Relevance model increases from 19.2% to 26.5% on average. Intuitively, while app descriptions are written by app developers, user reviews and user status texts are written by ordinary people with similar vocabulary, so the effect of exploiting parallel corpora can be amplified by using user reviews.

Table 7: NDCG measures for baseline models and Intention model. QL and Relevance models do not leverage parallel corpora while Intention model does. (% imp.) indicates Intention model's percentage improvement over Relevance model. The symbols * and $^\circ$ indicate statistical significance over QL and Relevance, respectively.

Data	Model	N@3	N@5	N@10	N@20
$\neg\boldsymbol{R}$	QL	0.404	0.420	0.415	0.418
	Relevance	0.437	0.430	0.441	0.461
	Intention	**0.531**$^{*\circ}$	**0.534**$^{*\circ}$	**0.527**$^{*\circ}$	**0.515***
	(% imp.)	+21.5%	+24.2%	+19.5%	+11.7%
$\boldsymbol{R}$	QL	0.445	0.438	0.440	0.438
	Relevance	0.463	0.460	0.444	0.455
	Intention	**0.596**$^{*\circ}$	**0.585**$^{*\circ}$	**0.564**$^{*\circ}$	**0.560**$^{*\circ}$
	(% imp.)	+28.7%	+27.2%	+27.0%	+23.1%

Table 8: NDCG measures for models leveraging our parallel corpora. (% imp.) indicates Intention model's percentage improvement over CL-Rel.+O. The symbols * and $^\circ$ indicates statistical significance over Trans.+O and CL-Rel.+O, respectively.

Data	Model	N@3	N@5	N@10	N@20
$\neg\boldsymbol{R}$	Trans.	0.480	0.486	0.480	0.498
	CL-Rel.	0.376	0.374	0.363	0.386
	Trans.+O	0.468	0.472	0.472	0.474
	CL-Rel.+O	0.477	0.470	0.473	0.484
	Intention	**0.531**$^{\circ}$	**0.534**$^{*\circ}$	**0.527**$^{*\circ}$	**0.515**
	(% imp.)	+11.3%	+13.6%	+11.4%	+6.4%
$\boldsymbol{R}$	Trans.	0.483	0.481	0.478	0.482
	CL-Rel.	0.425	0.409	0.395	0.406
	Trans.+O	0.519	0.519	0.503	0.510
	CL-Rel.+O	0.526	0.513	0.510	0.507
	Intention	**0.596**$^{\circ}$	**0.585**$^{\circ}$	**0.564**$^{*\circ}$	**0.560**$^{\circ}$
	(% imp.)	+13.3%	+14.0%	+10.6%	+10.5%

Table 8 compares models leveraging *our* parallel corpora. Again, it seems that all the models except Translation model improve when reviews are used. CL-Rel. does not perform well, and this is because each text pair in our parallel corpora is relatively short and noisy so that the top retrieved text from the corpora may be corrupted with noise. We mix the original query language model and the query language model estimated from CL-Rel. by linear interpolation as discussed in Section 5.3. When it is mixed with the origi-

Table 6: Top retrieved apps from each model for query "i'm pretty tired after work today".

QL	Translation+ORIG	CL-Relevance+ORIG	Intention
Wheel Tire Calculator	Sleep Cycle	Sleep Cycle alarm clock	Sleep Better with Runtastic
Tires Plus	Sleep Cycle alarm clock	Sleep Diary Pro	Sleep Diary Pro
Pregnancy Workout Today	Sleep Hypnosis: Cure Insomnia	Sounds for Baby Sleep Music	Sleep Cycle
Tire Calculator PRO	Nature sounds to sleep	Sleep Analyzer	Sleep Cycle alarm clock
Pretty Calculator	Horoscopes for Facebook	Sleep Better with Runtastic	Sleep as Android Unlock
ForzaTune 5	Sleep Better with Runtastic	Sleep Cycle	Sounds for Baby Sleep Music
Verizon Roadside Assistance	Squats	Sleep Talk Recorder	Relax Melodies: Sleep & Yoga
Boxing Interval Timer	Fetch: Your Buying Assistant	Sleep as Android	Deep Sleep and Relax Hypnosis
7-Eleven, Inc.	SleepyTime: Bedtime Calculator	SleepBots - Sleep Cycle Alarm	Sleep Analyzer
Shut Up Button	Dog Licker Live Wallpaper FREE	Sleep as Android Unlock	Relax Music & Sleep Cycle

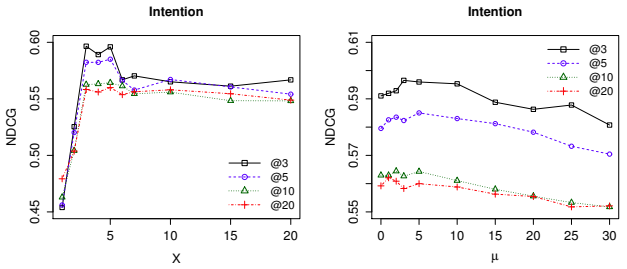

Figure 2: NDCG measures for different X (left) and μ (right) values of Intention model.

nal query language model as CL-Rel.+O, it improves quite much, which means that the original query language model and the estimated query language model should be incorporated for better performance. Translation model outperforms CL-Rel. as itself, which means that query expansion with the whole parallel corpora may be better than that with only the top retrieved texts from the parallel corpora, when the parallel corpora is relatively noisy. However, when Translation model is mixed with the original query language model as Trans.+O, it performs comparably with CL-Rel.+O. Our Intention model significantly outperforms all the other models in general. Although Intention model infers user intention from only top retrieved texts from the parallel corpora, it effectively removes the noisy intentions, yielding good results.

In order to filter out noisy intentions, Intention model uses only top X intention topics and estimates intention topics with per-topic smoothing. Figure 2 depicts Intention model with different number of top intention topics (X) values and different amount of smoothing (μ). The idea of removing unpopular intention topics, which are regarded as noisy topics, seems to yield better results indeed, especially for NDCG at 3 and 5. When we keep only one or two top intentions, Intention model does not perform well. However, its performance peaks when X is between 3 and 5, and then the performance degrades as it adds more intentions. Meanwhile, we can assign more weights to the intention topics pre-estimated from LDA by increasing μ value. The figure shows that adding a small amount (μ between 1 and 5) of the pre-estimated topics helps a little, but adding too much of them worsens the performance. From the results, it seems that removing noisy intentions plays a more important role than exploiting pre-estimated topics in estimating a good intention language model.

Trans.+O, CL-Rel.+O, and Intention models, which leverage parallel corpora, combine the original query language model and the intention language model, estimated from our parallel corpora, with a linear interpolation parameter γ as in equation (1). When $\gamma = 0$, the query language model becomes the original query language model $p_{ml}(w|q)$, and when $\gamma = 1$, the query language model is purely the intention language model from the parallel corpora. That is, when $\gamma = 1$, Trans.+O and CL-Rel.+O models become Translation and CL-Rel. models, respectively. Figure 3 depicts the models with different γ values. We can see that relying more on the intention language model results in better performance in general, which means that leveraging our parallel corpora is indeed important for the task. Intention model seems to outperform the other models with almost all γ values, even when the query language model comes entirely from the parallel corpora ($\gamma = 1$). Interestingly, all the models perform better when the original query language model is incorporated than when it is not. This means that the original query language model and the intention language model have their own strong points, so their combination makes even better results.

7. CONCLUSIONS AND FUTURE WORK

In this work, we studied the problem of mobile app retrieval for social media text that contains a user's implicit intention. Recommending mobile apps for such text can be very useful for both users and app developers, but no previous work has addressed this novel problem. We proposed how to build parallel corpora that can convert implicit intent text into explicit intent text. Then, we proposed the Intention model that leverages the parallel corpora to infer user intent to recommend satisfying mobile apps. Evaluation results show that (i) leveraging our parallel corpora built from social media is indeed beneficial, (ii) exploiting user reviews help us reduce vocabulary gap between users and app developers, (iii) removing noisy intentions plays an important role in Intention model, (iv) the original query language model and the estimated intention language model complement each other well, and (v) intentions inferred from Intention model help us understand various intentions in a straightforward way.

There are limitations in this work. When we collect the parallel corpora, we exploited templates to match texts with certain patterns. Those templates and words were manually compiled in order to ensure high precision. The templates do not ensure high recall since there may be many more templates we can exploit. When we evaluate the retrieved mobile apps from the suggested methods, we tested them with a relatively small number of queries since it is expensive to obtain query relevance data. More queries along with their relevance data would help us evaluate the task more reliably.

Our work can be further extended in several ways. First, while we recommended mobile apps, one can recommend entities in other domains, leveraging our parallel corpora. Second, while we studied recommending mobile apps for hidden intent of users, one can study preventing tragic events such as suicide and mass shooting by analyzing user text, because such users often implicitly reveal their intention. Lastly, since our parallel corpora made from social media are domain independent, one can leverage the corpora to other applications such as chat bots or virtual assistants to understand implicit intention.

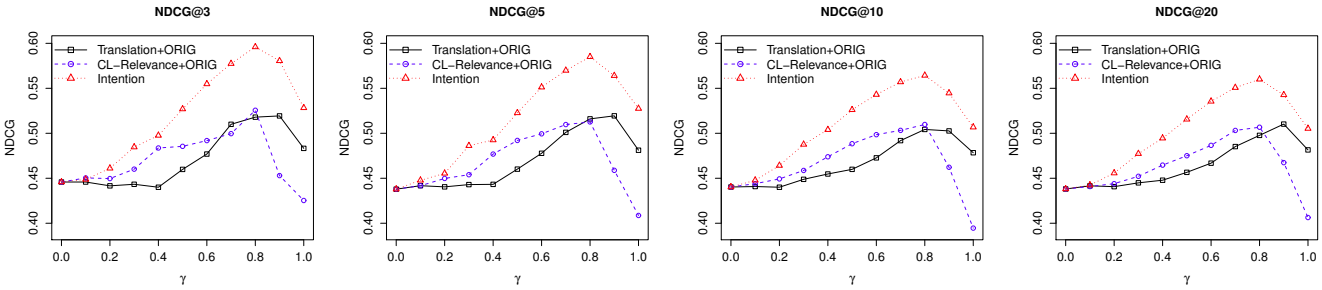

Figure 3: NDCG measures for different γ values of models that leverage the parallel corpora.

8. ACKNOWLEDGMENTS

This work is supported in part by a gift fund from TCL and by the National Science Foundation under Grant Number CNS-1027965.

9. REFERENCES

[1] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, and R. Passonneau. Sentiment analysis of twitter data. In *Workshop on Languages in Social Media*, pages 30–38. Association for Computational Linguistics, 2011.

[2] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *ACM conference on Digital libraries*, pages 85–94. ACM, 2000.

[3] A. Ashkan, C. L. Clarke, E. Agichtein, and Q. Guo. Classifying and characterizing query intent. In *Advances in Information Retrieval*, pages 578–586. Springer, 2009.

[4] R. Baeza-Yates, C. Hurtado, and M. Mendoza. Query recommendation using query logs in search engines. In *Current Trends in Database Technology-EDBT 2004 Workshops*, pages 588–596. Springer, 2005.

[5] R. Baeza-Yates, D. Jiang, F. Silvestri, and B. Harrison. Predicting the next app that you are going to use. In *WSDM*, pages 285–294. ACM, 2015.

[6] L. Ballesteros and W. B. Croft. Phrasal translation and query expansion techniques for cross-language information retrieval. In *ACM SIGIR Forum*, volume 31, pages 84–91. ACM, 1997.

[7] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[8] A. Broder. A taxonomy of web search. In *ACM SIGIR Forum*, volume 36, pages 3–10. ACM, 2002.

[9] P. F. Brown, J. Cocke, S. A. D. Pietra, V. J. D. Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85, 1990.

[10] J. Budzik and K. Hammond. Watson: Anticipating and contextualizing information needs. In *ASIS&T*, volume 36, pages 727–740. Citeseer, 1999.

[11] C. Carpineto and G. Romano. A survey of automatic query expansion in information retrieval. *ACM Computing Surveys (CSUR)*, 44(1):1, 2012.

[12] H. K. Dai, L. Zhao, Z. Nie, J.-R. Wen, L. Wang, and Y. Li. Detecting online commercial intention (oci). In *WWW*, pages 829–837. ACM, 2006.

[13] X. Ding, T. Liu, J. Duan, and J.-Y. Nie. Mining user consumption intention from social media using domain adaptive convolutional neural network. In *AAAI*, 2015.

[14] J. Duan, X. Ding, and T. Liu. Mining intention-related products on online q&a community. In *Social Media Processing*, pages 13–24. Springer, 2014.

[15] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964–971, 1987.

[16] Q. Guo and E. Agichtein. Ready to buy or just browsing?: detecting web searcher goals from interaction data. In *SIGIR*, pages 130–137. ACM, 2010.

[17] B. Hollerit, M. Kröll, and M. Strohmaier. Towards linking buyers and sellers: detecting commercial intent on twitter. In *WWW*, pages 629–632. International World Wide Web Conferences Steering Committee, 2013.

[18] B. J. Jansen, A. Spink, J. Bateman, and T. Saracevic. Real life information retrieval: A study of user queries on the web. In *ACM SIGIR Forum*, volume 32, pages 5–17. ACM, 1998.

[19] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *TOIS*, 20(4):422–446, 2002.

[20] I.-H. Kang and G. Kim. Query type classification for web document retrieval. In *SIGIR*, pages 64–71. ACM, 2003.

[21] M. Kaufmann and J. Kalita. Syntactic normalization of twitter messages. In *International conference on natural language processing, Kharagpur, India*, 2010.

[22] R. Kraft and J. Zien. Mining anchor text for query refinement. In *WWW*, pages 666–674. ACM, 2004.

[23] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, pages 159–174, 1977.

[24] V. Lavrenko, M. Choquette, and W. B. Croft. Cross-lingual relevance models. In *SIGIR*, pages 175–182. ACM, 2002.

[25] V. Lavrenko and W. B. Croft. Relevance based language models. In *SIGIR*, pages 120–127. ACM, 2001.

[26] U. Lee, Z. Liu, and J. Cho. Automatic identification of user goals in web search. In *WWW*, pages 391–400. ACM, 2005.

[27] H. Lieberman et al. Letizia: An agent that assists web browsing. *IJCAI (1)*, 1995:924–929, 1995.

[28] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *ACL*, pages 55–60, 2014.

[29] N. Naveed, T. Gottron, J. Kunegis, and A. C. Alhadi. Bad news travel fast: A content-based analysis of interestingness on twitter. In *Web Science*, page 8. ACM, 2011.

[30] D. W. Oard. A comparative study of query and document translation for cross-language information retrieval. In *Machine Translation and the Information Soup*, pages 472–483. Springer, 1998.

[31] F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

[32] D. H. Park, M. Liu, C. Zhai, and H. Wang. Leveraging user reviews to improve accuracy for mobile app retrieval. In *SIGIR*, pages 533–542. ACM, 2015.

[33] M. J. Pazzani and D. Billsus. Content-based recommendation systems. In *The Adaptive Web*, pages 325–341. Springer, 2007.

[34] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR*, pages 275–281. ACM, 1998.

[35] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *WWW*, pages 851–860. ACM, 2010.

[36] B. Tan, A. Velivelli, H. Fang, and C. Zhai. Term feedback for information retrieval with language models. In *SIGIR*, pages 263–270. ACM, 2007.

[37] N. Viennot, E. Garcia, and J. Nieh. A measurement study of google play. In *ACM SIGMETRICS Performance Evaluation Review*, volume 42, pages 221–233. ACM, 2014.

[38] J. Wang, G. Cong, X. W. Zhao, and X. Li. Mining user intents in twitter: A semi-supervised approach to inferring intent categories for tweets. In *AAAI*, 2015.

[39] D. Watson and A. Tellegen. Toward a consensual structure of mood. *Psychological bulletin*, 98(2):219, 1985.

[40] H. Yang and Y. Li. Identifying user needs from social media. Technical report, IBM Tech Report. goo.gl/2XB7NY, 2013.

[41] C. Zhai. Statistical language models for information retrieval. *Synthesis Lectures on Human Language Technologies*, 1(1):1–141, 2008.

[42] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR*, pages 334–342. ACM, 2001.

[43] X. Zhang, H. Fuehres, and P. A. Gloor. Predicting stock market indicators through twitter "i hope it is not as bad as i fear". *Procedia-Social and Behavioral Sciences*, 26:55–62, 2011.