Discovering Contextual Information from User Reviews for Recommendation Purposes

Konstantin Bauman,[1] Alexander Tuzhilin[1]
[1]Stern School of Business, New York University

Abstract

The paper presents a new method of systematically discovering relevant contextual information from the user-generated reviews and, therefore, converting the latent contextual information into the explicit representational view in order to provide better recommendations to the users when the reviews complement traditional ratings used in recommender systems. In particular, we classify all the user reviews into the "context rich" *specific* and "context poor" *generic* reviews and present a *phrase-based* and an *LDA- based* methods of extracting contextual information from the *specific* reviews. We also show empirically on the Yelp data that, these two methods extract almost all the relevant contextual information from the reviews across three different applications.

1 Introduction

The use of recommender systems (RSes) in the industry has exploded over the last several years making them mission-critical technologies in some of the companies. For example, about 80% of movie rentals streamed at Netflix were influenced by their RS, making it of strategic importance to Netflix [14]. Similarly, the whole business model of Stitch Fix relies on their recommender system [12]. One of the important factors affecting performance of RSes is the contextual information [6]. As the CEO of Netflix, pointed out, Netflix can improve performance of its RS up to 3% when taking into account such contextual information as the time of the day or location in their recommendation algorithms [17].

The field of Context-Aware Recommender Systems (CARS) has experienced extensive growth since the first papers on this topic have appeared in 2000's [5] when it was shown that the knowledge of contextual information helps to provide better recommendations in various applications [4], including music [15, 20], movies [8], restaurants [22] and hotels [2, 16].

One of the key issues in the CARS field is the question of what context is and how it should be specified. According to [3, 9], context-aware approaches are divided into *representational* and *interactional*. In the representational approach, context can be described using a set of observable contextual variables, such as *Company, Time-of-the-Day, Season*, and *Mood*, the structure of which does not change over time and *all* their values are known *a priori*. Alternatively the context can be modeled using latent and other approaches if it is not known a priori according to the interactional view of CARS [18].

Although treating this contextual information as latent is useful in many cases, trying to *convert* it to the representational approach by explicitly identifying all the relevant contextual variables, their structure and values in an application, constitutes an important problem because it reduces the uncertain latent approach to the well-established representational paradigm of CARS with all its benefits [7]. Proposing a new method of converting latent contextual information to the representational view is the main contribution of this work.

In this paper, we focus on the applications dealing with user reviews of various establishments, such as restaurants and hotels, and also having user-specified ratings of these establishments. Among other things, these reviews may also contain certain contextual information "hidden" in them, such as the fact that the user went to the restaurant for lunch with her family to celebrate her son's birthday. This contextual information is "latent" in the sense that it is embedded in the review and is not known in advance (unobserved) for the application. Therefore, it needs to be "discovered" in the reviews in order to be used subsequently in the popular representational method described above.

One way to discover these latent contexts would be to extract particular contextual information from user reviews using text-mining methods. This direct approach is difficult, however, because there are numerous (up to 150 [13]) alternative definitions of what contexts are. Therefore, we propose an alternative *indirect* method for discovering relevant contexts from the user-generated reviews that is based on separating these reviews into *specific* (those having rich contextual information) and *generic* (those that do not have such information).

This paper makes the following contributions. First, we propose two novel methods, a

phrase- and an *LDA-based,* of discovering contextual information from the user-generated reviews when contexts are not known in advance. Second, we validate them on the restaurants, hotels, and beauty&spas applications and show that these two methods perform well across all the three applications. Third, we show that most of this contextual information mentioned in user reviews can be discovered effectively and efficiently. These contributions are important because they show that our proposed method is viable and, therefore, makes the whole context-aware approach more practical in those applications that have user reviews.

2 Background and Prior Work

There has been much work done in recommender systems on identifying what context is and formalizing this multidimensional concept [7]. Most of the papers in this area assume the *representational* view [7] that was described above. In this paper we follow the representational approach, as described in [26] and [4], and define contextual information with *contextual variables* that can have a complicated structure reflecting the nature of this information. Although we can identify a contextual variable in an application, it may be hard to specify its structure and its values since this information can be idiosyncratic for the application. For example, contextual variable *Season* may be specified by a non-traditional categories of Winter and Summer, each of which are further subdivided into Holiday/Non-Holiday classes, which is more meaningful in a retailing domain than the classical taxonomy [27].

Since full specification of the contextual variables, including their structure and values, is idiosyncratic for a particular application, it is hard to automate this construction process. Therefore, at least parts of this specification should be done manually by the domain expert. We follow this principle in our context discovery method and leave some parts of the contextual variable specification process to the domain expert, as explained in Section 3.

The idea of extracting the contextual information in recommender systems from the user-generated reviews has been previously studied in [2, 16, 22]. In [2], authors identify sentences in the review that contain contextual information based on the classification and the text mining techniques. They applied their method to a hotel application and *Objective-of-trip*

Figure 1: Exampled of a *generic* (left) and *specific* (right) reviews.

contextual variable. Authors of [22] proposed a method of extracting *Companion, Occasion, Time,* and *Location* contextual variables in restaurant application based on NLP techniques. Finally, [16] used Labeled-LDA method for categorizing hotel reviews by their *Trip-type* contextual variable. All the three methods [2], [16], and [22] have one common assumption that *all the types* of contextual information are *a priori* known. Therefore, the main problem that these papers address is the determination of the *specific* values of the *known* contextual variables based on the reviews, whereas we focuses on *discovering* the contextual variables from the user reviews in those cases where the contextual information is not known *a priori*.

3 Method of Context Discovery

The key idea of the proposed method is to *discover* the contextual information from the user-generated reviews. However, not all the reviews contain rich contextual information. For example, *generic* reviews, describing overall impressions of a restaurant or a hotel, such as the one presented in Figure 1 (left), contain only limited contextual information, if any. In contrast, the *specific* reviews describing a particular visit to a restaurant may contain rich contextual information. For example, the review in Figure 1 (right) describes the specific dining experience at a restaurant. It contains such contextual information as the time of the visit ("lunch time"), with whom the person went to the restaurant ("bros"), and the date of the visit (7/19/2013). Therefore, the first step in our approach is to separate such *generic* from the *specific* reviews, and we present a separation method in Section 3.1.

After that, we use the *specific/generic* dichotomy to extract the contextual information from the user reviews using the two approaches based on (1) the identification of the context-related *phrases*, such as "lunch time," "wedding anniversary" or "groupon"; (2) the LDA method [10] that identifies context-related topics. By discovering such phrases and topics

we can identify the contextual variables in an application, as described in Section 3.4. These phrase- and LDA-based approaches are described in Sections 3.2 and 3.3.

3.1 Separating Reviews into Specific and Generic

The main idea of the method of separating *specific* from *generic* reviews lies in identification of certain characteristics that are prevalent in one type but not in the other type of reviews. For example, users who describe particular restaurant experiences tend to write long reviews and extensively use past tenses (e.g., "I came with some friends for lunch today"), while generic reviews tend to use present tense more frequently (e.g., "they make wonderful pastas"). In our study, we used the following features separating the *specific* and *generic* reviews: the number of sentences in the review; the number of words; the number of verbs; the number of verbs in the past tenses; *VRatio* – the ratio of the last two measures.

Given these characteristics, we apply the K-means clustering to separate the reviews into the *specific* and *generic* clusters. We describe the specifics of this method in Section 4.2.

3.2 Identifying context related phrases

The key idea of this method is to identify those phrases that occur with a significantly higher frequency in the *specific* than in the *generic* reviews. As explained earlier, *specific* reviews are more context rich and, therefore, many phrases describing the contextual information fit into this category. We capture them by analyzing the dichotomy between the patterns of words in the two categories of reviews as follows:

1. Produce a part of speech tagging of the reviews using the rule-based techniques [11].
2. Search only for the phrases matching the patterns: {*N, AN, NN, AAN, ANN, NAN, NNN, NPN*}, where N stands for "Noun", A for "Adjective," and P for "Pronoun," to avoid phrases with functional words. Based on [19] these patterns are likely to represent a certain syntactic meaning and, therefore, are good candidates for our phrases.
3. Use the five word collocation window [24] within each sentence to identify all the phrases matching to at least one pattern.

4. For phrase n_i calculate its frequencies $N_s(n_i)$, $N_g(n_i)$ in *specific* and *generic* reviews.
5. Select the frequent phrases occurring in more than α cases of the specific reviews, i.e. those having $\frac{N_s(n_i)}{R_s} \geq \alpha$, where R_s is the number of specific reviews. We set $\alpha = 0.5\%$.
6. For each phrase n_i, estimate the relative frequency with respect to the overall lengths of specific and generic reviews $p_s(n_i) = N_s(n_i)/W_s$ and $p_g(n_i) = N_g(n_i)/W_g$, where $N_s(n_i)$ ($N_g(n_i)$) is number of occurrences of phrase n_i in specific (generic) reviews, and W_s (W_g) is the sum of numbers of words in all specific (generic) reviews.
7. Select the phrases with $p_s(n_i) > p_g(n_i)$.
8. To check if the relative frequency $p_s(n_i)$ is significantly higher than $p_g(n_i)$, we test the hypothesis $H_0 : p_s(n_i) = p_g(n_i)$ versus $H_1 : p_s(n_i) \neq p_g(n_i)$, by calculating statistics: $z_i = \frac{p_s(n_i) - p_g(n_i)}{\sqrt{p \cdot (1-p) \cdot \left(\frac{1}{N_s(n_i)} + \frac{1}{N_g(n_i)}\right)}}$, where $p = \frac{p_s(n_i) \cdot N_s(n_i) + p_g(n_i) \cdot N_g(n_i)}{N_s(n_i) + N_s(n_i)}$. In order to reject the null hypothesis H_0 with 95% confidence level, we check if $z_i > 1.96$. Finally, we select only those phrases having significantly higher relative frequency in the specific reviews.
9. For each phrase n_i, determine $ratio(n_i) = \frac{p_s(n_i)}{p_g(n_i)}$ of its relative frequencies.
10. Sort the phrases by $ratio(n_i)$ in the descending order.

By running this procedure, we obtain the list of phrases that are sorted based on the *ratio* defined in Step 9 above. As explained earlier, it is expected that the phrases describing the contextual information will appear high in this list (we show this empirically in Section 5.2.5). We next extract the contextual information from this list, as described in Section 3.4.

3.3 Identifying context related LDA topics

The key idea of this method is identify those LDA [10] topics corresponding to the contextual information for an application. In particular, we proceed as follows:

1. Pre-process the reviews using the standard text analysis techniques by removing punctuation marks, stop words, high-frequency words, etc. [23].
2. Build the LDA model on the set of the *specific* reviews identified in Section 3.1.
3. Apply this LDA model to *all* the user-generated reviews (i.e., specific and generic) in order to obtain the distribution of topics T_i for each review R_i.

4. For each topic t_k from the LDA model, determine its weighted frequencies $w_s(t_k)$, $w_g(t_k)$ corresponding to the set of *specific* (S) and *generic* (G) reviews as: $w_s(t_k) = \frac{\sum_{i:R_i \in S} T_i(t_k)}{|S|}$, $w_g(t_k) = \frac{\sum_{i:R_i \in G} T_i(t_k)}{|G|}$, where $T_i(t_k)$ is the probability of topic t_k in T_i.
5. Filter out the topics t_k that have low overall frequency, i.e., $w(t_k) = \frac{|R_i:T_i(t_k)>\alpha|}{|S \cup G|} < \beta$, where α and β are a threshold values. In our case we set $\alpha = 0.01$ and $\beta = 0.005$.
6. For each topic t_k determine $ratio(t_k) = \frac{w_s(t_k)}{w_g(t_k)}$ of its weighted frequencies.
7. Filter out those topics with $ratio(t_k) < \gamma$. In our experiments we selected $\gamma = 1.0$.
8. Sort the topics by $ratio(t_k)$ in the descending order.

By running this procedure, we generate the list of LDA topics sorted by *ratio* defined in Step 6 above. Since contextual information is usually related to the specific user experiences, we expect that these contextual topics will appear high in the list. We can extract the contextual information from this list, as described in Section 3.4.

3.4 Discovering Contextual Variables

In Sections 3.2 and 3.3 we have obtained sorted lists of phrases and topics respectively and we presume that the phrases and topics pertaining to contextual information appear high in those lists and, therefore, can be easily identified and extracted from them. We show it empirically in Section 5.2.5. In this section, we describe the process of discovering the contextual variables from the lists of phrases and LDA topics, and subsequent creation of their structures used in the representational approach, as explained in Section 2.

First, we manually examining the list of phases or the LDA topics and identify the groups of phrases or topics having the same meaning in an application. For example, phrases "colleague", "friend", and "daughter" specify with whom the reviewer went for lunch in the restaurant application and, therefore, can be grouped together into the *Company* contextual variable. We also determine the values of contextual variables based on these phrases or topics, such as the "parent" value based on the phrases "father" and "mother."

After we have discovered the contextual variables via this grouping process, it is also important to determine their hierarchical structure described in Section 2. Note that we

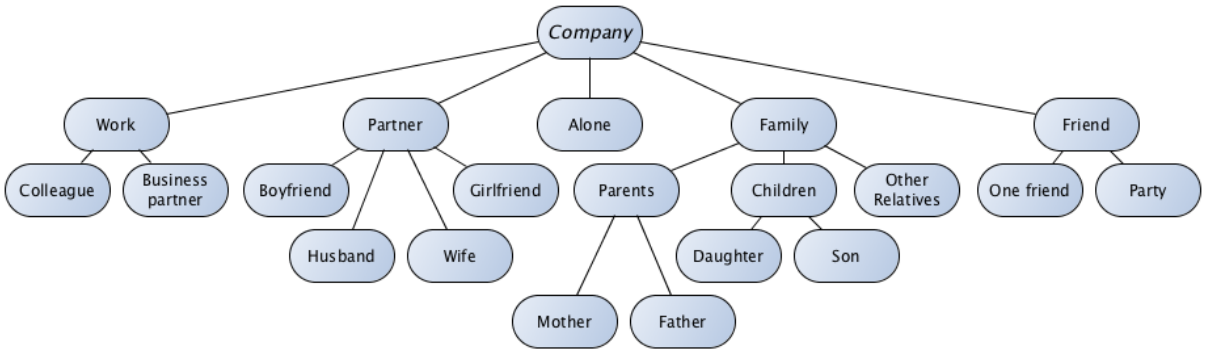

Figure 2: Hierarchical structure of contextual variable *Company*

do not address this problem in this paper because our main goal lies in discovering relevant contextual variables in an application. However, to complete the standard process of building contextual variables, the domain expert can manually identify their structures as follows.

First, one needs to form the "closure" of the set of the contextual values determined as described above, where the "closure" is the full list of possible values of the contextual variable associated with it. For example, we may add contextual values "boyfriend," "parent in law," "child," etc. to the contextual variable *Company*, specified above. Second, one needs to construct the hierarchical structure for the contextual variable from this identified set of values (i.e., the closure set). An example of such a hierarchical structure for the variable *Company* built from the Yelp reviews of restaurants is presented in Figure 2.

In summary, in Section 3 we presented (a) a method of separating the *specific* and *generic* reviews, (b) phrase- and LDA-based approaches to generating contextual values, and (c) a process of discovering contextual variables based on the obtained lists of phrases and topics.

4 Experimental Settings

We next test our methods on the Yelp data [1] to demonstrate how well it works in practice.

4.1 Yelp Dataset Descriptions

The Yelp dataset contains the user-reviews of various businesses, such as restaurants, bars, hotels, and also the user-specified ratings of these businesses. For instance, Figures 1 provide

Category	*Restaurants*		*Hotels*		*Beauty & Spas*	
Cluster	specific	generic	specific	generic	specific	generic
Number of reviews	168	132	195	105	173	127
Number of reviews with contextual information	146	25	127	13	103	9
% of reviews with contextual information	87%	19%	65%	12%	59%	7%

Table 1: Comparing *specific* and *generic* reviews in a random sample of selected reviews.

examples of restaurant reviews. In this study, we used all the reviews in the dataset for all the 21892 restaurants (990627 reviews), 1307 hotels (77497 reviews) and 4738 beauty & spas (64004 reviews) collected over the period of 6 years. We selected these three categories of businesses (out of 22 in total) because they differed significantly from each other.

4.2 Applying the Proposed Methods

We applied our context discovery method to the three applications described in Section 4.1.

4.2.1 Separation Process. First, we have separated all the reviews into the *specific* and *generic* clusters, as explained in Section 3.1. We used Scikit library [28] to build K-means clustering of the Yelp reviews. Furthermore, in order to determine how well this separation process works on the Yelp data, we manually labeled 300 reviews into *specific* vs. *generic* for each of the three applications used in this study based on whether any particular visit to the establishment was described in the review or it was a general review of it. The performance results of comparing the obtained clusters with these labels are reported in Section 5.1.

We have also counted the number of occurrences of contextual information in generic and specific reviews. The results presented in Table 1 support our claim that specific reviews contain richer contextual information than generic reviews across all the three applications.

4.2.2 Context Phrase and Topics Identification. We have applied the phrase-based method described in Section 3.2 to the Yelp data and obtained 315, 263 and 292 phrases in the sorted lists of phrases for restaurants, hotels and beauty&spas respectively. For example, the top of the phrases list generated based on the Yelp reviews for the restaurant application include { *"main course", "bill", "restaurant week", "disappointment", "second time", "last*

night"}. Some of these phrases, such as "restaurant week" or "last night", are clearly related to the contextual information and, therefore, can be used for discovering contextual variables.

We have also applied the LDA-based method described in Section 3.3 to the Yelp data with $m = 250$ topics for each of the three applications using *gensim* package [29], and produced 170, 136 and 141 topics for restaurants, hotels and beauty&spas applications.

Clearly, computational complexity of the phrase-based method is linear in the number of reviews. The LDA-based method is linear in the number of topics [21] and is polynomial in the number of reviews [31]. There is extensive research on how to speed up LDA and make it parallelizable and more scalable [30, 34]. Although there was no need to use any of these methods in our case, we can potentially deploy these performance enhancing LDA techniques on larger sets of reviews to speed up execution.

4.2.3 Discovering Contextual Variables. Section 3.4 described the method of how contextual variables are discovered by grouping phrases (or topics) with similar meanings. In this subsection we describe how this process is applied to the Yelp data.

For the restaurant reviews, we started with the list of 315 phrases produced using the phrase-based method described in Section 4.2.2. We, next, grouped some of these phases into 9 contextual groups, as described in Section 3.4. For example, for the *Day-of-the-Week* variable these would be {*Monday, Friday, Saturday, Sunday*} group of phrases. Based on this discovered group, the domain expert can complete the process of specifying the full structure of the contextual variable, i.e., by completing the generated group of phrases with additional related phrases, such as {*Tuesday, Wednesday, Thursday*}. As another example, we have also produced the following group of phrases {*friend, hubby, boyfriend, girlfriend, daughter, dad, mother, son, husband, wife, parents*} that falls into the category of the contextual variable *Company* specifying with whom the user visited the restaurant. An example of the hierarchical structure of contextual variable *Company* that could be constructed as described in Section 3.4 is presented in Figure 2. The structure of the *Company* variable is complicated and different domain experts can construct it differently based on their knowledge of the

application domain. However, this issue is *orthogonal* to the focus of this paper since our goal lies in discovering the contextual information from user reviews, and proper structuring of this information is left, eventually, to the domain expert.

In this manual part of the process of discovering contextual variables we had to examine the lists of phrases and topics and group them into particular contextual variables. The volume of work on the first part of this process depends on the number of phrases and topics in the lists generated by our method. Although, the numbers of reviews in some of our applications (hotels - 77,497; beauty & spas 64,004; and restaurants - 990,627) differ by more than 10 times, the number of phrases (and topics) in the generated lists do not differ very significantly, as shown in Section 4.2.2. The amount of work on grouping phrases or topics into contextual variables does not depend on the number of reviews in the system but depends on the number of phrases and topics in the generated lists and on the important contextual variables in the application. Therefore, the volume of this manual part of the work is limited and does not depend significantly on the total number of analyzed reviews.

4.3 Validation Procedure

In this study we use two methods for validating the quality of discovered contextual variables. First, we determine the "importance" of these contextual variables for rating prediction purposes. Second, we check the completeness of the discovered set of contextual variables by comparing it with the "ground truth" set of variables. These quality measures are closely related, but not identical, to the *precision* and *recall* measures used in information retrieval.

4.3.1 Precision. The Precision measure used in information retrieval is defined as the fraction of retrieved instances that are relevant. More formally, $Precision = \frac{TP}{TP+FP}$, where TP and FP are True Positive and False Positive values respectively [25]. In our case we define TP as the number of contextual variables discovered by our method *and* identified as being "important" for the rating prediction problem, while $TP + FP$ is defined as the whole number of contextual variables identified by the proposed method.

Although the "importance" of a contextual variable can be measured in several different

ways, in this paper we follow [4, 26] and [35] and measure this importance as follows. Assume contextual variable X is binary, i.e., it has only two values v_1 and v_2, such as "weekday" vs. "weekend," or "reservation" vs. "no reservation." To see if this variable "important," we examine all the ratings corresponding to the case of context being $X = v_1$ and compare how the distribution of these ratings differs from the distribution of ratings corresponding to the context being $X = v_2$. If the t-test shows that there is significant difference between these two distributions, then contextual variable X is "important" for the prediction purposes.

To generalize this approach from the binary to the general case, we assume that contextual variable X is *important* if there are at least two values v_1 and v_2 of variable X such that the distributions of the ratings corresponding to values v_1 and v_2 are significantly different in the sense explained above (i.e., the t-test results are positive). Note that this measure of "importance" of a contextual variable does not mean that the variable can actually lead to better recommendations, only that it affects the distribution of ratings in one way or another. However, as it was shown in [4] and [26], this measure is the necessary first step for selecting and using the contextual variables to provide better recommendations.

To apply this general approach to our particular case of Yelp user reviews, we need to learn the values of contextual variables from the text of the reviews. We followed the prior work on how to extract the values of contextual variables from reviews [2, 22] in our case and created rule-based classifiers for each variable in the list identified in Section 4.2.3.

As an example of constructing these rules for the restaurant case consider variable *Day-of-the-week* that has the values of Monday through Sunday. The rule for this variable states that if the review mentions a particular day of the week (either Monday, Tuesday, ..., or Sunday) in a single form and there is no adjective "every" before it, then we assign this particular day of the week (e.g., Monday) as the value of contextual variable *Day-of-the-week* for this review. We also ignore occurrences of names of the days in the well-known restaurants names, such as "Ruby Tuesday". However, as is shown below, this simple rule is quite robust since most of the reviews mention only one value of the *Day-of-the-week* contextual variable. Furthermore, by excluding the reviews having the pattern "every *",

	Context variable	*Precision*	*Recall*
1	Company	0.95	0.93
2	Time of the day	0.97	0.90
3	Day of the week	0.85	0.89
4	Advice	0.8	0.88
5	Prior Visits	0.84	0.91
6	Came by car	0.96	0.85
7	Reservation	0.93	0.83
8	Discount	0.87	0.9
9	Occasion	0.95	0.92

Table 2: The performance of context extraction rules in *restaurants* application.

we avoid the generic mentioning of the day of the week (such as "every Sunday").

We define similar rules assigning values to other contextual variables for the reviews across the restaurant, hotels and beauty&spas applications. On average we define 2-3 rules for each contextual variable. Further, for each contextual variable we estimate the performance of the developed rules using the *Precision* and *Recall* measures in a standard way where the "ground truth" comes from the manually labeled data. These performance measures are presented for all the contextual variables across the restaurant, hotels and beauty&spas applications in Tables 2, 3a and 3b respectively. As shown in Tables 2, 3a and 3b, the generated rules worked well in our applications despite the fact that they were not very complicated.

	Context variable	*Precision*	*Recall*
1	Company	0.94	0.95
2	Day of the week	0.87	0.92
3	Time of the year	0.98	0.96
4	Reservation	0.90	0.89
5	Came by car	0.86	0.92
6	Length of stay	0.93	0.88
7	Occasion	0.96	0.93
8	Airplane	0.93	0.91
9	Advice	0.79	0.82
10	Discount	0.89	0.80
11	Prior Visits	0.92	0.81

(a) *hotels* application

	Context variable	*Precision*	*Recall*
1	Company	0.86	0.83
4	Time of the day	0.88	0.91
8	Advice	0.81	0.85
6	Reservation	0.95	0.95
2	Day of the week	0.93	0.96
3	Prior Visits	0.90	0.92
7	Discount	0.89	0.88
5	Occasion	0.93	0.90
9	Came by car	0.95	0.85
10	Vacation	0.83	0.80

(b) *beauty & spas* application

Table 3: The performance of context extraction rules.

Once we associate the values of contextual variables with the reviews, we can compare

distributions of ratings for these variables using the t-tests method described above.

4.3.2 Recall. Recall is usually defined as $Recall = \frac{TP}{TP+FN}$, where TP and FN are True Positive and False Negative value respectively [25]. In our case we define TP as the number of contextual variables discovered using our method, while FN is the number of contextual variables that were missed by our method from among those discussed in the user reviews, that were identified "manually" within a random sample of n reviews. We selected $n = 400$ in our case because all the frequent contextual variables that appear in at least 1% of all the user reviews would appear in this random sample with the probability of at least 98%.

In addition, we calculated the *weighted recall* measure by using the sum of the weights of the contextual variables, where the weight of a variable is computed as its frequency in the sample of 400 user reviews selected as described above. This *weighted recall* measure captures the importance of discovered contextual variables vis-à-vis the missed ones.

5 Results

We first present the results of splitting Yelp reviews into *generic* and *specific* and then describe the discovery of contextual information from the reviews based on this split.

5.1 Separation Results

The results of separation of the user-generated reviews into the *specific* and *generic* clusters across the three applications are presented in Table 4 that has the following rows: *Size* – absolute and in % size of the specific and generic classes; *AvgSentences* – the average number of sentences in the reviews; *AvgWords* – the average number of words in reviews from the class; *AvgVBDsum* – the average number of verbs in past tense; *AvgVsum* – the average number of verbs; *AvgVRatio* – the average ratio of VBDsum and Vsum for reviews from the class; *AvgRating* – the average rating for reviews from the class; *Precision*, *Recall*, and *F1* measures for the class as described in Section 4.2.1; and the overall *Accuracy* of classification.

As Table 4 shows, the difference between *specific* and *generic* reviews is observed not only across the five parameters used to classify them, as described in Section 3.1, but also

Category	*Restaurants*		*Hotels*		*Beauty & Spas*	
Cluster	*specific*	*generic*	*specific*	*generic*	*specific*	*generic*
Size	556209	434418	58299	19198	37787	26217
Size (%)	56.1%	43.8%	75.2%	24.7%	59.0%	40.9%
AvgSentences	9.59	5.04	6.509	1.770	10.8	5.1
AvgWords	129.42	55.97	103.500	23.179	165.3	62.8
AvgVBDsum	27.07	1.09	6.283	0.499	11.1	0.9
AvgVsum	91.54	23.93	20.258	4.023	31.4	10.7
AvgVRatio	0.43	0.02	0.297	0.090	0.35	0.09
AvgRating	3.50	3.99	3.31	3.65	3.78	4.45
Precision	81.21%	83.79%	78.82%	88.16%	86.49%	80.50%
Recall	83.23%	81.10%	89.27%	76.55%	77.82%	87.67%
F1	81.75%	82.02%	83.28%	81.45%	81.30%	83.41%
Accuracy	82.20%		82.58%		82.72%	

Table 4: Clustering performance of *specific* vs. *generic* reviews.

the average rating (*AvgRating*) is significantly higher for the *generic* than for the *specific* reviews across all the three applications. Finally, the *Accuracy* of our separation described in Section 4.2.1 is $82.2\%, 82.5\%$ *and* 82.72% for restaurants, hotels and beauty&spas applications respectively. Although not perfect, these are good performance measures showing that our method separates the *generic* from the *specific* reviews well, thus allowing us to discover most of the contextual information, as demonstrated in Section 5.2.

5.2 Results of Contextual Variables Discovery

In this section we present our results of applying the proposed method to the Yelp reviews. But first, we present the list of all the contextual variables that we have obtained by manually inspecting a sample of reviews, as described in Section 4.3.2. We used this list as the "ground truth" to compute the performance of our discovery method, as explained below.

5.2.1 Identified Contextual Variables. Manual inspection of a sample of 400 *specific* reviews for each of the three applications identified all the contextual variables discussed in these reviews. For example, in the restaurants case we identified the following contexts:

- *Company:* specifying with whom the user went to the restaurant.
- *Time of the day:* information about the time of the day, e.g. morning or evening.
- *Day of the week:* specifying the day of the week (Monday, Tuesday, etc.).

- *Advice:* type of advice given to the user, such as a recommendation from a friend indicating that the user knows the opinions of other parties before going to the restaurant.
- *Prior Visits:* specifying if the user is the first time visitor or a regular in the restaurant.
- *Came by car:* specifying if the user came to the restaurant by car or not.
- *Occasion:* specifying the special occasion for going to the restaurant, such as birthday.
- *Reservation:* specifying if the user made a prior reservation in the restaurant or not.
- *Discount:* specifying if the user used any types of discount deals that he or she obtained before coming to the restaurant, such as groupon/coupon, a voucher or a gift certificate.
- *Traveling:* specifying if the user traveling and doesn't live in the city permanently.
- *Takeout:* specifying if the user did not stay in the restaurant but ordered a takeout.
- *Previous location:* specifying the location from where the user arrived to the restaurant.

Note that the analysis of these 400 reviews is an extensive, non-trivial and time consuming task because it involves careful reading, cognitive analysis and good understanding of the text, followed by reflection on which pieces of this text indeed constitute contextual variables and which do not. The main goal of our study is to automate this extensive and time-consuming process, which we have successfully done as demonstrated in the next section.

5.2.2 Discovered Contextual Variables. We extracted the contextual information from the *specific* reviews using the phrase- and the LDA-based methods. As shown in Section 4.2, we obtained the sorted lists of 315, 263 and 292 phrases for restaurants, hotels and beauty & spas categories using the phrase-based approach. We next identified the contextual variables in these three lists as described in Section 4.2.3.

The results are presented in Tables 5, 6 and 7, where the rows represent all the contextual variables identified in Section 5.2.1. *Frequency* column in those tables present the frequencies with which particular types of contextual variables appear in the *specific* reviews of the application that were calculated within manual inspection of the specific reviews discussed in Section 5.2.1. Further, the contextual variables discovered using our phrase-based method from Section 3.2 are marked with the check marks (✓) in Column *Phrase* in those tables.

	Context variable	*Frequency*	*Phrase*	*LDA*
1	Company	57.75%	✓(37)	✓(60)
2	Time of the day	49.75%	✓(45)	✓(11)
3	Day of the week	28.75%	✓(22)	✓(13)
4	Advice	14.0 %	✓(11)	✓(16)
5	Prior Visits	6.25%	✓(14)	✓(49)
6	Came by car	3.75%	✓(271)	✓(53)
7	Reservation	3.25%	✓(53)	✓(8)
8	Discount	3.0 %	✓(25)	✓(102)
9	Occasion	2.5 %	✓(58)	✓(9)
10	Takeout	1.75%	X	X
11	Traveling	1.25%	X	X
12	Previous location	0.75%	X	X

Table 5: Contextual variables in *restaurants* application (✓ marks discovered variables)

	Context variable	*Frequency*	*Phrase*	*LDA*
1	Company	41.75%	✓(7)	✓(21)
2	Day of the week	30.5 %	✓(5)	✓(34)
3	Time of the year	25.5 %	✓(155)	✓(80)
4	Reservation	19.25%	✓(33)	✓(23)
5	Came by car	16.5 %	✓(253)	✓(85)
6	Length of stay	8.0 %	✓(216)	X
7	Occasion	7.5 %	✓(64)	✓(8)
8	Airplane	5.75%	✓(40)	✓(120)
9	Advice	4.75%	✓(66)	✓(29)
10	Discount	4.5 %	✓(83)	X
11	Prior Visits	2.25%	✓(148)	✓(41)
12	City Event	1.0 %	X	X

Table 6: Contextual variables in *hotels* application (✓ marks discovered variables)

	Context variable	*Frequency*	*Phrase*	*LDA*
1	Company	26.5 %	✓(103)	✓(29)
2	Time of the day	19.25%	✓(21)	✓(7)
3	Advice	19.0 %	✓(2)	✓(5)
4	Reservation	18.25%	✓(77)	✓(4)
5	Day of the week	17.25%	✓(23)	✓(14)
6	Prior Visits	10.75%	✓(43)	✓(80)
7	Discount	9.75%	✓(108)	✓(11)
8	Occasion	5.0 %	✓(84)	✓(6)
9	Came by car	4.0 %	✓(147)	✓(42)
10	Vacation	2.75%	✓(148)	✓(123)

Table 7: Contextual variables in *beauty & spas* application (✓ marks discovered variables)

Also, the numbers in parentheses in the *Phrase* column appearing next to the check marks identify the occurrence of the phrase pertaining to the contextual variable that first appeared in the sorted lists of the phrases produced by the phrase-based method of Section 3.2.

Similarly, as explained in Section 4.2, we obtained the sorted lists of 170, 136 and 141 topics for restaurants, hotels and beauty & spas applications using the LDA-based approach. We also went through the three lists of topics, identified the contextual variables among them and marked these variables with the check marks in the LDA column in Tables 5, 6 and 7. As before, the number in parentheses in the table row next to the check marks (✓) also identify the first occurrence of the topic among the sorted lists of the topics produced by the LDA-based method that contains words pertaining to the contextual variable in that row.

As Tables 5 - 7 show, the phrase-based method identified 9 out of 12, 11 out of 12 and 10 out of 10 contextual variables for the restaurants, hotels and beauty&spas applications respectively. Similarly, the LDA-based method identified 9 out of 12, 9 out of 12 and 10 out of 10 contextual variables for the restaurants, hotels and beauty&spas applications. Note that the phrase based method discovered all the contextual variables that the LDA-based method did and a few additional variables that the LDA-based method missed.

5.2.3 Precision. We applied the method for computing precision, as defined in Section 4.3.1, to the obtained results, and determined that *all* the identified contextual variables are *important* across all the three applications, i.e., for each of the variables there exist at least two values, for which corresponding ratings have statistically significant differences. This means that the precision of our method is 100% across all the three applications.

As an example, the importance of contextual variables *Company*, *Time-of-the-Day*, *Discount* and *Reservation* for the restaurants application is examined using the data presented in Tables 8 - 10. These tables show different values and the corresponding average ratings of these contextual variables. *Support* column in those tables present the numbers of reviews in which our rule-based techniques discussed in Section 4.3.1 identified the corresponding value of the contextual variable. As Table 8 shows, most of the values of variable *Company*

Value	*Average Rating*	*Support*
boyfriend	3.61	17883
brother	3.6	3045
sister	3.56	4733
partner	3.56	83880
parent	3.55	6621
husband	3.55	35425
girlfriend	3.55	7768
wife	3.53	22804
family	3.52	83436
father	3.51	1175
colleague	3.5	688
friend	3.49	41187
mother	3.42	2618
kids	3.34	10808

Table 8: Values of contextual variable *Company*

Value	*Average Rating*	*Support*
morning	3.62	44339
afternoon	3.6	97251
evening	3.55	38812
night	3.54	96092

(a) *Time of the Day*

Value	*Average Rating*	*Support*
restaurant week	3.74	2418
None	3.51	539366
groupon	3.22	5687
coupon	3.2	7223

(b) *Discount*

Table 9: Values of contextual variables.

have significantly different rating averages. Similarly, the average rating of 3.62 for value "morning" differs significantly from the average rating of 3.54 for value "night" in Table 9a, which demonstrates that contextual variable *Time-of-the-Day* is "important" in our case.

5.2.4 Recall. We have applied recall computing method described in Section 4.3.2 to the obtained results and found that we have discovered 9 out of 12 variables (as shown in Table 5) using both the phrase-based and the LDA-based methods. This gives us the value of $recall = 75\%$ for the restaurant application. For comparison, [22, 32, 33] also use contextual variables for providing better recommendations in restaurant applications, and

Value	*Average Rating*	*Support*
no reservation	3.73	3400
reservation	3.66	15964

Table 10: Values of contextual variable *Reservation*.

they use 1, 3 and 4 contextual variables, respectively. In contrast, our method discover 9 contextual variables, thus significantly enriching known contextual information for the restaurant application. Table 6 shows the recall values of $recall = 11/12 = 91.67\%$ for the phrase-based and $recall = 9/12 = 75\%$ for the LDA-based methods in the hotels application. For comparison, [2, 16] also uses contextual variables for providing better recommendations in the hotel application and it identifies only one such variable, whereas we did it for 11. Finally, Table 7 shows $recall = 100\%$ for both the phrase- and the LDA-based methods.

Although our method managed to discover all the contextual variables in some of the applications, such as in beauty & spa, it missed some variables in other applications, such as restaurants, where it missed contextual variables *Takeout*, *Traveling* and *Previous-Location*.

The proposed method missed variables *Traveling* and *Previous-Location* because they (a) seldom appear in the reviews and (b) are described by fairly generic narratives that can be expressed in many different ways and, thus, do not have clearly identifiable contextual values. For example, it is hard to identify the value of the *Previous-Location* variable ("airport") in the following part of a review: "I was very tired from my morning flight from New York and, therefore, could not keep up with my colleagues during the lunch at restaurant X." We also could not discover the contextual variable *City-Event* in the hotels application for the very same reasons. Furthermore, our method could not identify the *Takeout* contextual variable in restaurants because the reviews pertaining to takeouts tend to be short, and our method often classified them as *generic* according to the clustering method described in Section 3.1.

Finally, as discussed above, *Frequency* column in Tables 5, 6 and 7 presents the frequencies of contextual variables in the *specific* reviews of the application. As you can see, both of our methods discover the *most frequently* occurring contexts, and the phrase-based method discovers all the contextual variables occurring in at least 2% of all the *specific* reviews.

Moreover, we have calculated the weighted recall measure, as described in Section 4.3.2. For restaurants, it is 97.8% for both the phrase- and the LDA-based methods. For hotels, it is 99.4% for the phrase-based and 91.9% for the LDA-based methods. Finally, it is 100% for the beauty & spa application for both the phrase- and the LDA-based methods.

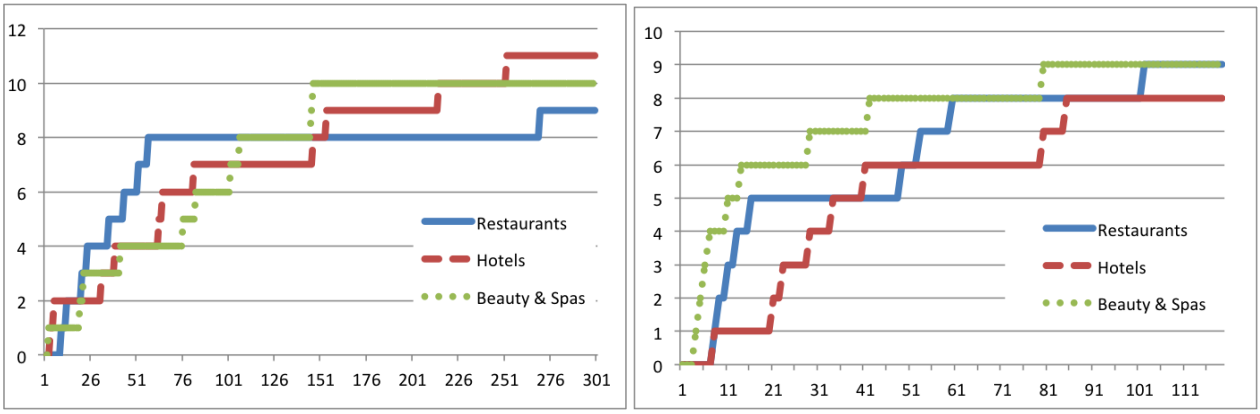

Figure 3: First appearance places of variables in the lists of (left) phrases and (right) topics.

5.2.5 Lists of Phrases and Topics. In addition, we would like to show that contextual phrases appear high in the resulting list of phrases and topics generated according to the methods described in Sections 3.2 and 3.3. Figure 3 (left) presents the performance of the phrase-based discovery method across the three applications (restaurants, hotels and beauty& spas). We plot the ordinal numbers of phrases in sorted lists obtained as described in Section 4.2.2 on the x-axes for the phrase-based method. We also plot on the y-axes the cumulative number of contextual variables $y(x)$ that are discovered by examining the first x phases of the sorted phrase list. Similarly, Figure 3 presents the performance of the LDA-based method for these applications. The three lines in Figures 3 (left and right) represent restaurants (blue), hotels (red) and beauty&spas (green) applications.

As we can see from Figure 3(left), the phrase-based method identified 8 contextual variables for each application within the first 150 phrases on the list, where 4 variables were identified from the first 50 phrases. This supports our observation that *many contextual variables appear high on the list of phrases* and therefore could be easily identified. As Figure 3(right) demonstrates, the *same* observation is true about the generated list of topics.

6 Conclusion

We presented a method for systematically discovering contextual information from user reviews that uses two methods for identifying the values of contextual variables. The first *phrase-based* approach identifies the contextual phrases appearing more frequently in the

specific than in the *generic* reviews, whereas the second *LDA-based* approach constructs the list of topics used for identifying the contextual variables. Our method is important because most of the CARS-related papers assume that the contextual information is given a priory, which is often not the case in practice, and the users have to employ various "tricks" to infer or estimate this contextual information. In contrast, our method provides for *systematic* discovery of most of the relevant contextual information from user reviews and thus enables deployment of the well-established representational paradigm of CARS with its benefits [7].

We validate our method on three applications and empirically show that the phrase- and the LDA-based methods work well on these applications by discovering most of the contextual variables mentioned in the user reviews: 9 of out 12, 11 out of 12, and 100% of all the contextual variables for the restaurant, hotels and beauty&spa applications respectively. We also show that our method missed only the variables that are infrequent or hard to describe in simple terms, such as *Traveling* that appeared in 1.25% of reviews. We compare the performance of the phrase- and the LDA-based methods across the three applications and show that the phrase-based approach performed at least as well as the LDA-based method for these applications. Finally, we show that all the variables that were discovered by our method are *important* for the rating-prediction problem since they affect rating distributions.

Although the whole context discovery process has been automated as described in the paper, nevertheless, our method assumes a certain amount of "manual" inspection. However, this is not a significant problem since it worked well on about one million reviews. Furthermore, our approach can be scaled to a much larger number of reviews because the amount of manual inspection is quite "manageable" for the applications described in this paper and grows only marginally in the total number of reviews, as argued in Section 4.2.3.

Finally, our method discovered not all the contextual variables in an application, but only those that were mentioned in the reviews. Furthermore, by manually examining only a subset of all the reviews, we might miss some of the contextual variables, albeit with low probability, as explained in Section 4.3.2. Note however, that these limitations are minor and, therefore, our method works well on the three applications described in the paper.

References

[1] Yelp dataset challenge, 2016. www.yelp.com/dataset_challenge.

[2] Aciar, S. Mining context information from consumers reviews. In *Proceedings of Workshop on Context-Aware Recommender System* (2010), ACM.

[3] Adomavicius, G., Mobasher, B., Ricci, F., and Tuzhilin, A. Context-aware recommender systems. *AI Magazine 32*, 3 (2011), 67–80.

[4] Adomavicius, G., Sankaranarayanan, R., Sen, S., and Tuzhilin, A. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Trans. Inf. Syst. 23*, 1 (Jan. 2005), 103–145.

[5] Adomavicius, G., and Tuzhilin, A. Multidimensional recommender systems: A data warehousing approach. In *Proceedings of the Second International Workshop on Electronic Commerce* (London, UK, UK, 2001), WELCOM '01, Springer-Verlag.

[6] Adomavicius, G., and Tuzhilin, A. Context-aware recommender systems. In *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer US, 2011, pp. 217–253.

[7] Adomavicius, G., and Tuzhilin, A. Context-aware recommender systems. In *Recommender Systems Handbook, 2nd edition*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer US, 2015.

[8] Ante Odic, Marko Tkalcic, J. F. T., and Kosir, A. Predicting and detecting the relevant contextual information in a movie-recommender system. In *Interacting with Computers* (2013), 25(1), Oxford University Press, pp. 74–90.

[9] Bazire, M., and Brézillon, P. Understanding context before using it. In *Proceedings of the 5th international conference on Modeling and Using Context* (Berlin, Heidelberg, 2005), CONTEXT'05, Springer-Verlag, pp. 29–40.

[10] Blei, D. M., Ng, A. Y., and Jordan, M. I. Latent dirichlet allocation. *J. Mach. Learn. Res. 3* (Mar. 2003), 993–1022.

[11] Brill, E. A simple rule-based part of speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing* (Stroudsburg, PA, USA, 1992), ANLC '92, Association for Computational Linguistics, pp. 152–155.

[12] Colson, E. Blending human computing and recommender systems for personalized style recommendations. In *RecSys* (2014).

[13] Dourish, P. What we talk about when we talk about context. *Personal Ubiquitous Comput. 8*, 1 (Feb. 2004), 19–30.

[14] Gomez-Uribe, C. A., and Hunt, N. The netflix recommender system: Algorithms, business value, and innovation. *ACM Trans. Manage. Inf. Syst. 6*, 4 (Dec. 2015).

[15] Hariri, N., Mobasher, B., and Burke, R. Query-driven context aware recommendation. In *Proceedings of the 7th ACM Conference on Recommender Systems* (New York, NY, USA, 2013), RecSys '13, ACM, pp. 9–16.

[16] Hariri, N., Mobasher, B., Burke, R., and Zheng, Y. Context-aware recommendation based on review mining. In *Workshop on Intelligent Techniques for Web Personalization and Recommender Systems* (2011), pp. 30–36.

[17] Hastings, R. 2012 re:invent day 1 keynote, 2012. youtu.be/8FJ5DBLSFe4?t=44m40s.

[18] Jin, X., Zhou, Y., and Mobasher, B. Task-oriented web user modeling for recommendation. In *Proceedings of the 10th international conference on User Modeling* (Berlin, Heidelberg, 2005), UM'05, Springer-Verlag, pp. 109–118.

[19] Justeson, J. S., and Katz, S. M. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering 1* (3 1995).

[20] Kaminskas, M., and Ricci, F. Location-adapted music recommendation using tags. In *User Modeling, Adaption and Personalization*, vol. 6787 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2011, pp. 183–194.

[21] Li, A. Q., Ahmed, A., Ravi, S., and Smola, A. J. Reducing the sampling complexity of topic models. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2014), KDD '14, ACM, pp. 891–900.

[22] Li, Y., Nie, J., Zhang, Y., Wang, B., Yan, B., and Weng, F. Contextual recommendation based on text mining. In *International Conference on Computational Linguistics: Posters* (2010), COLING '10, pp. 692–700.

[23] Manning, C. D., Raghavan, P., and Schütze, H. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[24] Manning, C. D., and Schütze, H. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA, 1999.

[25] Olson, D. L., and Delen, D. *Advanced Data Mining Techniques*, 1st ed. Springer Publishing Company, Incorporated, 2008.

[26] Palmisano, C., Tuzhilin, A., and Gorgoglione, M. Using context to improve predictive modeling of customers in personalization applications. *IEEE Trans. on Knowl. and Data Eng. 20*, 11 (Nov. 2008), 1535–1549.

[27] Panniello, U., Tuzhilin, A., and Gorgoglione, M. Comparing context-aware recommender systems in terms of accuracy and diversity. *User Modeling and User-Adapted Interaction 24*, 1-2 (Feb. 2014), 35–65.

[28] Pedregosa, F., and et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830.

[29] Rehurek, R., and Sojka, P. Software framework for topic modelling with large corpora. In *Proceedings of LREC 2010 workshop New Challenges for NLP Frameworks* (Valletta, Malta, 2010), University of Malta, pp. 46–50.

[30] Smola, A., and Narayanamurthy, S. An architecture for parallel topic models. *Proc. VLDB Endow. 3*, 1-2 (Sept. 2010), 703–710.

[31] Sontag, D., and Roy, D. Complexity of inference in latent dirichlet allocation. In *Advances in Neural Information Processing Systems 24* (2011), pp. 1008–1016.

[32] Tewari, G., Youll, J., and Maes, P. Personalized location-based brokering using an agent-based intermediary architecture. *Decision Support Systems 34*, 2 (2003), 127 – 137. Agents and E-Commerce Business Models.

[33] Tung, H.-W., and Soo, V.-W. A personalized restaurant recommender agent for mobile e-service. In *IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'04)* (2004), pp. 259–262.

[34] Yuan, J., Gao, F., Ho, Q., Dai, W., Wei, J., Zheng, X., Xing, E. P., Liu, T.-Y., and Ma, W.-Y. Lightlda: Big topic models on modest computer clusters. In *International Conference on World Wide Web* (2015), pp. 1351–1361.

[35] Zheng, Y., Mobasher, B., and Burke, R. D. The role of emotions in context-aware recommendation. In *RecSys* (2013), pp. 21–28.