Noname manuscript No.
(will be inserted by the editor)

Opinion-Based Entity Ranking (Author's Draft)

Kavita Ganesan · ChengXiang Zhai

the date of receipt and acceptance should be inserted later

Abstract The deployment of Web 2.0 technologies has led to rapid growth of various opinions and reviews on the web, such as reviews on products and opinions about people. Such content can be very useful to help people find interesting entities like products, businesses and people based on their individual preferences or tradeoffs. Most existing work on leveraging opinionated content has focused on integrating and summarizing opinions on entities to help users better digest all the opinions. In this paper, we propose a different way of leveraging opinionated content, by directly ranking entities based on a user's preferences. Our idea is to represent each entity with the text of all the reviews of that entity. Given a user's keyword query that expresses the desired features of an entity, we can then rank all the candidate entities based on how well opinions on these entities match the user's preferences. We study several methods for solving this problem, including both standard text retrieval models and some extensions of these models. Experiment results on ranking entities based on opinions in two different domains (hotels and cars) show that the proposed extensions are effective and lead to improvement of ranking accuracy over the standard text retrieval models for this task.

Keywords Vertical Search · Preference Based Entity Search · Opinion Matching · Product Search · Entity Search · Ad-hoc Faceted Navigation

Kavita Ganesan
Department of Computer Science,
University of Illinois at Urbana-Champaign,
Urbana, IL 61801
E-mail: kganes2@illinois.edu

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

1 Introduction

The era of Social Computing has kindled massive growth of opinions and reviews on the web, including reviews on businesses, products and opinions about people. Let us just consider reviews of movies. On yahoo's directory listing[1], the number of movie review sites alone is nearing two hundred. This number does not even include the growing number of blogs or social networking sites where people have the ability to freely express opinions about movies.

The vast amount of opinions expressed by experts and ordinary users can be very useful to help people make all kinds of decisions, ranging from what to buy to what treatment to choose for a disease. For example, shoppers at Amazon[2] typically would read the reviews about a product before buying it, and travelers may rely on opinions about hotels on Tripadvisor[3] to help them choose an appropriate hotel at the destination. It has been shown that 77% of online shoppers use reviews and ratings when making a purchase decision[4].

Unfortunately, the abundance of opinions also poses challenges in digesting all the opinions about an entity or a topic. For example, a popular product such as the *iPhone* may have hundreds of reviews on Amazon.com, and popular hotels like *Marriott* or *Hilton* may have over five hundred reviews on Tripadvisor. Thus, the task of developing computational techniques to help users digest and exploit all the opinions is a very important and interesting research challenge.

Most existing work on tackling this general challenge has focused on integrating and summarizing opinions to help users better digest all the opinions (see Section 2 for a detailed review of related work). In this paper, we propose a different way of leveraging opinionated content, that is to *directly* rank interesting entities based on how well the opinions on these entities match a user's preferences. Since a user is often interested in choosing an entity based on the opinions on the entity, ranking entities in this way provides a more *direct support* for a user's decision-making task. For example, the decision-making task in the case of a *user shopping for a product* is to decide which product out of the many to buy. Thus, it would be very helpful for such a user if we can take a keyword query from the user expressing his/her preferences for the product (e.g.,"comfortable seats, cheap and reliable" for a car), and return a ranked list of cars in the order of likelihood that a car matches the users preferences. With such a capability, the user is no longer overwhelmed by all the reviews available on all cars, but rather the user can now analyze a much smaller set of cars that roughly matches his/her preferences based on the judgment of other users. Further, this type of ranking is flexible in that it can be applied to any entity for which opinionated content is available.

To rank entities in this way, our idea is to represent each entity with the text of all the reviews of that particular entity, often available from various websites. Given a user's keyword query that expresses the desired features of an entity, we can then rank the relevant entities based on how well its reviews match the user's preferences. An ideal setup for an Opinion-Based Entity Ranking system is as shown in Figure 1, where the user can freely express preferences as a natural keyword query.

[1] http://dir.yahoo.com/
[2] http://www.amazon.com
[3] http://www.tripadvisor.com
[4] http://www.mediapost.com/publications/

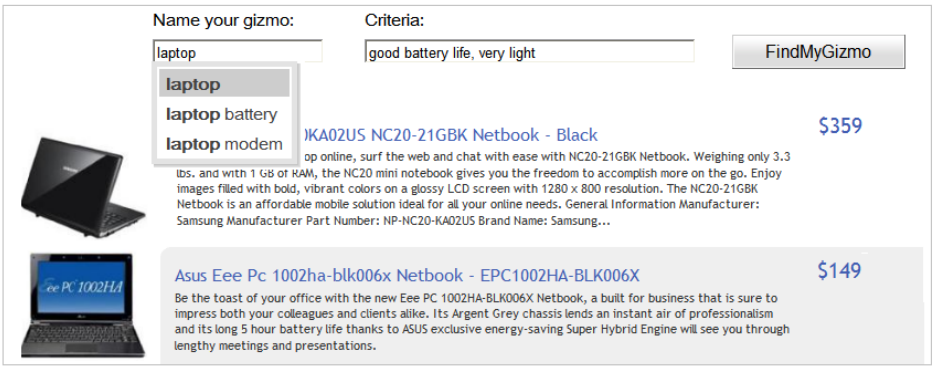

Fig. 1 An ideal Opinion-Based Entity Ranking System that accepts keyword preferences as a natural keyword query.

Fig. 2 One scenario of Opinion-Based Entity Ranking applications where keyword preferences are expressed on a set of aspects.

It is natural for a user to specify preferences on various aspects of an entity in the envisioned entity ranking task. Thus we can expect a user's query to consist of preferences on multiple aspects; for example, a preference query for a car might be "good gas mileage, cheap, reliable", which consists of preferences on three different aspects (i.e., efficiency, price, and reliability). In general, if a user enters a query in a single query box, we would need to parse a query to obtain preferences on different aspects. In this paper, we focus on studying effectiveness of different ranking methods, so we assume that the multiple aspects in a user's query have already been segmented in order to factor out the influence of query segmentation on retrieval accuracy. Such a query can also be naturally obtained by providing a multi-aspect query form or asking a user to use a delimiter (e.g., a comma) to separate multiple preferences. For example, in Figure 2, we show a system interface where the users can find hotels in any city by stating their preferences on the various aspects of hotels.

Although this ranking problem closely resembles an information retrieval problem where the reviews of an entity can be regarded as an "entity document," there are two important differences. First, the query is meant to express a user's preferences in keywords; thus it is expected to be longer than regular keyword queries on the Web. More importantly, the query generally would contain preferences on multiple aspects of an entity. As we will show later in the paper, modeling these aspects can improve ranking accuracy. Second, the ranking criteria are to capture how well an entity satisfies a user's preferences rather than the relevance of a document to a query as in the case of regular retrieval. Therefore, the matching of opinionated words or sentiment would be important. We will show that although traditional query expansion works reasonably

well in some cases, expanding a query with similar opinion words can significantly improve ranking accuracy on different types of data.

In addition to evaluating the effectiveness of standard text retrieval models for this task, we further propose several extensions of these models to better solve this special ranking problem. Specifically, we propose two heuristics: (1) *query aspect modeling* where we use each query aspect to rank entities and then aggregate the ranked results from the multiple aspects of the query; and (2) *opinion expansion* where we expand a query with related opinion words found in an online thesaurus. Our approach is light-weight, scalable and flexible as we avoid the need for costly information extraction and data mining.

Evaluation of this ranking task is a challenge since no existing test collection can be used for evaluation. We thus opted to create a benchmark data set by leveraging existing rating information. While it is not hard to collect reviews for different entities, it is a significant challenge to obtain reasonable queries and also to evaluate ranking accuracy quantitatively. We propose to solve this problem by leveraging the ratings of different aspects of cars and hotels available on Edmunds.com[5] and Tripadvisor.com[6], and created two different data sets as a gold standard for quantitative evaluation. The data sets are available at `http://sifaka.cs.uiuc.edu/ir/downloads.html`.

Experimental results on these two data sets show that the proposed extensions over standard retrieval models are effective for the task of opinion-based entity ranking. The focused expansion technique (i.e. opinion expansion) is shown to be particularly effective. Modeling the aspects in a user's query as opposed to just treating the query as a "long keyword query" is also beneficial, especially for longer queries with more aspects.

2 Related Work

To the best of our knowledge, no previous study has leveraged opinionated content to rank entities the way we have proposed. However, there are several lines of related work which we briefly describe in this section.

Sentiment Analysis. Sentiment analysis involves classifying opinions in text into categories like "positive" or "negative" often with an implicit category of "neutral". Methods in this line of work can be categorized as supervised (requires labeled training data) [18, 7, 16, 4], unsupervised (relies on lexicon and external knowledge) [29, 14] or hybrid approaches [17, 20]. While sentiment analysis provides a means to generate polarity ratings at different levels of granularity (document, sentence or phrase), it does not provide direct support in matching a user's preference on an aspect with polarity ratings on the aspect of interest. Moreover, since these ratings are categorical, it would be ineffective to rank entities based on whether its aspect is "positive" or "negative".

Rating Prediction and Decomposition. In recent years, there has been work in trying to decompose reviews to make aspect based rating predictions [30, 13, 26]. This line of work is closely related to ours as, once we obtain ratings on different aspects, we would be able to rank entities based on their ratings in the aspects interesting to a user.

[5] http://www.edmunds.com/
[6] http://www.tripadvisor.com/

This approach, however, has some practical limitations. First, these approaches assume a fixed number of aspects on a given entity. It is not only impractical to define or mine a set of aspects for each category of entities (e.g. politicians: *approval rating, character*; laptops:*battery life, screen*), but a fixed number of aspects would also severely limit the type of queries a user could issue. More importantly, all the work in this line, require some supervision in that they require the availability of ratings associated with reviews, which may not always be present. We take a more general stance, that is to assume limited knowledge on the opinions and the aspects being queried and focus on leveraging robust retrieval models to match the user's preferences for an entity with opinions on that entity.

Expert Finding. Another relevant area of research is *Expert Finding*. Expert finding is about finding people rather than documents and the goal is to retrieve a ranked list of experts with expertise on a given topic [6, 2, 11]. The techniques used range from standard retrieval methods [11] like the vector space model to state-of-the-art techniques [2, 6] that use probabilistic and language modeling approaches. Although our work is conceptually related, in that we use information about an entity to rank entities, unlike expert finding we can rank any type of entity for which opinionated content is available. Also, instead of trying to rank entities based on how well it matches a topic, we focus on ranking entities based on how well a user's preferences are matched with opinions on that entity.

Opinion Retrieval. Opinion retrieval was first explored in the TREC Enterprise Track (on email search). The goal of opinion retrieval is to locate documents (primarily blog posts) that have opinionated content. The idea here is to test the ability to find opinion expressing posts as this is essential in specialized searches like blog search. An opinion retrieval system [8, 32] is usually built on top of standard retrieval models where relevant content is first retrieved, and then opinion analysis is done on the retrieved content to return only opinionated documents. In contrast, our idea assumes that we already have the opinionated content for a given category of entities (e.g. *reviews for all hotels in San Francisco)*. The goal is thus to rank the entities in the order of likelihood that the entity matches the user's preferences.

Multifaceted Search. Multifaceted search is highly related to our general goal. Faceted search, also called *faceted navigation* or *faceted browsing*, allows users to explore and find information that they need by filtering or navigating with the help of some pre-determined facets [28]. The users often provide a very general query (some systems do not support queries), and then they use the various facets to navigate through the results until the items of interest are found. In other words, the goal is to connect users to items that are of most interest to them. While our goal is similar, the paradigm is different. First of all, in our setup, users find entities based on unstructured text containing opinions of other users rather than structured or categorical data (often used in faceted navigation). In addition, our focus is more on the keywords in the query that allows users to specify their interest on various facets. For example, a user who is looking for a laptop with a specific criteria, would provide a query such as 'Lenovo, very light, bright screen'. In such a query, the facets are actually implicit where in this case the facets being queried are *brand*, *weight* and *screen*. In traditional faceted navigation, these facets are explicitly defined and are usually fixed. Thus, our idea can be considered an *ad-hoc faceted navigation* or a *personalized faceted navigation*[10]

system. Our idea can be combined with 'traditional' faceted navigation to provide a powerful search system that can greatly improve user productivity.

3 Methods for Opinion-Based Entity Ranking

In this section, we present several methods for ranking entities based on how well its opinions match a user's preferences, including both standard retrieval models, which we treat as baselines, and some extensions of these models that we propose. To facilitate the discussion, we first introduce some notation. Let $E = \{e_1, ..., e_n\}$ be a set of entities to be ranked. For each entity e_i, we assume that we can collect a set of review documents $R_i = \{r_{i1}, ..., r_{in_i}\}$ that contain the opinions about the entity expressed by users or reviewers, where r_{ij} is a review document. Let D_i be the concatenation of all the review documents of an entity e_i. For convenience, we call D_i the opinion document for entity e_i. To solve the entity ranking problem, we cast it as a text retrieval problem where the text collection $\mathcal{C}$ consists of all the opinion documents for all the entities. That is, $\mathcal{C} = \{D_1, ..., D_n\}$.

From a user's perspective, the easiest way to express preferences for an entity would be to use keywords to describe desirable properties in various aspects. For example, a query for cars may look like "good gas mileage, small size, reliable." We denote such a keyword query by Q. On the surface, our problem is very similar to a regular retrieval problem. However, as discussed in Section 1, there are some important differences, which we will leverage to extend a regular retrieval model to improve ranking accuracy. In particular, our queries semantically consist of a set of sub-queries each describing preferences for one separate aspect of an entity, and we will show that it is indeed beneficial to model these semantic aspects. We will also show that emphasizing matching of opinion words through opinion expansion is very effective because it captures the desired matching criteria of relevance better for this ranking task. We now present three baseline standard retrieval models and then we present the two extensions mentioned.

3.1 Standard retrieval models

By casting the entity ranking problem as a problem of preference matching, we can directly use any standard retrieval model to solve the problem. Here we present three state-of-the-art standard retrieval models that we will experiment with; they are known to be most effective [1, 5] for the task of text retrieval.

3.1.1 BM25 (Okapi)

The BM25 (or Okapi) retrieval function was proposed by Robertson et. al [22] and has been shown to be quite effective and robust for many tasks. Although it was derived based on probabilistic models, it can also be regarded as a variant of the popular vector space model since it provides a term frequency-inverse document frequency (TF-IDF) weighting-based ranking formula. Formally, the score of an opinion document D in collection $\mathcal{C}$ (with n documents) and a query Q is given by:

$$S_{BM25}(D,Q) = \sum_{t \in Q \cap D} \frac{k_1 c(t,D)}{c(t,D) + k_1(1 - b + b * |D|/|\widetilde{\mathbf{D}}|)} \times \log \frac{n+1}{n_t}$$

where $c(t,D)$ and $c(t,Q)$ are the count of term t in document D and query Q, respectively, $|D|$ is the length of document D, $|\widetilde{\mathbf{D}}|$ is the average document length in the collection, n_t is the number of documents containing term t, and b, k_1, and k_3 are parameters that are typically set as $b = 0.75$, k_1 between 1.0 to 2.0, and k_3 between 0 and 1000. We replaced the IDF in the original Okapi formula with the normal IDF because the original one causes negative weights [5] and also performs significantly worse than the normal one in our experiments.

3.1.2 Dirichlet prior

The Dirichlet prior retrieval function is one of the most effective language models for retrieval [34]. It is derived based on query likelihood scoring [19] and Bayesian estimation of document language model [12], but its weighting formula also resembles TF-IDF weighting and document length normalization. Formally, the score of document D and query Q is:

$$S_{Dir}(D,Q) = \sum_{t \in Q \cap D} c(t,Q) \log(1 + \frac{c(t,D)}{\mu p(t|\mathcal{C})}) + |Q| \log \frac{\mu}{\mu + |D|}$$

where the notations are as in Okapi, $p(t|\mathcal{C})$ is the probability of term t according to a background collection language model, and μ is a smoothing parameter to be empirically set.

3.1.3 PL2

PL2 is one of the most effective functions in the family of divergence from randomness retrieval (DFR) models [1]. Its scoring formula is based on basic statistics similar to those used in other retrieval functions and is formally defined as:
$S_{PL2}(D,Q) = \sum_{t \in Q \cap D} c(t,Q)$

$$\times \frac{tfn_t^D \cdot \log_2(tfn_t^D \cdot \lambda_t) + \log_2 e \cdot (\frac{1}{\lambda_t} - tfn_t^D) + 0.5 \cdot \log_2(2\pi \cdot tfn_t^D)}{tfn_t^D + 1}$$

where $tfn_t^D = c(t,D) + \log_2(1 + c \cdot \frac{|\widetilde{\mathbf{D}}|}{|D|})$, $\lambda_t = \frac{n}{c(t,\mathcal{C})}$ ($c(t,\mathcal{C})$ is the count of term t in the collection $\mathcal{C}$) and $c > 0$ is a retrieval parameter.

All these three standard retrieval models have corresponding pseudo feedback methods that can take some top ranked documents in an initial retrieval result as if they were relevant documents to extract additional terms to expand a query. Since we use the Terrier[15] toolkit for our experiments, we leverage the pseudo feedback mechanism implemented in this toolkit. Terrier provides various DFR[1] based term weighting models for query expansion. We specifically use the Bose-Einstein 1 (Bo1) model, which is based on Bose-Einstein statistics [3] and is similar to Rocchio[24].

Although standard retrieval models can be used to solve the opinion-based entity ranking problem, these models do not consider the multiple aspects in the query. It also

does not consider the special notion of "relevance" when matching an opinion document with a query. Below, we present two extensions of a standard retrieval function to model query aspects and expand a query with opinion words.

3.2 Query Aspect Modeling (QAM)

In our setup, we assume that separate query fields would be provided for each aspect, thus the query would naturally consist of multiple aspects. However, a standard retrieval model would not distinguish these multiple aspects; as a result, it is possible that an entity may be scored high just because of matching one of the many aspects extremely well. Thus, one way to improve a standard retrieval function is to use each aspect query to score an opinion document (equivalently an entity) and then combine the scores of an entity in all the query aspects. This way, we can ensure that an entity matches all the aspects. Another potential advantage of modeling aspects in a query, though not explored in this paper, is the ability to add expansion terms that are relevant to the aspect. For example, say we have a two aspect query - '*good gas mileage*' and '*extremely comfy*'. If we distinguish this query based on aspects, for *'good gas mileage'*, terms like *'mpg','mileage', 'fuel'* can be potentially added. However, if we treat the user's preferences as long query, without distinguishing aspects, we have to be very careful on the type of terms added as we may end up retrieving items that are better in one aspect compared to the other.

While we have assumed separate query fields for different aspects, the aspects in a query can also be obtained explicitly by asking a user to use a special delimiter such as a *comma* to separate multiple aspects. These aspect queries can also be obtained from a regular keyword query using query *parsing* or *segmentation* techniques as shown in the work of [27]. Thus, by capturing multiple aspects in the query, we may now denote a query with $Q = \{Q_1, ..., Q_k\}$ where $k \geq 1$ and Q_i is a keyword query for an aspect of the entity, which we will refer to as an *aspect query*.

We now present several methods for leveraging this aspect structure. Let $S(D, Q)$ be any retrieval function. We can use the function to compute a score for each document with respect to each aspect query Q_i (i.e., $S(D, Q_i)$), and then combine the scores to generate an overall score for each document. Depending on how we combine the scores, we have several variants of this query aspect modeling (QAM) strategy. In particular, we can either combine the scores directly or combine the ranks of documents according to their scores in each query aspect. Moreover, we can also use different ways to aggregate the scores or ranks. In our experiments, we tested the following QAM scoring methods:

Average Score: $S_{AvgScore}(D, Q) = \frac{1}{k} \sum_{i=1}^{k} S(D, Q_i)$

Average Rank: $S_{AvgRank}(D, Q) = \frac{1}{k} \sum_{i=1}^{k} Rank(D, Q_i)$

Median Rank: $S_{MedRank}(D, Q) = Median_{i \in [1,k]} Rank(D, Q_i)$

Min Rank: $S_{MinRank}(D, Q) = Min_{i \in [1,k]} Rank(D, Q_i)$

Max Rank: $S_{MaxRank}(D, Q) = Max_{i \in [1,k]} Rank(D, Q_i)$

Here, $Rank(D, Q_i)$ refers to the rank of document D in the ranked list of documents for aspect query Q_i. Note that we did not consider other variations of score combination because of the concern that scores of a document in different aspects may not be comparable.

3.3 Opinion Expansion

Another limitation of the standard retrieval models for opinion-based entity ranking is that matching an opinion word and matching an ordinary topic word are not distinguished. Intuitively, since we would like to reward an opinion document where a query aspect is favorably reviewed, it is important to match opinion words in the user's query. However, since topic words are expected to be much more common in review documents and have less variation than opinion words, we hypothesized that expanding a query with additional "equivalent" opinion words may help in emphasizing the matching of opinion words.

Consider a query like *'fantastic battery life'*. Due to the non-uniform way in which people express opinions, for the same expression, some may say *'awesome battery life'* while others may say something brief such as *'good battery'*. Therefore, it would be beneficial to expand such a query by adding synonyms of the word *fantastic*.

We thus propose the following opinion expansion method to expand a query with related opinion words. We use a controlled online dictionary[7] to first extract two classes of words from the query: (1) **intensifiers**, which are adverbs such as *very, really, extremely* and (2) **common praise words**, which are adjectives such as *good, great, fantastic*. In the case of intensifier words, we use only words that are neutral, where the orientation of the word would depend on the word or phrase following. This is to avoid changing the intended orientation of the query. For example, for the query 'extremely comfortable car', related intensifiers such as *exaggeratedly* and *excessively* can change the actual meaning of the user's preference as both these words have negative connotation. Such words would thus not be included in our list or expanded on when opinion expansion is performed. Table 1 shows the complete list of intensifiers and praise words used for opinion expansion.

For a given query Q, we can add synonyms of query terms to the query to enrich the representation of opinions and accommodate flexible matching of opinions. Formally, let t_i be a term in a given query Q. Let $syn_{a_1}, ..., syn_{a_{35}}$ be the set of synonyms for praise words and $syn_{b_1}, ..., syn_{b_{23}}$ be the set of synonyms for intensifier words. If t_i matches an intensifier term or a praise term, the corresponding synonyms will be appended to the query. Even if there are multiple praise words or intensifiers in a query, the expansion is done only once.

4 Data Set

Since the task of opinion based entity ranking as we defined has not been studied previously, no test collection exists for this task. This makes it a challenge to quantitatively evaluate the proposed methods. In this section, we describe how we address this challenge by creating a benchmark data set from two different domains. While review

[7] thesaurus.com

praise words	intensifiers
acceptable	absolutely
admirable	acutely
agreeable	amply
amazing	astonishingly
awesome	certainly
commendable	considerably
decent	dearly
excellent	decidedly
exceptional	deeply
fantastic	eminently
favorable	emphatically
genius	extensively
good	extraordinarily
gratifying	extremely
great	highly
honorable	incredibly
lovely	really
marvelous	substantially
nice	tremendously
pleased	truly
pleasing	very
premium	
remarkable	
satisfactory	
satisfying	
sound	
splendid	
stupendous	
super	
superb	
superior	
terrific	
tremendous	
wonderful	
worthy	

Table 1 List of praise words and intensifiers used for opinion expansion

documents are easy to obtain from the Web, it is unclear how we can obtain queries and create a gold standard to quantitatively evaluate the proposed methods for entity ranking. We propose to use seed aspect queries to generate synthetic longer queries and leverage the available numerical aspect ratings as if they were relevance judgments. We believe that the creation of this first test data set and the associated evaluation methodology for ranking entities, is one of the important contributions of this work. The data set is available at `http://sifaka.cs.uiuc.edu/ir/downloads.html`.

4.1 Review Document Collection

Our task is to return a set of entities based on how well the user's keyword preferences match the opinions on these entities. Therefore, we need a reasonable sized opinion data set supporting each entity. Although our idea is to allow the retrieval of any entity with supporting opinions, we chose to limit to sources that have free-text opinions accompanied by numerical ratings on individual aspects. We restricted our search to such sources to facilitate the evaluation of our task (explained in detail in Section 4.3).

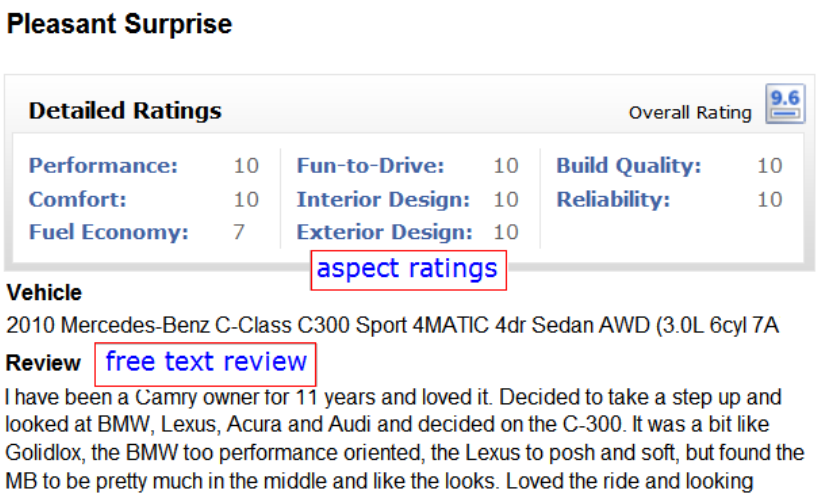

Fig. 3 A sample car review from Edmunds.com.

Car Data Set						Hotels Data Set					
		average aspect ratings						average aspect ratings			
year	# of cars	max	min	mean	var	city	# of hotels	max	min	mean	var
07'	**227**	10.00	5.13	8.72	0.54	beijing	98	5.00	2.56	4.10	0.25
08'	228	10.00	3.79	8.75	0.63	chicago	116	4.92	1.70	4.02	0.31
09'	**143**	10.00	6.03	8.85	0.41	dubai	148	5.00	1.60	3.92	0.49
						las-vegas	154	5.00	1.12	3.70	0.47
						london	**727**	4.96	1.00	3.53	0.71
						montreal	98	4.97	1.10	3.79	0.57
						new-delhi	**80**	5.00	1.58	3.55	0.51
						new york city	246	4.98	2.58	4.09	0.19
						san-francisco	186	4.94	1.32	3.78	0.52
						shanghai	92	4.93	2.09	3.95	0.27

Table 2 Basic statistics on collected review data used in experiments. Columns labeled min, max and mean are based on the averaged per aspect user ratings for each entity.

With careful analysis, we chose to use reviews from two different domains that represent **different types of reviews**. The first is car reviews from Edmunds.com and the second is hotel reviews from Tripadvisor.com. Both sources have free-text reviews accompanied by numerical ratings on several aspects (all provided by users).

The nature of car reviews on Edmunds.com differs from hotel reviews on Tripadvisor.com. The hotel reviews are far more verbose than the car reviews. Most reviews on cars are only 4-5 sentences long, while the hotel reviews can span several paragraphs with detailed explanation of the reviewer's experience. Figure 3 shows an example of a car review from Edmunds.com. The section titled *Detailed Ratings* provides us with the discrete aspect ratings for each review.

To construct our data set, we collected reviews on cars for model-year 2007, 2008, and 2009 and hotel reviews for hotels in 10 major cities internationally. This includes hotels in London, Beijing, Shanghai, Montreal, New Delhi, Dubai, New York City, Chicago, San Francisco and Las Vegas. In creating our data set, we avoided reviews that were too sparse as there would not be sufficient opinion text to test the effectiveness of a ranking method. Thus, we ensured that we only considered cars/hotels that had at least least 10 reviews.

The accompanying aspect ratings on Edmunds.com are on 8 different aspects, namely *fuel economy, comfort, performance, reliability, interior design, exterior design, build* and *fun to drive*. These ratings are on a scale of 1-10. As for hotel reviews, there are 5 aspects and this includes *cleanliness, value, service, location* and *room*. These ratings are on a scale of 1-5.

Table 2 provides a summary of the collected data. Columns labeled *min* and *max* show the absolute minimum and maximum aspect ratings for a given model-year/city, where the aspect ratings have been averaged across reviews of the same entity. The mean aspect ratings and variance are also shown in this table. Overall, the variance in ratings in both data sets is small.

4.2 Query Generation

The queries expected in an opinion-based entity ranking system are very different from a regular query one would issue to a typical vertical search engine, like a product search engine. If a user were looking for a laptop on Google Product Search[8], the user would typically type short keywords like *laptop* or *dell laptops*. Such systems generally return a list of entities without any specific order to start with, allowing the user to narrow down into the items of interest using different filters or through faceted navigation.

In our case, assuming that the type of entity (e.g. people, cars, hotels, restaurants) being searched for is known, users can then state their preferences for that entity using a set of descriptive keywords. These keywords would indicate what the user desires in the different aspects of that entity. For example, for a laptop we can have a query such as *'dell, good battery life, bright screen, very portable'*. The system would then return a ranked list of entities in the order of likelihood that the entity matches the user's preferences. Queries issued to a system such as this would thus have two important properties: (1) the query lengths can vary greatly - from short queries like *'good battery life'* to longer queries like *'excellent battery life, bright screen, lightweight'* and (2) the queries may contain opinion indicating words and intensifiers (e.g. very, extremely, good, super, excellent).

While there are many vertical search systems like Google Product Search, there exists no system that currently takes a set of keyword based preferences as shown in Figure 1. This makes it hard for us to obtain a natural sample of queries. We thus constructed our test queries from a set of seed queries. Since we expect the user to express his/her preferences on a fixed number of aspects, for the purpose of evaluation, we assume that these aspects would correspond to the aspects that have associated numerical ratings in our data set. We manually obtained a set of seed queries for each of these aspects and then we randomly combined the seed queries from different aspects to form longer multi-aspect queries that we call *generated queries*.

Specifically, we asked three average users to provide a few queries that they would issue on the various aspects of entities in our data set, to 'find' those that match their preferences. So, a user who desires a comfortable car with good gas mileage may issue a query such as '*comfortable seats, excellent mpg*', where 'comfortable seats' corresponds to the *comfort* aspect and 'excellent mpg' corresponds to the *fuel economy* aspect. The user thus specifies both the aspect being queried and the query keywords for that aspect. This is to simulate the behavior of obtaining queries from a query interface

[8] http://www.google.com/products

Aspect	Words used for keyword matching	Mentions
comfort	comfort	15530
interior	interior	13068
fuel economy	gas, fuel	10924
performance	performance	5013
build	built, build	4156
reliability	reliability, reliable	4119
exterior	exterior	3122

Table 3 Approximate aspect mentions in the car dataset.

such as the one in Figure 2. With this, we obtained an average of six seed queries per aspect (5 for hotels and 7 for cars) for the two domains. We ignored one aspect, *'exterior design'* as it was not a popular topic of discussion within the car reviews, and hence may not help in evaluating retrieval methods that rely on keyword matching. In Table 3, we show the estimated aspect mentions in the car dataset. These numbers were obtained by counting the number of times the representative words in each aspect were mentioned.

Through random combination of seed queries from different aspects, we generated 10,000 queries per data set. These queries are to be used with entities in each city (for hotels) and model-year (for cars). The shortest query is one aspect long and the longest query can be a query that touches each aspect of the car/hotel. Each *generated query* can have at most one seed query from a given aspect. Table 4 shows some sample seed queries defined on 2 different aspects of cars and hotels and Table 5 shows some sample *generated queries* for the car data set.

Since the seed queries were obtained without a real system in place, it is important to ensure that these seed queries indeed represent typical user queries in our evaluation domain. Queries submitted to a car or a hotel search engine would not be useful because such systems are typically very structured and have limited support for natural keyword queries. However, users tend to use the major search engines like Bing[9], Yahoo![10] and Google[11] as a starting point to many of their search activities. Since the *query suggestion* feature of search engines is based on what other users have searched on, and the *related searches* feature is typically mined from query logs [23], we use both these features to determine how representative our seed queries are in these two domains.

We append the entity type to each seed query (for e.g., 'very clean' + '*hotel*' for the cleanliness aspect of hotels) and use that as a query into the major search engines. We then note the related searches and query suggestions for each seed query. We call these the *common aspect queries*. For example, a query like 'clean hotels' may yield in common aspect queries like 'clean hotels in Las Vegas' and 'clean hotels NYC cheap'. With this, we know that the seed query indeed reflects a natural user query. Almost all seed queries (in both domains) returned a set of common aspect queries on the major search engines. Table 6 shows some of the seed queries with corresponding common aspect queries for each aspect in the two domains. The *build* aspect from the cars domain and the *service* aspect from the hotels domain are the only ones that had limited or no related queries (in all three search engines). This makes sense as some aspects are relatively more subjective or opinion oriented. So, it is not very

[9] www.bing.com
[10] www.yahoo.com
[11] ww.google.com

Cars	
Aspects	**Sample Seed**
fuel economy	good gas mileage, great mpg
comfort	comfortable, very comfy

Hotels	
Aspects	**Sample Seed**
cleanliness	very clean, clean
value	cheap, affordable

Table 4 Sample *seed queries* used to generate longer *multi-aspect queries*

Aspects	Generated Queries
comfort	*comfortable* *very comfy*
comfort, fuel	*comfortable, good gas mileage* *very comfy, great mpg*
comfort, reliability, fuel	*comfortable, reliable, good gas mileage* *comfy, dependable, great mpg*

Table 5 Example of *generated queries* for the car data set

likely that users would search for 'hotels with polite staff' on the major search engine sites. However, given a system like the one we envision, it would be more likely that such queries would be encountered. Therefore, these seed queries provide a nice mix of what a user typically looks for in these domains and what users could potentially search for in the future given an opinion-based search system. For further analysis, we looked into the Microsoft Live Labs query logs (released in 2006) to see what the most frequently mentioned aspects of preferences are in these two domains. This query log has 15 million queries, from US users, sampled over one month. Although this is a relatively small query log, it is sufficient enough to show some word distribution in these domains. For this, we used the words 'cars' and 'hotels' to retrieve all related queries from the query logs. For each domain, we then collect the counts of terms in these retrieved queries and sort them in decreasing order of their frequencies. The top 50 query words related to the purchasing of a car and the top 30 query words related to finding a place to stay are shown in Table 7. We see that all these words can be mapped into the aspects that we considered in generating our queries (the mappings are shown in parentheses in the table). Furthermore, in both domains, most of the aspects that we used for evaluation (i.e., aspects with known ratings from reviewers in Tripadvisor.com and Edmunds.com) were indeed queried by users. The aspects not well covered in these top query words are the *fun* and *comfort* aspects for cars and the *cleanliness* aspect for hotels. We believe that this does not necessarily indicate a lack of interest by users in these aspects, but rather, it is likely that users would not expect the current search engines to return meaningful results for such aspects, thus they would not even try such queries. Overall, the query log analysis results indicate that the queries we generated indeed represent typical aspects of preferences that users are interested in when ranking cars and hotels.

4.3 Relevance Judgments Generation

One of the most important task in our evaluation is to determine how well the retrieved entities match the user's preferences. Ideally, for a subjective task like this, given a user's preference query, we would need a human judge to read the related re-

Cars		
Aspect	**Sample Seed**	**Related User Queries from Google, Yahoo, Bing**
fuel	good gas mileage good fuel economy decent gas mileage excellent fuel economy	cars with high mpg cars with great gas mileage fuel efficient cars good fuel economy trucks cheap good gas mileage cars best fuel economy cars
comfort	comfortable very comfortable comfortable to drive	top 10 comfortable cars comfortable cars for back pain best comfortable cars small comfortable cars most comfortable ride
fun	fun driving fun to drive easy to drive	most fun driving cars most fun to drive cars 2010 fun to drive cars fun to drive sedans
build	well built good build solid build	well built cars most well built car
reliability	reliable very reliable durable dependable	most reliable car reliable used car dependable used car most dependable cars 2008 cheap dependable cars top ten durable cars cheap durable cars high reliability cars
performance	good overall performance good performance high performance	high performance cars performance cars for sale performance cars and trucks high performance used cars high performance electric cars
interior	quiet interior comfortable interior	cars with quiet interior quiet cars 2010 most quiet cars cars with quietest rides comfortable interior cars cars comfortable seats

Hotels		
Aspect	**Sample Seed**	**Related User Queries from Google, Yahoo, Bing**
value	cheap affordable good value reasonable price	hotels downtown chicago reasonable prices cheap downtown chicago hotels cheap hotels affordable hotels in nyc good value new york city hotels good value hotels cheap very cheap hotels in new york
cleanliness	clean place clean good cleanliness	hotel nice clean cheap clean hotels nyc clean hotels in hershey pa clean hotel rooms cheap clean hotel clean hotel hong kong clean hotel singapore
room	spacious room comfortable room nice room cozy rooms	cozy hotels in chicago comfortable hotels in paris comfy hotels dublin comfortable hotel rooms in las vegas spacious hotel rooms in new york really nice hotel rooms cheap nice hotel rooms nice hotel rooms in las vegas
location	great location nice location great view nice view	great location hotels london paris hotels in great location new york hotels with great views hotels with great views in washington hotels with nice views san francisco hotels with nice views in nj
service	helpful staff polite staff good service	N/A

Table 6 Seed queries and corresponding related user queries on major search engines like Yahoo!, Bing and Google.

Top 50 query words related to cars (p=performance, g=mileage, i=interior, e=exterior, a=affordability, r= reliability)		
454 seat (i)	96 mileage (g)	35 alarms (i)
433 cheap (a)	93 diesel (g)	32 light (e)
352 muscle (e)	89 video (i)	31 speed (p)
217 hybrid (g)	79 performance (p)	31 efficient (g)
217 fast (p)	78 carseat (i)	31 compact (i)
211 seats (i)	64 safety (r)	31 cheapest (a)
190 sports (e)	64 fastest (p)	30 coupons (a)
173 gas (g)	63 small (e)	29 japanese (r)
172 electric (g)	50 convertible (e)	29 ipod (i)
171 fuel (g)	45 economy (g)	28 milage (g)
157 cool (e)	42 storage (i)	28 charger (i)
139 luxury (e)	41 alarm (i)	26 player (i)
130 stereo (i)	35 tv (i)	25 sound (i)
123 big (e)	35 miles (g)	
101 price (a)	35 dvd (i)	

Top 30 query words related to hotels (l=location, p=price, r=room, s=service)	
576 cheap (p)	38 niagara (l)
324 airport (l)	37 oceanfront (l)
305 island (l)	34 sea (l)
200 downtown (l)	32 university (l)
186 discount (p)	30 worth (p)
168 pet (s)	28 beachfront (l)
166 friendly (s)	24 romantic (l)
165 ocean (l)	24 coast (l)
161 lake (l)	23 rates (p)
113 luxury (r)	22 budget (p)
95 beach (l)	16 service (s)
78 falls (l)	16 pools (s)
52 water (l)	14 honeymoon (l)
45 jacuzzi (r)	
41 close (l)	
40 around (l)	

Table 7 List of most frequent co-occurring terms in queries "cars" and "hotels" in the Microsoft Live Labs query logs and their corresponding aspects of preferences.

views and provide a judgment score of how well the retrieved entities match the user's preferences. This would involve understanding the underlying opinions in the reviews of each retrieved entity for each aspect involved in the user's query. This process is not only time consuming but can also be overwhelming and it may be hard for the human judges to keep track of the 'key opinions'. We thus need a reasonable way to approximate human judgment. To solve this problem, we propose to leverage the existing aspect ratings that come with the user reviews in our two data sets.

Both our data sets come with free-text reviews accompanied by a set of numerical ratings on several aspects. Some of the mentions in the free-text reviews directly reflect on the aspect score that an entity receives. Figure 4 shows a car review with corresponding aspect ratings. In this review, there are mentions of the car being 'comfortable and quiet' and accordingly a very high score was given to the *comfort* aspect. There was also a mention of the 'car being not too exciting' and accordingly, a moderate rating was given to the *fun* aspect. As in most user reviews, users tend to write about aspects that stands out most to them either in a good way or a bad way. In our two data sets,

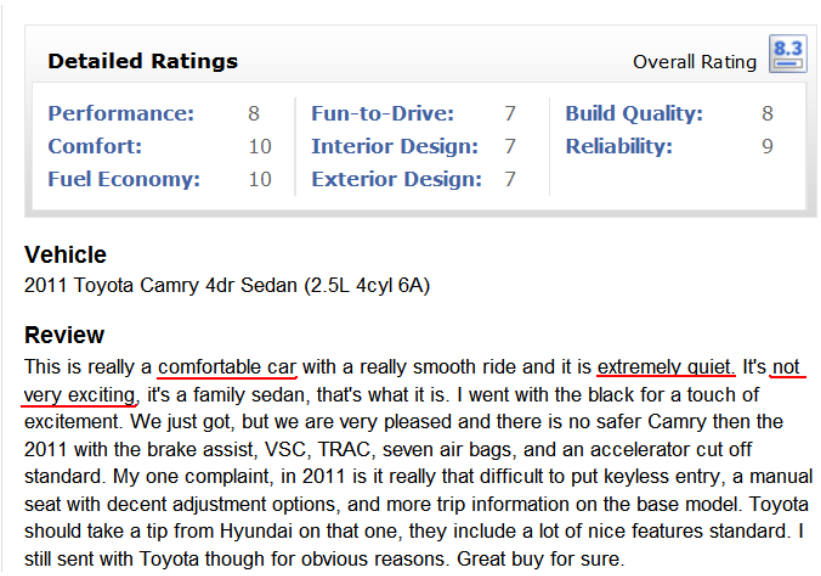

Fig. 4 A car review with accompanying aspect based score ratings. There are mentions of the car being comfortable and quiet and accordingly a high score was given to the *comfort* aspect. There is also a mention of the car not being very exciting and as can be observed only a moderate rating was given to the *fun* aspect.

users are also allowed to provide aspect scores that may be reflective of some of their free-text comments. These aspect scores can thus serve as a relevance judgment score that indicates how well an entity performs on each of its aspects. We believe that this is a good approximation to human judgment. For example, if most users find that a particular car has excellent gas mileage, then the *fuel economy* aspect would have a high aspect score. In the other extreme, apart from negative mentions about the *fuel economy*, the score for this aspect would also be low. So, if a user is looking for a car with 'very good mpg' then ideally we should return all cars that have very high scores on the *fuel economy* aspect or otherwise the system should be penalized. However, such a judgment is based on average ratings of a group of users, thus it may not reflect the real preferences of any particular user. As a result, the evaluation results using such judgments are only meaningful for relative comparison of different ranking methods, which is our goal.

Judgment scores are needed on individual aspects (to evaluate how well an entity matches one query aspect) and also on a combined set of aspects (to assess how well an entity matches the entire query). To compute judgment scores for individual aspects, we use the ratings provided by each user on a given aspect and average it. We call this score the Average Aspect Rating (AAR). For queries that span multiple aspects, we take individual AAR scores of the aspects involved and average it. This, we call the Multi-Aspect AAR (MAAR). Let $Q = Q_1, ..., Q_k$ be a query with k aspects and E be an entity. Let $r_i(E)$ be the AAR of E in aspect i. Thus, $MAAR(E, Q)$ is defined as:

$$MAAR(E, Q) = \frac{1}{k} \sum_{i=1}^{k} r_i(E)$$

We assume that an ideal ranking of entities for query Q would correspond to ranking E in the descending order of $MAAR(E, Q)$, and this enables us to quantify how close a retrieval result is to this ideal ranking.

5 Experiments

In this section, we describe our experimental setup and present the experiment results on the two test sets.

5.1 Experimental Setup

5.1.1 Evaluation Measures

Since our gold standard has multiple levels of ratings for a car, we used the Normalized Discounted Cumulative Gain (nDCG) [9] measure as the evaluation metric of our ranking task. In an opinion-based entity ranking system, only the *top-k* items ($k = 10$ in our case) that closely match the user's preferences are deemed critical. Thus, we used nDCG of the top 10 entities (denoted as nDCG@10) as a main measure.

The Discounted Cumulative Gain (DCG) accumulated at a particular rank position p is defined as:

$$DCG_p = MAAR_1(E, Q) + \sum_{i=2}^{p} \frac{MAAR_i(E,Q)}{\log_2 i}$$

To allow the DCG to be comparable across queries and search results, it is normalized by its ideal ranking, which is obtained by sorting documents based on their MAAR values available from our gold standard. Let the DCG at position p of the ideal ranking be denoted by $IDCG_p$. The nDCG is then computed as:

$$nDCG_p = \frac{DCG_p}{IDCGp}$$

5.1.2 Data Pre-processing

To evaluate the effectiveness of the proposed methods, we retained only the text segments of the reviews, dropping all HTML overhead and numerical ratings. The ratings were removed from our data set so that our experiments are in no way influenced by them. So, in essence, each document in our collection is a concatenation of text based reviews about a car/hotel. The length of each document varies greatly based on the number of reviews and also the size of individual reviews.

5.1.3 Implementation of retrieval methods

We use the three retrieval models (i.e., BM25, Language Modeling, and PL2) implemented in the Terrier 2.2 [15] toolkit for our experiments. We, however, had to make a few implementation changes to support Dirichlet Prior based Language Models [34] and fix the IDF problem of Okapi BM25 model discussed in [5].

5.2 Experiment Results

5.2.1 Standard Retrieval Models

We first look into the performance of the three state of the art standard text retrieval models. We used the default model parameters for Okapi BM25 (*b=0.75, k3=8, k1=1.2*) on both data sets as varying them did not make much difference in performance. PL2 uses a parameter c, a value for the term frequency normalization. This value was set to 1000 for both the car and hotels data set. We varied this value and found that a large value works well for the type of collection that we have. For the language modeling based retrieval, we set $\mu = 1000$ for both data sets as has been done in some previous work [33] and this value works well in our experiments.

	Hotels			**Cars**		
	PL2	**LM**	**BM25**	**PL2**	**LM**	**BM25**
StdNoFb	0.890	0.889	0.847	0.926	0.926	0.924
StdFb	0.897	0.896	0.869	0.926	0.923	0.923
change	**0.81%**	**0.74%**	**2.48%**	**-0.03%**	**-0.32%**	**-0.08%**

Table 8 nDCG@10 using standard (Std and StdNoFb) retrieval models.

The nDCG values based on 10,000 queries (for each data set) averaged across queries is reported in Table 8, where, in addition to comparing the three methods, we also compare these methods using the pseudo feedback mechanism explained in section 3. Based on Table 8, we can make several observations: (1) It appears that, overall, PL2 is most effective, followed by Dirichlet prior LM and then BM25. Interestingly, as we will show later, BM25 appears to perform the best with the proposed extensions. (2) We further see that pseudo feedback consistently helps improve the ranking of hotels but deteriorates the ranking performance of cars. Since the hotel reviews are much denser, the use of pseudo feedback is effective as the terms added to expand the query are more meaningful for the ranking process. Upon analysis of the pseudo feedback for the ranking of cars, it becomes clear why performance is degraded. For the query *'good fuel efficiency'*, some of the words added are *4cycl, jeep* and *kia*, and these words have no relation to fuel efficiency being good, resulting in the wrong cars being ranked highly. Even though pseudo feedback seems promising for this task, it only helps when the reviews are verbose. We will show later that our proposed opinion expansion is consistently effective and improves performance on both data sets.

5.2.2 Opinion Expansion

We now look into the question of whether the proposed opinion expansion method helps improve ranking accuracy. To test the idea of opinion expansion, we alter a query if it contains a praise word or an intensifier, and add the corresponding opinion synonyms to expand the query (explained in section 9). Table 9 shows the results obtained using opinion expansion on top of standard models and models that use query aspect modeling (to be discussed in the next section). From this table, it is indeed clear that opinion expansion helps all models in generating better ranking of hotels and cars. The performance improvement for BM25 is especially clear. With the use of opinion expansion, BM25 proves to be most effective amongst the three retrieval models. (We

	Hotels			Cars		
	PL2	**LM**	**BM25**	**PL2**	**LM**	**BM25**
StdNoFb	0.890	0.889	0.847	0.926	0.926	0.924
+ OpinExp	0.921	0.918	0.923	0.936	0.932	0.950
change	**3.38%**	**3.17%**	**8.18%**	**1.06%**	**0.48%**	**2.73%**
AvgScoreQAM	0.898	0.894	0.848	0.926	0.927	0.924
+ OpinExp	0.924	0.920	0.928	0.936	0.934	0.951
change	**2.77%**	**2.85%**	**8.61%**	**1.08%**	**0.67%**	**2.75%**

Table 9 nDCG@10 using opinion expansion

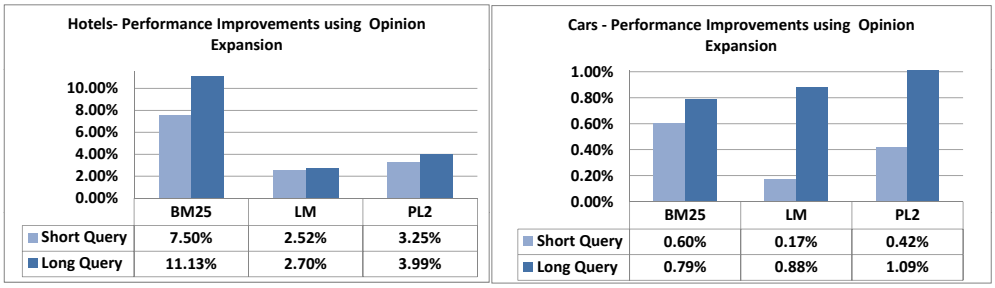

Fig. 5 Performance improvements over the AvgScoreQAM model with the use of opinion expansion for long and short queries. Better improvements are achieved on longer queries than shorter queries.

will further compare the three retrieval models in Section 5.2.4. The Wilcoxon signed rank test [31] shows that all the improvements in Table 9 are *statistically significant with a very low p-value* ($p < 10^{-6}$). This indicates that enriched opinion words in the query can indeed accommodate flexible matching of opinions, which is needed for the opinion based entity ranking task; in contrast, the standard pseudo feedback-based query expansion is only effective in some cases (see Table 8). Moreover, the improvements observed with pseudo-feedback are not as high as can be achieved with opinion expansion.

It is possible that the improvement of opinion expansion may have come from simply favoring entities with more 'positive' reviews. That is, it is possible that the System selects entities that are positive overall, which would naturally have higher MAAR scores, thus yielding better nDCG than the baseline method. To analyze the actual behavior, we look into the performances of two subgroups of queries, short queries and long queries. Short queries are those that touch 1-2 aspects, while long queries are those touching 4-5 aspects for hotels and 6-7 aspects for cars. If the System was only picking out entities that were more positive in general, the improvements on shorter queries should be just as high or in fact higher (since it is less affected by score combination across aspect queries). This is however not the case as can be seen in Figure 5. The graphs show that the improvements achieved on longer queries is considerably higher than that achieved on shorter queries, which means that the system is not just favoring entities that are simply more positive.

	Hotels			Cars		
	PL2	**LM**	**BM25**	**PL2**	**LM**	**BM25**
StdNoFb	0.890	0.889	0.847	0.926	0.926	0.924
AvgScoreQAM	0.898	0.894	0.848	0.926	0.927	0.924
change	**0.97%**	**0.58%**	**0.12%**	**0.00%**	**0.16%**	**0.00%**
StdNoFb + OpinExp	0.921	0.918	0.923	0.936	0.932	0.950
AvgScoreQAM + OpinExp	0.924	0.920	0.928	0.936	0.934	0.950
change	**0.35%**	**0.25%**	**0.58%**	**0.00%**	**0.18%**	**0.00%**

Table 10 nDCG@10 of using standard models against QAM models.

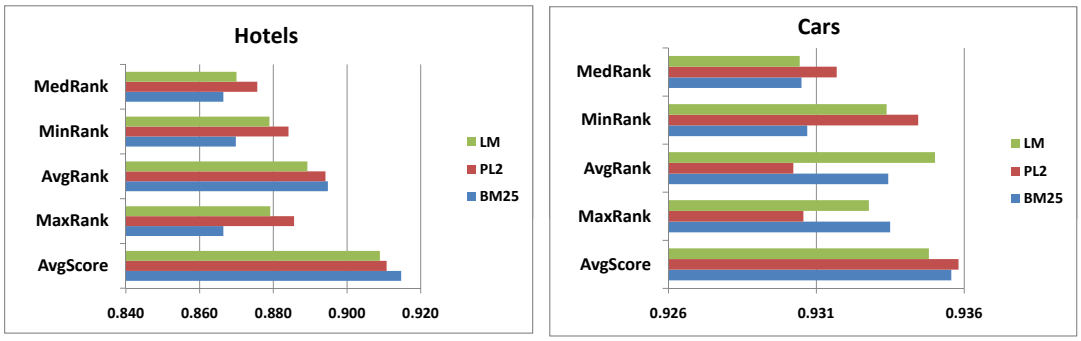

Fig. 6 nDCG@10 using different ranking strategies with QAM+OpinExp

5.2.3 Query Aspect Modeling

Another extension we proposed is to model the multiple aspects in the query explicitly and then combine the scores from multiple aspects to generate an overall score for a document. We now examine the effectiveness of this extension.

Table 10 summarizes results obtained with the query aspect modeling approach when the aggregation method is "Average Score" (i.e., $S_{AvgScore}(D,Q)$), which, as will be shown later, is the best among all the four ways of aggregation when used with opinion expansion. From this table, we see that query aspect modeling improves performance of ranking on both data set. Even though opinion expansion significantly improves the performance of the standard method (as shown in Table 9), introducing query aspect modeling provides further improvements. Wilcoxon signed rank test [31] shows that all the improvements above 0.1% in Table 10, are statistically significant with a very low p-value ($p < 10^{-6}$).

In Figure 6, we further provide a comparison of performance results using the different ranking strategies. This comparison is essential as the ranking strategy has a direct impact on how the entities are ranked. Based on this graph, we can say that the average score (AvgScore) based strategy works the best on the whole. The use of the actual ranks like AvgRank only works well in some cases as can be seen in the graph.

One advantage of our evaluation method is that we can easily analyze queries of different numbers of aspects. Since this factor is intuitively related to effectiveness of query aspect modeling, we further looked into how well the base method compares to the aspect modeling method on queries of different numbers of aspects.

Users who provide short queries are typically flexible users who have limited preferences. Queries that such users issue could be short queries like *'good mpg'*. There are

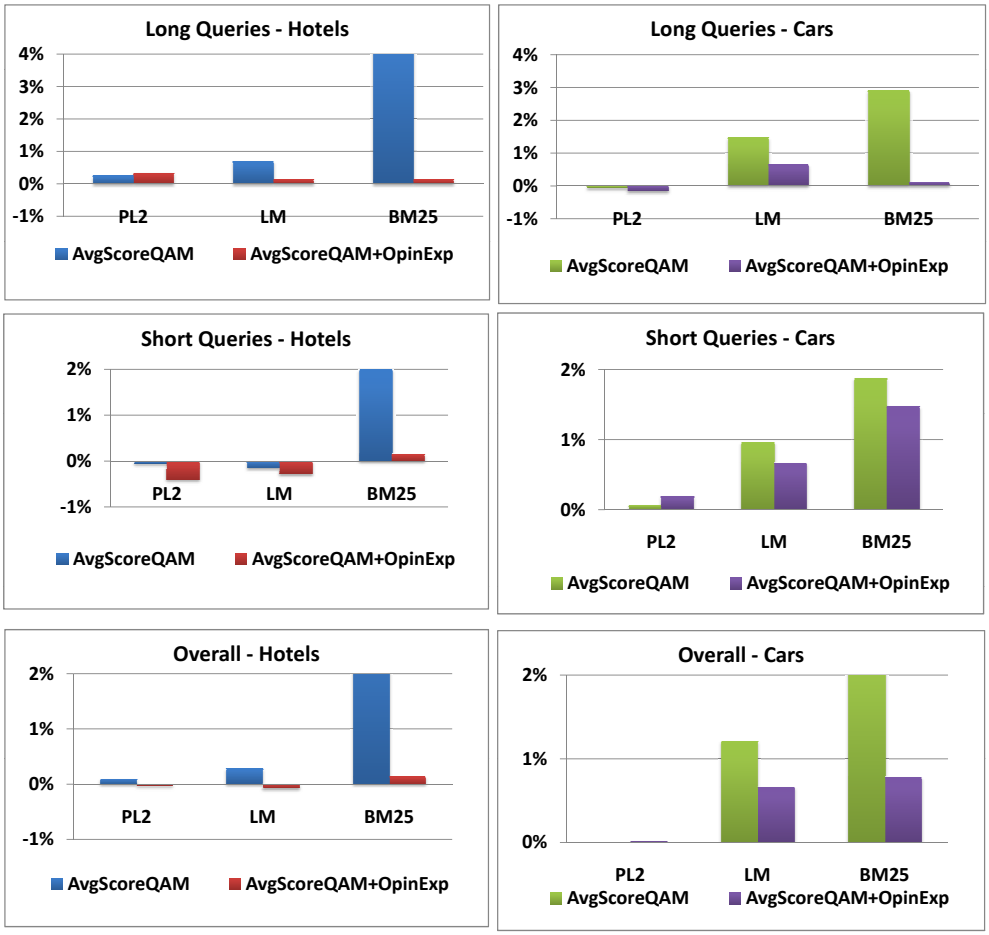

Fig. 7 Performance change of AvgScoreQAM over StdNoFb and AvgScoreQAM+OpinExp over StdNoFb+OpinExp on queries of different length

also the "picky" or "rich" users who have very specific preferences on many aspects. These users will typically issue long queries like *"excellent fuel economy, comfortable interior, solid build, highly reliable"*. For both the data sets, we manually selected some of the shortest queries (covering 1-2 aspects) and some of the longest queries (covering 6-7 aspects for cars and 4-5 aspects for hotels). We compare the performance of the QAM runs with its corresponding standard run on these queries. The percentage of change in performance is shown in Figure 7.

On the car data set, it can be seen that the aspect modeling of queries consistently yields performance improvement on very short queries. On longer queries however, performance improvements can only be seen with the LM and BM25 models. The reverse is the case for hotels. Modeling aspects in short queries seems to be effective only with BM25. On longer queries however, all three models benefit from the use of query aspect modeling. Overall, the use of QAM shows to be most beneficial with the

BM25 model with consistent performance improvements on both data sets and for both long and short queries.

5.2.4 Behavior of Retrieval Models with Opinion Expansion

While all three retrieval models show performance improvements with the use of opinion expansion, BM25 consistently outperforms its counterparts with the use of this expansion technique. To understand why, we looked deeper into the details of the rankings. Specifically, we compared these three models in two subgroups of queries (short vs. long) and three subsets of review documents with different sizes. Each city (for hotels) and model-year (for cars) has a set of review documents, where each review document represents a distinct real-world entity. For the purpose of this discussion, we will refer to all review documents in a given city or model-year as a *collection*. As shown in Table 2, each collection can have a varying size of review documents.

Figure 8 shows the performance of the AvgScoreQAM and AvgScoreQAM+OpinExp models on the hotels data set at different collection sizes for both long queries and short queries. Here, we see that for both types of queries, when no opinion expansion is used, the LM approach is most stable to variation in the collection size, but as the collection size grows, the other two models suffer a degradation in performance. In particular, BM25 is worse than the other two methods in all cases. With the use of opinion expansion, it is interesting that we now see a different pattern: the BM25 model performs the best overall, and in particular, it does much better than the other two models when the collection size is large (i.e., more entities to rank). A similar behavior was also observed with the cars data set. This means that BM25 gains much more than the other two models from opinion expansion.

Analytically, a major difference between BM25 and the other two models is that BM25 has an upper bound on the score contribution that can be made by each matched query term, no matter how frequently the term occurs in the document [21], while the other two do not have this property. Thus intuitively, BM25 would favor documents that match more query terms, while the other two models would be more prone to favoring non-relevant documents that match just a few query terms many times. Since opinion expansion would introduce many additional opinion and intensifier words, we hypothesize that the reason why BM25 gains more from opinion expansion is because PL2 and LM cannot properly handle the additional words added to the query, which could occur frequently in the review documents. The mistakes that it makes in terms of ranking become far more apparent when the collection size is large. However, with BM25, any one term's contribution to the document score cannot exceed a saturation point.

To validate this hypothesis, we looked into the result set of a query that yielded in high discrepancies in the rankings between the competing paradigms. The query is 'very clean, cozy rooms, excellent staff'. For this query, we took the first ranked entity of each result set (PL2 and LM ranked the same entity as the first) and plotted a graph that shows the query terms (after expansion), against the average term frequencies of the query terms in its corresponding entity document. The resulting graph is as shown in Figure 9. The MAAR score of the first ranked entity by PL2 and LM is 4.54 (denoted by A), while the one by BM25 is 4.83 (denoted by B). The highest MAAR from the gold standard for this query is 4.87.

Figure 9 shows that the top ranked entity by BM25 indeed has a more balanced matching of all query terms, while the top ranked entity by PL2 and LM has more

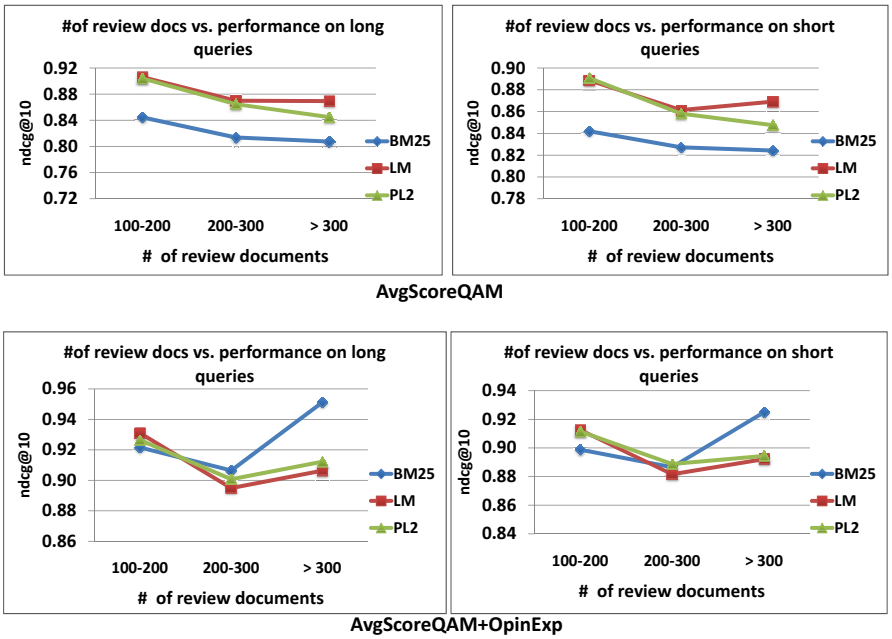

Fig. 8 Performance of **AvgScoreQAM** and **AvgScoreQAM+OpinExp** vs the number of review documents in each city from the hotels data set.

skewed frequencies of query terms. For example, A has a very large number of occurrences of the term 'very', while an important original query term 'cozy' has a very low average frequency. In contrast, B matches the query terms in a more balanced fashion, where the original query terms (labeled in the graph) and the expanded terms have average frequencies that are not extremely high or extremely low.

Such a concern about the skewness of matched query terms becomes more serious after opinion expansion as an expanded query would contain many redundant words, increasing the chance of a non-relevant document to dominate the ranking result. Similarly, when the collection size is large, the problem also becomes more serious as there is a higher chance of having such a distracting non-relevant document.

5.2.5 Influence of the availability of review data

One assumption in our problem setup is that we have enough review data to represent opinions about an entity. We now try to understand how much data we actually need to get a reasonable ranking of entities. This will also help us understand if the proposed extensions can be expected to perform better and better as we accumulate more review data. To understand this, we varied the amount of reviews used by selecting a different percentage of reviews for ranking. We ran the best performing configuration, (which by far is the AvgScoreQAM+OpinExp run) on these different sizes of reviews.

Figure 10 illustrates the performance versus the amount of review data used. Notice that for the hotels data set, the performance peaked when we used only 60%-70% of the data, after which there was a slight degradation in performance. On the car data, performance consistently improved after about 60% of the data was used.

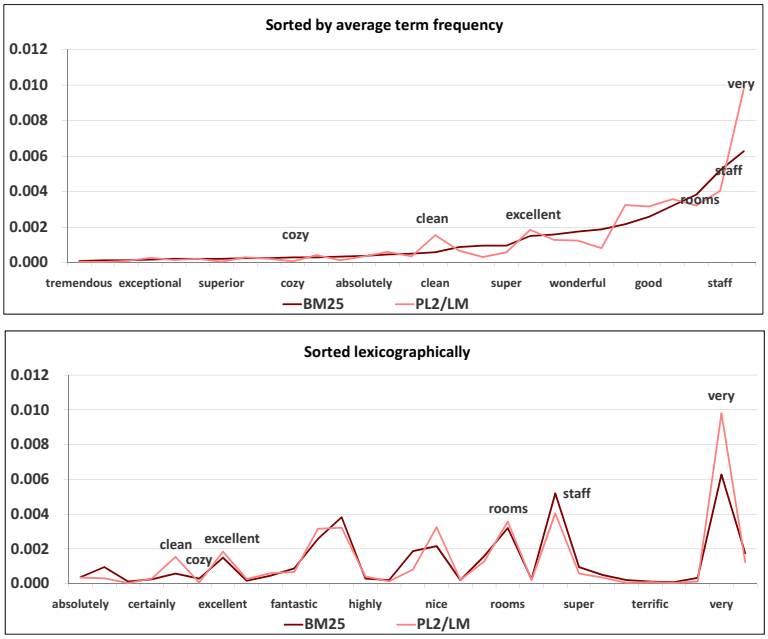

Fig. 9 Average term frequency of query words of the first ranked entity for the query 'very clean, cozy rooms, excellent staff'. The labeled terms are the original query terms. All other terms are the result of opinion expansion. PL2 and LM ranked the same entity as the first.

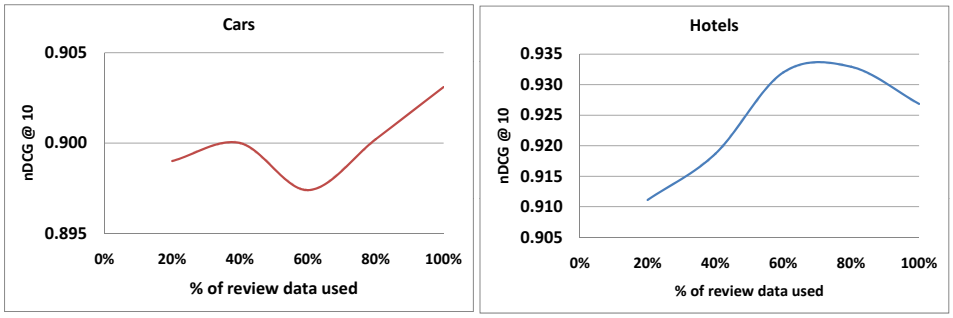

Fig. 10 Performance vs % review data used

The quick performance improvement for the hotels data set is likely due to the verbose nature of this data set. While for the car data set, due its sparse nature, almost the entire data set was needed for the performance to peak. The trend of this curve indicates that there could be more improvements if more reviews were introduced. It is possible that the quality of reviews used would also play a role in how much review data is actually needed for this task.

5.2.6 Sample Results

To illustrate some sample results of ranked hotels and cars, we show results from the two domains. First we show how a ranked list of hotels change as aspect queries are added to it. Then, we show the top ranked cars for an interesting query. The results shown were obtained using the AvgScoreQAM+OpinExp configuration.

Table 11 shows the top 10 ranked hotels in Dubai (with corresponding AAR) that match the query, *'very clean'*. Then, in Table 12, we show how this ranked list changes as a new aspect query, *'great views'* is added to the original query. From Table 11 we can see that the lowest AAR for the *cleanliness* aspect (for all hotels in Dubai), is 2.71 and the highest is 4.951. The AAR scores of all the top 10 hotels that match this query are above the average AAR for this aspect. This clearly shows that the users are indeed getting reasonable matches. However, the ordering of these entities are still not perfect. For example, the first ranked hotel, *Hatta Fort Hotel*, has an AAR score that is lower than that of *Burj Al Arab*, the hotel that ranks second in this list.

Next, when a new aspect query, *'great views'*, is added to the current query, there is a noticeable change in the ranking of hotels (as shown in Table 12). The *Burj Al Arab* which previously ranked second, now ranks first with the addition of this new aspect query. The *Le Royal Meridien Beach Resort* which ranked third, now ranks tenth in the second ranked list. The *Hatta Fort Hotel* that previously ranked first, is not even in the top 10 of this new ranked list. This is reasonable because the AAR of the *Hatta Fort Hotel* on the *location* aspect is only 4.107 compared to 4.745 for the *Burj Al Arab*. Most entities in this list have AAR scores that are well above the average in their respective aspects.

Below are some interesting review snippets for *Burj Al Arab* with regards to *cleanliness* and *location*.

"The rooms are really huge and spotlessly clean, the gym is state of the art with great sea views from the tread mills and the Spa is fantastic...."

"...The rooms are all suites and very spacious. they are all 2 floors with beautiful views.....The rooms are clean and the hotel is well situated."

"...the hotel itself is just beautiful, and in a lovely location, with fantastic views from all the floor to ceiling windows in our suite (13th fllor) across the marina..."

The second illustration of results is based on the query *'very reliable'* on the car data set, a query that most people can relate to. The top 10 cars that match this query is shown in Table 13. As can be seen in this list, the cars returned are mostly Japanese cars which are known for their reliability[12]. While these cars have high AAR scores on the *reliability* aspect, the overall ratings of these cars are not necessarily high. This shows that the system is not simply retrieving cars that are positive overall. The following snippets show some of the supporting comments for the first ranked car, *2007 Honda Accord*.

[12] http://www.independent.co.uk/life-style/motoring/motoring-news/japanese-cars-are-still-the-most-reliable-2016405.html

System Rank	Hotels	'cleanliness' AAR
1	hatta fort hotel	4.607
2	**burj al arab**	**4.920**
3	**hilton dubai creek**	**4.642**
4	**le royal meridien beach resort spa**	**4.914**
5	renaissance dubai hotel	4.600
6	the ritz carlton dubai	4.693
7	al manzil hotel	4.915
8	**le meridien dubai**	**4.586**
9	**hilton dubai jumeirah**	**4.762**
10	bel ali golf resort spa	4.620
Highest possible AAR		4.951
Lowest possible AAR		2.710
Average AAR		4.220

Table 11 Top 10 ranked hotels for the query 'very clean'. This ranking has an nDCG of 0.960. All hotels in this list have AARs above 4.5, which is above the average AAR for this aspect.

System Rank	Hotels	'cleanliness' AAR	'location' AAR
1	**burj al arab**	**4.920**	**4.745**
2	jw marriott hotel dubai	4.373	3.608
3	**hilton dubai creek**	**4.642**	**4.112**
4	al qasr at madinat jumeirah	4.833	4.817
5	mina a salam at madinat jumeirah	4.918	4.881
6	dar al masyaf at madinat jumeirah	4.951	4.848
7	grand hyatt dubai	4.895	4.289
8	**le meridien dubai**	**4.586**	**4.069**
9	**hilton dubai jumeirah**	**4.762**	**4.312**
10	**le royal meridien beach resort spa**	**4.914**	**4.694**
Highest possible AAR		4.951	4.881
Lowest possible AAR		2.710	1.900
Average AAR		4.222	3.767

Table 12 Top 10 ranked hotels for the query 'very clean' and 'great views'. This ranking has an nDCG of 0.944. The bolded hotels appear in the result set of the query 'very clean' shown in Table 11.

"...Solid, reliable car with low cost of ownership. Nice computerized maintenance notification system. Comfortable heated leather seating..."

"...I had to find something reliable, with good resale. This car is incredible....."

"...My experience with this vehicle has been as follows - the engine & transmission provide a smooth, powerful and reliable ride. The suspension is awful though..."

6 User Study

We performed a small user study to further understand the effectiveness of our proposed method in retrieving entities and also assess the effectiveness of our evaluation strategy. In this study, we asked users to judge the relevance of entities retrieved by

System Rank	Cars	'reliabillity' AAR	overall ratings
1	2007 honda accord	9.350	8.846
2	2007 honda civic	9.280	8.870
3	2007 toyota camry	9.720	8.115
4	2007 toyota yaris	9.690	9.275
5	2007 toyota corolla	9.360	8.700
6	2007 honda fit	9.580	9.079
7	2007 honda cr-v	9.380	8.933
8	2007 toyota tundra	9.170	8.871
9	2007 ford fusion	9.460	9.101
10	2007 toyota tacoma	9.090	8.790
Min		6.320	6.888
Max		9.940	9.790
Average		8.951	8.722

Table 13 Top 10 ranked cars from model-year 2007 that match the query 'very reliable'. Most cars have AAR scores that are above average.

our best performing system (BM25 with AvgScoreQAM+OpinExp). These relevance scores were then used for various analysis.

6.1 Procedure

We recruited two undergraduate students (referred to as *User1* and *User2*) who were asked to act as 'real users' of a system that enables them to search for entities based on a set of preferences. These users were presented with a query, and corresponding results (i.e. the ranked list of entities that satisfy the query) along with its respective reviews. The users were informed that the query is meant to be a set of user preferences and the entities presented as results should ideally match these preferences based on the reviews. With this in mind, for each query, the users were asked to *analyze the reviews* of the top 10 entities and then assign a relevance score to those entities based on how well it satisfies the query. This judgment is based on a 3-point rating scale defined as follows:

Score 1: Poor match. The entity does not satisfy the query well.
Score 2: Reasonable match. The entity satisfies the query reasonably well.
Score 3: Good match. The entity is a very good match for the query.

For each relevance score that the user assigns, the user was also asked to provide a brief justification for those scores. For example *Score (1) - Does not match most preferences* or *Score (2) - Matches only some preferences really well.* This study was performed on 25 queries which were randomly selected from both our car and hotel dataset. Our goal is to obtain a representative set of queries of different characteristics. In total, we had 12 long queries (touching > 2 aspects) and 13 short queries (touching 1-2 aspects). The entities presented as results were generated by our best performing system (BM25 with AvgScoreQAM+OpinExp).

	User 1	User 2
Average Rating	2.14	2.44
Std. Dev	0.40	0.25
# Entities rated 1	56	29
# Entities rated 2	110	81
# Entities rated 3	84	140
Total	**250**	**250**

Table 14 Average user judgment scores.

	User1	User2
Score (1)	-does not match one or more preference -does not match any of the preferences well	-no preference matched except one -no preferences are matched
Score (2)	-matches all preferences, but not too much -match most preferences well, but some do not match that well	-all preferences are matched, but some conflicting opinions -all preferences are matched to some extent -not much information about one preference
Score (3)	-matches all preferences well	-matches all preferences well -matches all preferences well, except one

Table 15 Summary of relevance score justification given by User1 and User2

6.2 Analysis of Relevance Ratings

In Table 14 we report the average relevance ratings assigned by User1 and User2. On average, it can be seen that both users thought that the entities retrieved by the system were a reasonable match to the queries. Notice that in the majority of cases, both users thought the entities were either a *reasonable match* (User1 - 110 entities; User2 - 81 entities) or a *good match* (User1 - 84 entities; User2 - 140 entities), rather than a *poor match*(User1 - 56 entities; User2 - 29 entities). This shows that our proposed *retrieval based method* for this special task is actually quite effective, with an average rating of above 2.0.

We further look into the entities that were assigned a low score. In Table 15, we summarize the most common justification provided by User1 and User2 on their rating assignments. As can be seen, a score of 1 is typically assigned when the reviews do not contain any mentions about one or more preferences within the query. A score of 2 is assigned when (1) there is limited evidence in the reviews about the preferences or (2) only some preferences are matched well or (3) there are conflicting opinions about a preference. A score of 3 is only assigned when most of the preferences are matched well (with sufficient evidence).

The agreement in terms of relevance ratings assigned by User1 and User2 is shown in Table 16. As can be seen, the kappa scores show that the agreement is quite low with most of the disagreement happening when the users were to choose between a rating of 2 and 3. Also, the disagreement is higher on longer queries than on shorter ones. This may be because with longer queries, we have more preference criteria, which amplify the variances of subject judgments. The results also suggest that User1 seems to have used a different rating strategy than User2 and this is also quite clear from the justification summary provided in Table 15.

Deeper analysis into the rating assignments reveals that User1's strategy is to look into both the number of matched aspects as well as how many people praised the relevant aspect. The user first checks if all preferences in the query are matched in

overall agreement				short queries				long queries			
	1	**2**	**3**		**1**	**2**	**3**		**1**	**2**	**3**
1	5	14	37	**1**	2	13	9	**1**	3	1	28
2	2	22	94	**2**	1	15	43	**2**	1	7	51
3	0	5	71	**3**	0	5	42	**3**	0	0	29
kappa	0.09			kappa	0.12			kappa	0.07		

Table 16 Agreement on relevance ratings between User1 and User2

the reviews. If all preferences are matched and if the user feels that there is 'enough' evidence for each of those preferences, then User1 assigns a rating of 3. Otherwise, the user only assigns a rating of 2. User2's strategy is to look at the bigger picture. On short queries, if all preferences are matched well then a rating of 3 is assigned. If all the preferences are matched well but there are some conflicting opinions, then a score of 2 or 1 is assigned depending on the severity of conflict. On longer queries however, if just one preference is not matched well, the entity is still considered a good match and a score of 3 is assigned. A score of 2 or 1 is only assigned when there are either conflicting opinions or more than one preference does not match well.

These differences are indeed very interesting as this tells us that different users have different criteria in judging the relevance of an entity. Some users may prefer entities ranked based on the level of evidence (positive mentions) on an aspect. Other users may prefer entities with no conflicting opinions even though not all preferences are matched well. This suggests that the ranking of entities can be further personalized according to what matters most to the user.

While the individual ratings provided by User1 and User2 do not agree all that well, it is quite possible that correlation exists in their relative preferences of entities. We thus measured rank correlation using the relevance ratings provided by both users. In particular, we computed the average Gamma correlation coefficient [25] between the rankings. The Gamma statistic was preferred over Kendall τ as ties are taken into account explicitly. Note that ties are common in the rankings of User1 and User2 as they were only allowed to use a 3-point rating scale. The correlation ranges between -1 and +1. A value of 0 means that there is no correlation; 1 is perfect positive correlation; -1 is perfect negative correlation. Based on the 25 queries, we obtained an average correlation score of 0.69. This correlation score shows that the two users actually agree reasonably well on the relative rankings of the entities even though the actual score assignments may be different.

6.3 Effectiveness of Gold Standard Rankings

In our evaluation, we have assumed that the average numerical ratings provided by review writers (on various aspects), would reflect the best ordering of entities. These ratings were thus used as the gold standard rankings. To validate this assumption, we compare the nCDG of the gold standard rankings and system rankings using the relevance ratings provided by User1 and User2. Specifically, we assume that the *actual* ideal ordering of entities is based on the ratings provided by User1 and User2 (as opposed to our gold standard rankings). Then, to compute the system nDCG, the relevance ratings provided by User1 and User2 are re-ranked according to the system rankings. Similarly, to compute the nDCG of our gold standard rankings, these relevance scores are re-ranked according to the gold rankings. The intuition here is that,

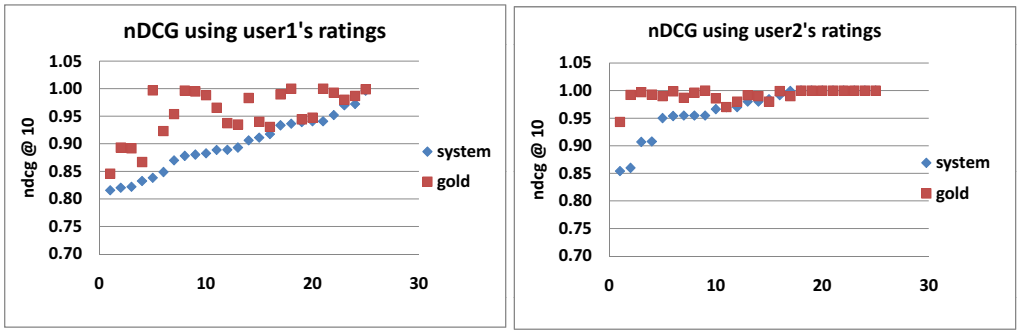

Fig. 11 nDCG @ 10 scores of system rankings and gold standard rankings using judgments provide by user1 and user2

User1		**User2**	
System Avg.	Gold Avg.	System Avg.	Gold Avg.
0.865	**0.910**	0.923	**0.950**

Table 17 Average nDCG @ 10 scores of system rankings vs. gold standard rankings using judgments provide by user1 and user2.

if our gold standard ranking is indeed an accurate measure of relevance, it should have stronger agreement with human rankings than the system rankings would. In other words, compared to the system, the gold standard should be better at recovering human rankings.

Figure 11 shows the resulting nDCG scores of system rankings and gold standard rankings using the relevance ratings provided by User1 and User2. In Table 17, we report the average scores. Based on Figure 11, we see that in many cases (especially for User1), the resulting nDCG scores of the gold standard rankings is higher than that of system rankings. The cases where the scores overlap almost perfectly was due to ties in the rankings. As an example, when a rating of 3 is assigned to all entities, this results in the same nDCG scores for both the system rankings and gold standard rankings regardless of any ordering. As can be seen, this mainly happens to entities ranked by User2. On average however (see Table 17), it is clear that the gold standard agrees more with the two users than does the system. Thus, our assumption that the average numerical ratings given by web users can be a good approximation to human judgment is indeed reasonable.

7 Discussion

Overall, our experiments show that the idea of ranking entities based on a user's keyword preferences and the opinions of other users is promising and opens up a new application area of retrieval models. Even the simple extensions that we made to the standard retrieval models have already shown promising results, and there are many possibilities to further optimize a retrieval model for this task.

In this paper, we only studied the effectiveness of our proposed method in two specific domains and on a fixed set of aspects (to facilitate evaluation). However, our idea

itself can be expanded to a variety of real world domains which includes ranking people, products, businesses and services using a set of keyword based preferences expressed on any arbitrary aspect. The basic requirement in setting up such an opinion-based entity ranking system is the need for a large number of opinion containing documents. For example, using all the mentions about different politicians in blog articles, news articles from CNN[13] and BBC[14] and micro-blogging sites such as Twitter, we can rank these politicians based on a user's preferences. These preferences can be attributes such as 'honest' and 'liberal' or the politician's promises such as 'better health care plan' and 'against child abortion', etc. Similarly, using all the reviews from e-commerce sites like Amazon.com[15], BestBuy.com[16] and Walmart.com[17], we can rank products based on the user's preferences. For example, if the user is interested in purchasing a laptop, the user could find laptops based on his/her personal tradeoffs using a set of keywords such as 'lightweight', 'bright screen','highly reliable', 'long battery life' and so on. Thus, instead of reading many reviews for a large number of laptops (to check if the laptop actually satisfies the user's preferences), the entity ranking system tries to shortlist a set of laptops that match these preferences. With this, the user would only need to analyze the laptops ranked by the system.

In terms of accepting a user's preferences, different types of user interfaces may be used. The most general interface would be a single text field that would allow users to express preferences using a natural keyword query. Aspects in the query can then be obtained using query segmentation techniques. Another approach is to ask users to specify a special delimeter to separate their preferences. While this would require just one additional character between two preferences, users could find this requirement rather unnatural to their usual browsing and searching pattern. A more practical user interface would be to provide separate text fields to represent the different preferences. While all these are reasonable suggestions, the question with regards to the best user interface for an entity ranking task such as this remains open until a full user study has been performed.

Our use of retrieval models for this task represents a shallow but general solution to the problem. If we assume that users will only express preferences on a set of common aspects, then it is possible to leverage existing work in rating prediction [30, 13, 26] to rank entities more accurately based on a user's preferences. Although such a refined approach could lead to more accurate ranking, as we have mentioned in Section 2, these approaches pose practical limitations. With the rating prediction approach, scaling up to different domains would involve a lot more text processing compared to our retrieval based approach. For example, aspect discovery in each domain would be a necessity and once found, users are tied to these limited number of aspects. Further, the rating prediction approaches require some form of supervision such as the presence of overall ratings, which severely limits the type of textual content that can be utilized.

[13] www.cnn.com
[14] www.bbc.com
[15] www.amazon.com
[16] www.bestbuy.com
[17] www.walmart.com

8 Conclusions and Future Work

In this paper, we proposed a novel way of utilizing opinion data - that is to directly rank entities like people, businesses and products based on a user's preferences and existing opinions on those entities. We studied the use of several state-of-the-art retrieval models for this task and propose some new extensions over these models. We also leverage rating information associated with some car and hotel reviews to create a benchmark data set for quantitative evaluation of opinion-based entity ranking.

Experimental results show that the use of opinion expansion is especially effective for improving the ranking of entities according to the user's preferences. We also show that the aspect modeling of queries as opposed to treating queries as set of keywords, is effective on longer queries. While all three state-of-the-art retrieval models show improvement with the proposed extensions, the BM25 retrieval model is most consistent and works especially well with these extensions.

Our evaluation, in two very different domains (cars and hotels), shows that the proposed methods can be directly applied to rank different types of entities for which we have reviews available. We thus believe that this is a very promising line of study with good prospects of practical applications. Our user study shows that the ranking results of entities from the proposed methods have high NDCG values based on human judgments and can be very useful for users to help them choose entities based on opinions.

Our work opens up many interesting future research directions. First, in this paper, we only explored techniques that are unique to the problem of opinion-based entity ranking. We believe that many of the existing techniques and refinements in information retrieval especially in areas like expert finding can further help in improving the performance of this task. Also, in both query aspect modeling and opinion expansion, we explored some simple ideas in this paper. The fact that these simple techniques are effective suggests that more sophisticated methods such as structured query language models [35] and sentiment analysis techniques can be potentially leveraged to further improve performance. The data set and evaluation methodology introduced would greatly facilitate further exploration in this direction.

Second, it would be very interesting to study how to obtain further clarification from users about their preferences through opinion feedback; for example, a user can indicate which query aspect is already matched well and which is still unsatisfactory, and the system can learn from such feedback to improve ranking.

References

1. G. Amati and C. J. van Rijsbergen. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Trans. Inf. Syst.*, 20(4):357–389, 2002.
2. K. Balog, L. Azzopardi, and M. de Rijke. A language modeling framework for expert finding. *Inf. Process. Manage.*, 45(1):1–19, 2009.
3. J. P. B. H. I. O. D. Hannah, C. Macdonald. University of Glasgow at TREC2007: Experiments in Blog and Enterprise tracks with Terrier. In *Proceeddings of the 16th Text REtrieval Conference (TREC 2007)*, 2007.
4. K. Dave, S. Lawrence, and D. M. Pennock. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *WWW '03: Proceedings*

of the twelfth international conference on World Wide Web, pages 519–528. ACM Press, 2003.

5. H. Fang, T. Tao, and C. Zhai. A formal study of information retrieval heuristics. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–56, New York, NY, USA, 2004. ACM Press.
6. H. Fang and C. Zhai. Probabilistic models for expert finding. In *ECIR*, pages 418–430, 2007.
7. M. Gamon. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *Proceedings of the 20th international conference on Computational Linguistics*, page 841, Geneva, Switzerland, 2004. Association for Computational Linguistics.
8. B. He, C. Macdonald, J. He, and I. Ounis. An effective statistical approach to blog post opinion retrieval. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 1063–1072, New York, NY, USA, 2008. ACM.
9. K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.
10. J. Koren, Y. Zhang, and X. Liu. Personalized interactive faceted search. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 477–486, New York, NY, USA, 2008. ACM.
11. B. Krulwich and C. Burkey. The contactfinder agent: Answering bulletin board questions with referrals. In *AAAI/IAAI, Vol. 1*, pages 10–15, 1996.
12. J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 111–119, New York, NY, USA, 2001. ACM.
13. Y. Lu, C. Zhai, and N. Sundaresan. Rated aspect summarization of short comments. In *Proceedings of the 18th international conference on World wide web*, pages 131–140, Madrid, Spain, 2009. ACM.
14. T. Nasukawa and J. Yi. Sentiment analysis: capturing favorability using natural language processing. In *Proceedings of the 2nd international conference on Knowledge capture*, pages 70–77, Sanibel Island, FL, USA, 2003. ACM.
15. I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald, and C. Lioma. Terrier: A High Performance and Scalable Information Retrieval Platform. In *Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006)*, 2006.
16. B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *In Proceedings of the ACL*, pages 271—278, 2004.
17. B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*, pages 115–124, 2005.
18. B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 79–86, 2002.
19. J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281,

New York, NY, USA, 1998. ACM.
20. R. Prabowo and M. Thelwall. Sentiment analysis: A combined approach. *Journal of Informetrics*, 3(2):143–157, April 2009.
21. S. Robertson. The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.
22. S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *TREC*, pages 0–, 1994.
23. E. Sadikov, J. Madhavan, L. Wang, and A. Halevy. Clustering query refinements by user intent. In *WWW '10: Proceedings of the 19th international conference on World wide web*, pages 841–850, New York, NY, USA, 2010. ACM.
24. G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. pages 355–364, 1997.
25. S. Siegel and Castellan. *Nonparametric statistics for the social sciences*. McGraw-Hill, New York, 1988.
26. B. Snyder and R. Barzilay. Multiple aspect ranking using the good grief algorithm. *In Proceeddings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL*, pages 300–307, 2007.
27. B. Tan and F. Peng. Unsupervised query segmentation using generative language models and wikipedia. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 347–356, New York, NY, USA, 2008. ACM.
28. D. Tunkelang. *Faceted Search*. Morgan and Claypool Publishers, 2009.
29. P. D. Turney and M. L. Littman. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Trans. Inf. Syst.*, 21(4):315–346, 2003.
30. H. Wang, Y. Lu, and C. Zhai. Latent aspect rating analysis on review text data: a rating regression approach. In *KDD '10: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 783–792, New York, NY, USA, 2010. ACM.
31. F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.
32. K. Yang, N. Yu, A. Valerio, H. Zhangm, and W. Ke. Fusion approach to finding opinions in blogosphere. *ICWSM*, 2007.
33. C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 334–342, New York, NY, USA, 2001. ACM.
34. C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, 2004.
35. C. X. Zhai. *Statistical Language Models for Information Retrieval*. Morgan & Claypool, 2009.