Rest-Lens: An Experience Aware Restaurant Search System

Submission ID: 12473

Abstract

Restaurant search and selection are common parts of consumer decision making. Existing platforms support search through factual filters such as cuisine, location, price, and ratings. However, users often need to inspect review details to determine whether candidate restaurants actually meet their needs, since factual filters may be inaccurate, incomplete, or outdated. In this paper, we present Rest-Lens, an interactive interface for restaurant search. Rest-Lens uses LLMs to identify needs in natural language queries and translate them into editable search conditions. It also searches reviews for information related to these needs and organizes relevant review text into a review graph. Through two use cases and participant feedback, we show that Rest-Lens helps users refine search conditions, understand retrieved results, and use review information as further reference.

Keywords: Restaurant review, natural language interface, large language model, knowledge graph.

1. Introduction

Restaurant search and selection are common parts of consumer decision making (Chua et al., 2020). Platforms such as Google Maps and Yelp help users narrow down restaurant options using *factual filters*, including cuisine, location, price, and ratings (Li & Hecht, 2021). However, selecting a restaurant depends not only on factual filters but also on review information (Bu et al., 2021). For a query such as "a quiet Chinese restaurant with good dumplings," existing platforms may return relevant restaurants, yet it is often desirable for users to read reviews and assess how well candidate restaurants meet their needs.

Prior work has studied restaurant search and selection from several related directions. Some systems support restaurant search through predefined factual filters (Doan et al., 1996; Medynskiy et al., 2009). These factual filters are often provided by businesses or platforms, but they may be incomplete or outdated. (Glaeser et al., 2022) Therefore, users often need to read reviews to decide whether candidate restaurants meet their needs, especially for experiential needs such as "delicious gluten-free desserts" or "easy parking." Prior work has developed methods to identify *aspects* and *opinions* in customer reviews (Cai et al., 2021). For example, in "delicious Asian food," "Asian food" is the aspect and "delicious" is the opinion. However, this line of work primarily extracts aspects and opinions from reviews rather than linking them to users' needs. Recent work also explores how large language models (LLMs) can support restaurant search with natural language queries (Liu et al., 2024). However, existing approaches often fail to clearly show how user needs are transformed into factual filters or how relevant review information is linked to user needs. This limitation becomes more prominent in group restaurant search, where different members may add needs over several rounds. For example, one person may ask for "a quiet Chinese restaurant," another may add "good dumplings," and a third may add "easy parking." As these needs are added, the group goal becomes clearer, As these needs are added, the group goal becomes clearer. However, users may still need to read through many reviews to determine whether each candidate restaurant meets the group's combined needs.

In this paper, we propose Rest-Lens, an interactive interface for restaurant search. Rather than asking LLMs to directly search restaurants, our system uses LLMs to help users inspect and refine how their needs are

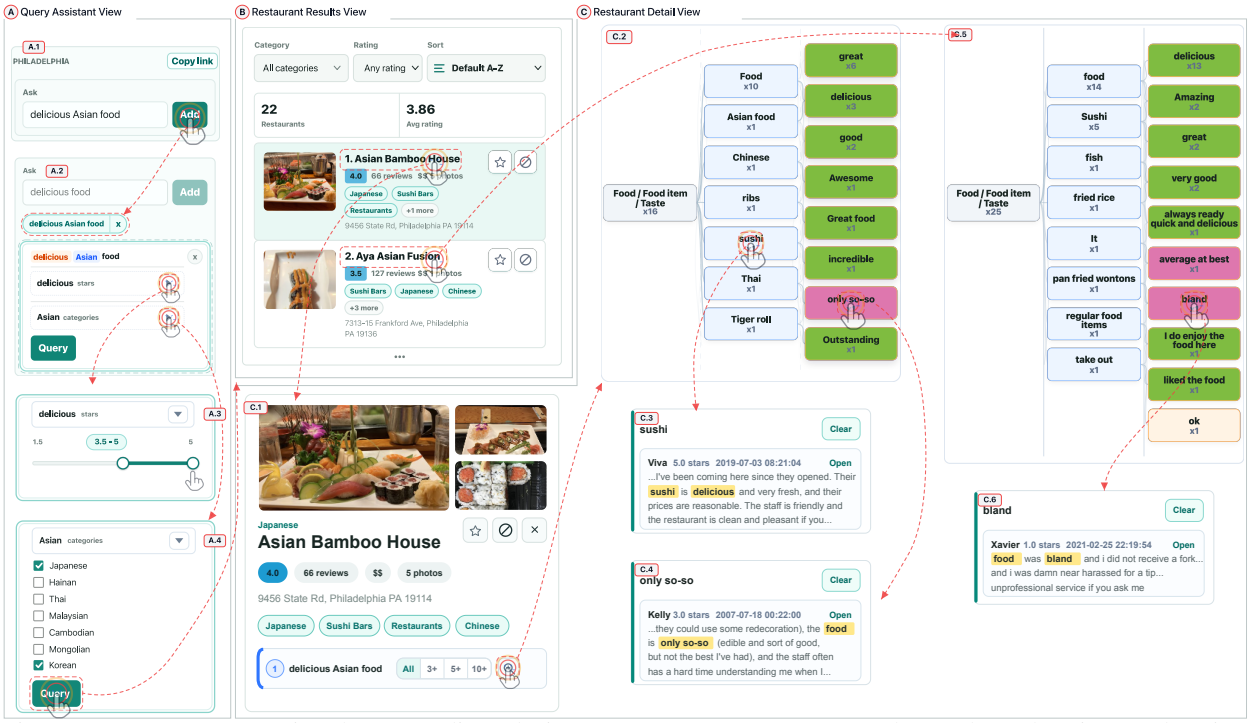

Figure 1: Rest-Lens contains three coordinated views to support restaurant search, result exploration, and review analysis: (A) Query Assistant View, enables users to input human query by specifying constraints such as food type, cuisine, and rating; (B) Restaurant Results View, displays the retrieved restaurants with metadata including ratings, review counts, price levels; and (C) Restaurant Detail View, presents the retrieved restaurant with an interactive review graph, where clicking nodes such as "sushi," "only so-so," and "bland" reveals the corresponding review snippets.

translated into search conditions. Users enter queries in natural language, and the system identifies the needs expressed in the query and explicitly presents these identified needs. It then turns these needs into editable search conditions. When a need can be transformed into factual filters, such as `stars`, `cuisine`, or `price`, Rest-Lens suggests candidate options and allows users to refine them before searching.

A key design principle of Rest-Lens is to connect restaurant search with customer experiences reported in reviews. In addition, for each retrieved restaurant, Rest-Lens searches its reviews for information related to the needs in a query. The system then organizes relevant review information into a review graph for each restaurant. In the graph, user needs are linked to related review text. This representation enables users to use review information as an additional reference, see how reviews discuss each need, and refer back to the original reviews. Our contributions are as follows:

- We introduce Rest-Lens, an interactive restaurant search system that uses LLMs to help users identify needs in human queries and translate them into editable search conditions.
- We design a review graph that links user needs to relevant review text, helping users see how reviews discuss those needs for each candidate restaurant.
- Through two use cases and participant feedback, we demonstrate the utility of Rest-Lens in supporting users as they refine search conditions, understand retrieved results, and use review information as further reference.

2. Related Work

2.1. Restaurant Search

Restaurant search systems vary in the types of user input they support and in how they interpret that input as search conditions. Early systems mainly relied on manual filters, where users directly selected factual filters such as cuisine, price, rating, and parking. For example, Plaisant et al. (1999) presented a prototype in which users could manually adjust factual filters to search for a restaurant. Faceted search made such filters visible and easy to revise during exploration (Hearst, 2006; Medynskiy et al., 2009). Such interfaces make search conditions visible, but they are primarily designed to address needs that can be expressed clearly.

Natural language interfaces support a different form

of input. Users can describe their needs in their own words through the interfaces, even when those needs are not fully specified. The system then interprets the input and transforms it to search conditions. The earlier system used a pipeline approach to transform natural language queries to structured fields, including food type, price range, area, and opening hours (Banchs & Kim, 2014). Wen et al. (2017) used neural networks to infer search conditions from user queries. Recent advances in LLMs have introduced new ways to interpret natural language in restaurant search. For example, SUQL uses LLMs to interpret restaurant search queries and explain the search conditions used by the system (Liu et al., 2024).

Prior work has shown that natural language queries can be ambiguous, which may cause a system to interpret user needs in ways that do not match the user's intent (Gao et al., 2015). Therefore, clarification is important when a query is underspecified (Aliannejadi et al., 2019). To address this problem, Rest-Lens uses LLMs to identify needs in a human query and make the interpretation visible and editable. It converts needs that correspond to factual filters into editable search conditions and supports refinement when a need has multiple possible interpretations.

2.2. Restaurant Review Analysis and Graph Visualization

Restaurant reviews can be transformed into structured information and visualized to help users understand restaurants. Prior work has studied how to extract fine-grained review elements, including aspects, aspect categories, and opinions, from restaurant reviews (Cai et al., 2021; Chebolu et al., 2022; Zhang et al., 2023). These studies show how reviews can be used to identify what customers mention and how they evaluate different aspects of a restaurant.

Once review information is organized around aspects and opinions, it can help users explore restaurants. Review Spotlight (Yatani et al., 2011) provides an interface that summarizes reviews with frequent adjective-noun word pairs, such as comments about food, service, and place, and lets users inspect the original sentences that contain selected pairs. RevMiner (Huang et al., 2012) extracts structured information from reviews to support restaurant search, comparison, and inspection. These systems demonstrate the value of reviews in understanding restaurants, but they primarily rely on review data rather than factual filters for retrieval. In Rest-Lens, when a user's needs can be matched to factual information, the system first uses factual information to retrieve candidate restaurants. It then uses reviews to help users explore among the candidates.

Because review information may span multiple sentences, Rest-Lens organizes it into a review graph. Graph visualizations are commonly used to represent entities and relations and to support the exploration of connected information (Herman et al., 2000). In Rest-Lens, the review graph links user needs with review categories, aspects, opinions, and original review text. Similar categories, aspects, and opinions are grouped to keep the graph compact. This representation helps users use review information as further reference.

3. Design Goals

We developed the design goals for Rest-Lens through an examination of existing restaurant search platforms, realistic restaurant selection scenarios, and grounding our work in the relevant literature (Hearst, 2006; Liu et al., 2024; Medynskiy et al., 2009; Yatani et al., 2011). Existing restaurant search platforms, such as Google Maps and Yelp, support factual filters and human queries. However, they often provide limited visibility into how broad or ambiguous queries are interpreted, how users can refine search conditions over time, and how the candidate restaurants are relevant to the review information. They also provide limited support for sharing a search session with others in a lightweight and anonymous way. These limitations become more important in group restaurant selection, where different members may add, revise, or remove needs across multiple rounds of discussion. As the search goal changes, users need to understand the current search conditions, the candidate restaurants, and the review information behind the results.

Based on these observations, we formulate four design goals for Rest-Lens.

DG1: Support editable query interpretation and clarification. The system should allow users to enter restaurant needs in natural language and see how the system interprets them. When a need is broad or ambiguous, the system should present possible interpretations for users to confirm or revise. The confirmed interpretation should then be translated into editable search conditions for restaurant retrieval.

DG2: Support anonymous session sharing. The system should allow users to share a restaurant search session through a copied anonymous link. The shared session should preserve the current queries, search conditions, and candidate restaurants, so that other group members can view the same search state without account-based collaboration.

DG3: Support iterative search refinement. The

system should allow users to add, inspect, revise, and remove needs and search conditions during the search process. This supports restaurant selection as an iterative process, in which users may start with broad needs and refine them as their goal becomes clearer.

DG4: Support detailed restaurant examination through reviews. The system should help users explore candidate restaurants using review information. Users should be able to see how each restaurant relates to their needs, examine supporting review snippets, and trace the information back to the original reviews.

4. Method

Rest-Lens is an interactive system that uses an LLM to support restaurant search. It interprets users' natural language queries and provides relevant aspects and opinions from reviews that match their needs. We use `gpt-oss-120b` (OpenAI, 2025) as the LLM and employ few-shot prompting (Brown et al., 2020).

4.1. Data Processing

This subsection describes Rest-Lens's data preparation. It includes three parts: selecting restaurant data, organizing business records, and extracting categories, aspects, and opinions from reviews.

Selecting Restaurant Data. We use the Yelp Open Dataset (Yelp Inc., 2026), which provides real-world business records, reviews, and attributes such as hours, parking, and ambience (Sun et al., 2023; Yelp Inc., 2026). The full dataset contains 150,346 businesses and 6,990,280 reviews. To keep the analysis manageable, we focus on Philadelphia, which has the largest number of business records in the dataset, with 14,569 businesses. Since this study focuses on restaurant search, we keep only open businesses labeled as restaurants. This step yields 3,525 restaurants and 511,138 reviews. We further select restaurants with the `Chinese` cuisine label. The final subset contains 300 restaurants and 47,314 reviews. Because Yelp uses multi-label cuisine types, these restaurants may also have related labels, such as `Korean` or `Japanese`. Keeping these multi-label restaurants allows Rest-Lens to consider other cuisine labels when translating user queries.

Organizing Business Records. The original Yelp business records are stored in JSON format. We convert them into a relational table, where each row represents a restaurant, and each column represents a business attribute. The resulting business table contains 52 columns. This table allows Rest-Lens to translate a user query into SQL filters.

Extract Information from Reviews. To structure the review text, we extract review records based on three fields: `category`, `aspect`, and `opinion`, which respectively capture the aspect type, the specific item or attribute, and users' evaluation.

For example, in the sentence "The dumplings were delicious, but the service was slow," `dumplings` and `service` are aspect mentions. Following the aspect category classification proposed by (Panchendrarajan et al., 2017), these aspects can be transformed to broader categories such as `Food` and `Service`. The words `delicious` and `slow` are opinion expressions. Thus, this sentence can produce two review records: one about the taste of dumplings and one about service quality.

We use an LLM with few-shot prompting to extract these records (Brown et al., 2020). For each review, the LLM identifies aspect mentions, transforms them to predefined categories, and extracts the related opinion expressions. This process produces 180,711 review records for the Chinese restaurant subset.

4.2. Query Processing and Review Linking

Fig. 2 shows how Rest-Lens translates a human query to target restaurants with relevant reviews. We use the Chinese restaurant dataset, which we have cleaned in Section 4.1 to illustrate this procedure.

Problem. Consider a restaurant search scenario on Yelp. A group of users enters a short query such as "delicious Asian food". Yelp returns a ranked list of restaurants on the map without asking for clarification. In this search, the first result is "Kissho House" with a 4.4-star rating, while the second result is "Jeong's Noodle" with a 4.5-star rating.

This ranking is hard to interpret because the lower-rated restaurant appears first. Users may not know whether Yelp interprets "good" as star rating, popularity, distance, or a mix of hidden signals. They may also not know how Yelp interprets "Chinese", since it could refer to Chinese restaurants only or related categories such as dim sum, noodles, or hot pot.

System workflow. Rest-Lens helps users clarify restaurant search needs through interaction. Given a user query, it suggests relevant business fields, such as `stars`, `cuisine`, or `price`, and candidate values from the business table. After user refinement and confirmation, Rest-Lens then uses the confirmed conditions to retrieve restaurants. Business records are used for filtering as they provide factual information for retrieval. Reviews are later used as a reference for the retrieved restaurants.

Step 1: Identify column fields. Given a user query, Rest-Lens uses an LLM with few-shot prompt engineering to identify business fields that may represent the user's needs. The prompt includes the Yelp

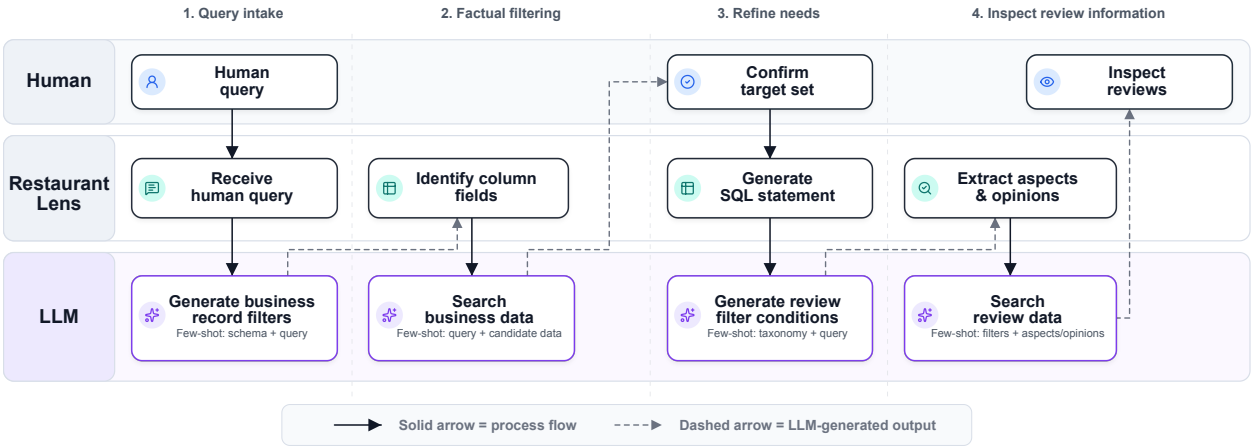

Figure 2: Overview of the query translation process in Rest-Lens. The system identifies factual filters that may represent a human query, asks users to clarify their needs, creates the target restaurant set with SQL, and then connects the results with relevant reviews.

Table 1: Example transforming from a user query to business record fields.

User need	Factual filter field	Candidate value
`delicious`	`stars`	From 3.5 to 5
`Asian`	`cuisine`	`Asian Fusion`

business record schema and the user query. The output is a set of search conditions, including the user's need, the matched business field, and a candidate value.

Consider the user query: delicious Asian food. Table 1 shows the structured output produced by the LLM. Each row shows how a need in the user query corresponds to a business record field and a candidate value. Here, the need `delicious` corresponds to the `stars` field. Since `stars` stores numerical values, the candidate value is a range from 3.5 to 5. The need `Asian` corresponds to the `categories` field. Since `categories` stores category labels, the candidate value is `Asian Fusion`.

Step 2: Refine candidate values with user input. After Step 1, the transformation may still need user refinement before retrieval.

For a numerical field, Rest-Lens can directly compute the valid range from the data. For example, after `delicious` is transformed to `stars` field, the user can define the final condition, such as `stars >= 3.5`.

For a string field, the initial candidate value may be incomplete. In this example, the LLM transforms `Asian` to the `categories` field and gives `Asian Fusion` as the candidate value. However, the same field may also contain related values, such as `Chinese`, `Korean`, and `Pan Asian`. These values may match the user's intent but not be included in the first output.

To address this issue, Rest-Lens uses another LLM prompt that includes the user's need, the matched field, and the unique values in that field. The LLM selects related string values and returns them to the user. The user then selects the values that match their intent.

Step 3: Retrieve the target restaurant set with SQL. After the user confirms the search conditions, Rest-Lens translates them into SQL and queries the restaurant business table. Conditions from different needs are combined with `AND`, while multiple selected values within the same field are combined with `OR`.

For the query "delicious Asian food," suppose the user selects restaurants with at least 3.5 stars and chooses `Korean` and `Japanese` as the final category values. The SQL statement has the following form:

```
SELECT *
FROM restaurants
WHERE stars >= 3.5
  AND ( categories = 'Korean' OR
        categories = 'Japanese' );
```

After executing the above statement, Rest-Lens retrieves restaurants that satisfy both confirmed conditions. Each returned restaurant must have at least 3.5 stars and must include at least one of the selected category values.

Step 4: Link review information to retrieved restaurants. After retrieving the target restaurant set, Rest-Lens links these restaurants to review records related to the user's needs, helping users understand whether the retrieved restaurants are related to the query.

For the query "delicious Asian food," the confirmed business record filters return 22 restaurants. Rest-Lens then searches only the reviews associated with these 22 restaurants.

This step uses two LLM prompts and requires no additional user input. The first prompt includes the category taxonomy and the user query. It transforms the query to a review category and generates initial candidates for aspects and opinions. For "delicious

Asian food," the prompt transforms the category to `Taste`, with aspect candidates such as `Asian food` and opinion candidates such as `delicious` and `great`.

The second prompt refines the aspect and opinion candidates using the distinct `aspect` and `opinion` values from reviews of the 22 target restaurants. In this example, the refined aspect values include `Asian food`, `Thai`, and `sushi`. The refined opinion values include `delicious`, `tasty`, and `great`.

Finally, Rest-Lens uses the refined aspect and opinion values to retrieve matching review records from the 22 target restaurants. In this example, it retrieves records whose category is `Taste` and whose `aspect` or `opinion` matches the refined values. These records are then used as relevant review information for the retrieved restaurants.

4.3. Interface

As shown in Fig. 1, Rest-Lens provides three coordinated views to support restaurant search: Query Assistant View, Restaurant Results View, and Restaurant Detail View. Together, these views guide users through three steps: entering their search needs, refining these needs into search conditions, and inspecting review information for candidate restaurants.

Query Assistant View. The Query Assistant View helps users express their needs in human query and turn them into editable search conditions. Users can enter a query such as "delicious Asian food" (Fig. 1(A.1)). They can also use the *Copy Link* button to anonymously share the current search state, including the query and selected conditions. The system then presents the identified needs in a query card. For example, Fig. 1(A.2) shows that "delicious" and "Asian" are identified as two needs. Users can inspect and refine how each need is represented as a search condition. When a need can be linked to factual filters, the system shows candidate filters and lets users adjust their values before searching. For example, "delicious" can be represented by the `stars` filter, and users can set a condition such as `stars` > 3.5 using a slider (Fig. 1(A.3)). Similarly, "Asian" can be represented by the `cuisine` filter, where users can select values such as `Japanese`, `Korean`, or `Thai` (Fig. 1(A.4)). After users confirm the conditions, the system retrieves candidate restaurants and related review information.

Restaurant Results View. The Restaurant Results View presents the restaurants that match the confirmed search conditions (Fig. 1(B)). In this view, users can browse the candidate restaurants and refine the result set by sorting them by rating, review count, or the default order, and by applying filters such as minimum star rating and cuisine. Each restaurant row provides an overview of the restaurant, including its photo, name, rating and so on. This view helps users quickly visit candidates and select a restaurant to inspect further in the Restaurant Detail View.

Restaurant Detail View. The Restaurant Detail View helps users examine whether a selected restaurant satisfies the needs expressed in the query. After users select a restaurant, such as *Asian Bamboo House*, the system opens a detail page with the restaurant profile, including photos, cuisine labels, rating, and address (Fig. 1(C.1)). Users can then enlarge the review graph to inspect review information for the selected restaurant.

The review graph summarizes customer reviews in a structured and traceable way (Fig. 1(C.2)). It contains three types of nodes: review categories, aspects, and opinions. For example, an aspect node such as *sushi* indicates that customers discussed sushi in their reviews, while an opinion node such as *only so-so* reflects an opinion about the restaurant. The number on each node, such as "x3", indicates how many times the node appears in the reviews. To make the graph easier to read, similar aspect and opinion expressions are grouped together using character bigrams, and the Sørensen-Dice similarity is computed (Maciejewski et al., 2025).

Users can click graph nodes to view the original reviews. For example, clicking *sushi* or *only so-so* shows the corresponding review snippets and original review text in Fig. 1(C.3–C.4).

Because the review graph is generated for each restaurant, different restaurants may exhibit distinct review patterns. For example, when users select *Aya Asian Fusion*, the system presents a different review graph based on its reviews (Fig. 1(C.5)). If users click the opinion node *bland*, they can inspect the original review text linked to this opinion (Fig. 1(C.6)). In this way, the Restaurant Detail View helps users inspect candidate restaurants using review information, rather than relying only on factual filters such as cuisine, price.

5. Case Study

5.1. Case Study 1: Group Selection

A group of four coworkers in Philadelphia, Emily, Jason, Olivia, and Daniel, are interested in choosing a Chinese restaurant after work. They want a place with good food, convenient parking, an acceptable noise level, and dumplings. Instead of writing one complete query at the beginning, they use Rest-Lens to add their needs one by one.

Emily begins by entering a simple query:

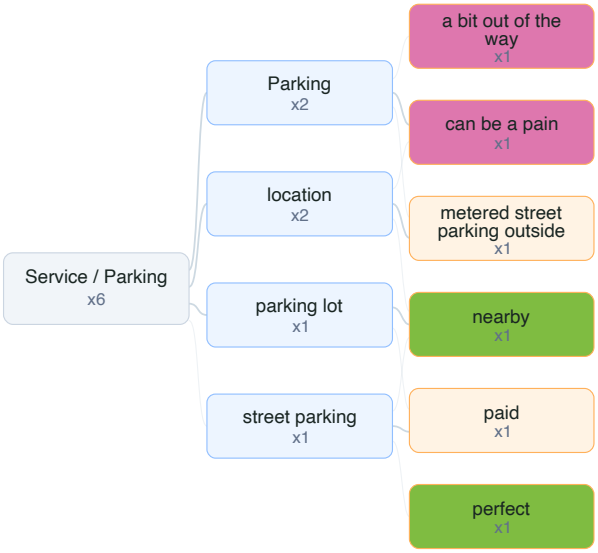

Figure 3: Parking review graph for *Dim Sum House*.

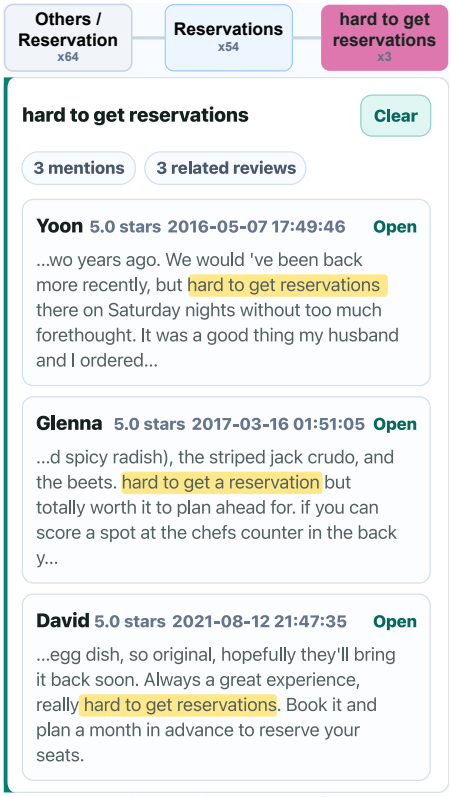

Figure 4: Reservation review graph for *Vernick Food & Drink*.

{good Chinese place}. In response, Rest-Lens identifies two needs from the query: "good" and "Chinese." The system considers "good" to be the `stars` field and considers "Chinese" to be cuisines. Emily confirms a `3.5--5` star range and refines the cuisine condition to `Szechuan`. After applying these conditions, the candidate set is reduced from 300 Chinese restaurants to 15 restaurants. Then Emily click copy link to share this website to Jason.

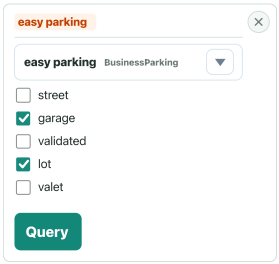

Figure 5: Parking filter interface.

Jason then adds his need {easy parking} through the interface (Fig. 5). The system considers this need in the `BusinessParking` field and suggests parking options. Jason selects `garage` and `lot`, reducing the candidate set to 8 restaurants.

Olivia follows up with the phrase: {not too loud}. The system considers this need in the `NoiseLevel` field. Olivia selects `average` as the preferred noise level, and the system updates the results again. Six restaurants remain after this condition is applied.

Daniel then adds the last need: {delicious dumplings}. Rest-Lens suggests `Dim Sum` as a related cuisine and also searches the reviews for dumpling-related information. After Daniel confirms this interpretation, three restaurants remain: *Bai Wei*, *Dim Sum House*, and *DJ Kitchen*.

The group then compares the three restaurants using the review information panel. *Bai Wei* satisfies the `Szechuan` and `Dim Sum` conditions, but a review notes difficult parking. *Dim Sum House* has positive dumpling reviews, but its review graph shows paid or inconvenient parking (Fig. 3). *DJ Kitchen* has a 4.5-star rating, positive dumpling reviews, and free parking behind the restaurant. Based on the confirmed search conditions and the linked review information, the group selects *DJ Kitchen*.

5.2. Case Study 2: Business Dinner Selection

Four coworkers in Philadelphia, Mia, Kevin, Sarah, and Noah, need to choose an American restaurant for a business dinner with visiting collaborators. Their needs include cuisine, ambience, alcohol availability, reservations, and overall quality, and are incrementally identified and added to the queries during discussion.

Mia begins by entering a simple query: {American restaurant for business} In response, Rest-Lens identifies two needs from the query: "American" and "for business." The system considers "American" to `cuisine` and suggests American-related values, such as `American (New)` and `American (Traditional)`. It also considers "for business" to `Ambience`, since a business dinner depends on the restaurant setting. Mia confirms

`American (New)` and selects business-appropriate ambience values, including `classy` and `casual`. After applying these conditions, the result set contains 64 restaurants with an average rating of 3.66.

Kevin then adds another need: nice alcohol. Rest-Lens considers this needs to the `Alcohol` field and presents the available values: `full_bar`, `none`, and `beer_and_wine`. Because the group wants drinks available during dinner, Kevin keeps `full_bar` and `beer_and_wine`, and removes `none`. After this update, the candidate set is reduced to 44 restaurants with an average rating of 3.74.

Sarah follows up with the phrase: accepts reservations and all the restaurants accept reservations. Since the remaining list is still too large for discussion, Noah adds a minimum rating condition of 4.5 stars. After this final filtering step, five restaurants remain: *SouthHouse*, *Glory Beer Bar & Kitchen*, *Jansen*, *Original 13 Ciderworks*, and *Vernick Food & Drink*.

The group then compares the five restaurants using the review information panel. Sarah focuses on reservation details as she is coordinating the dinner. *Vernick Food & Drink* satisfies all confirmed conditions. But as shown in Fig. 4, its review graphs describe it is `hard to get reservations`.

Based on this review information, the group treats *Vernick Food & Drink* as a reservation-risk candidate. They hide *Vernick Food & Drink* from the active shortlist and continue comparing the remaining four restaurants.

6. Human Subjective Feedback

We conducted six remote online study sessions to collect feedback on the system's usability and usefulness (P1–P6). Each session lasted about 45 minutes, including a 10-minute demonstration of the main workflow and 35 minutes of free exploration, during which participants provided think-aloud feedback (Ericsson & Simon, 1984). We manually recorded the feedback and coded the comments into high-level themes (Boyatzis, 1998). The quotes below are translated and lightly edited from our notes.

Editable Search conditions. Participants (P1–P4) valued features that made the system's query clarification visible and editable. P3 liked that one ambiguous need could be mapped to multiple candidate values, saying that this made the system "simple and useful to use." Besides this, P1 focused on broad food terms, explaining that a word such as "noodles" may refer to ramen, pasta, or other noodle dishes. P1 said, "When I type noodles, the system shows choices such as ramen and pasta, so I can choose more precisely."

Review information. Participants (P2, P4–P6) also valued the review graph as it connected the retrieved restaurants. P2 said that the graph made the relation between user needs and reviews easy to understand: "I can directly see how the reviews are related to what I asked for." P2 also found the `category`, `aspect`, and `opinion` structure easy to read, as it separated what the review was about from how the reviewer evaluated it. P4 and P5 both emphasized the importance of linking graph nodes back to original reviews. P4 said, "Clicking aspect and jumping to the specific review is useful because I can check the review directly." P5 further noted that review information can be more useful than static business labels, "business tags may be outdated, but reviews are closer to recent customer experiences."

Suggestion and Future Work. Participants also suggested features to make the system more useful for restaurant searches. P2 wanted region-based search and said "Users may want to search within a specific area." P3 suggested "Voice input for more natural interaction." P4 praised the system's value and said, "I really like this system. It feels very valuable, and honestly, I think it could become something much bigger, like a browser plugin or a mobile app that people could use in their daily work." P6 focused on the scale of the review graph. P6 said, "The small knowledge graph for each restaurant is already very helpful. It gives users a clear way to understand what people are saying about that restaurant. In the future, I think it would be even more useful if the system could also connect the knowledge graphs of all the restaurants that users have selected, rather than only adding review information to the graph of a single restaurant."

Overall, the feedback suggests that Rest-Lens helped users clarify ambiguous search needs, use review information as further reference, and assess restaurant matches through review information. Participants also pointed to several improvements, including region and distance search, voice input, more accurate evidence matching, and larger-scale review graphs across restaurants.

7. Discussion, Implication, and Future Work

Discussion. Recent platforms (e.g., Google Maps), have begun to support restaurant searches using AI to process human queries. These platforms can suggest restaurants and provide review summaries, showing that AI is becoming part of restaurant search platforms.

Rest-Lens differs from these platforms because it focuses on transparency and user interaction. Rather than asking LLMs to directly search restaurants,

Rest-Lens uses LLMs to help users inspect how their needs are identified and translated into search conditions. After users enter a human query, the system turns it into editable search conditions. When a need can be considered to factual filters, users can refine the options in factual filters before searching.

Rest-Lens also helps users connect search results with review information. Existing platforms may provide useful summaries, but it is more desirable to inspect detailed reviews to make informed decisions. To address this issue, Rest-Lens builds a review graph for each restaurant, in which user needs are linked to related review text, highlighting the relevant information that users want to know.

In addition, the design is useful for group restaurant search, where different members may incrementally add their needs. By keeping these needs visible and linking them to editable search conditions and relevant review text, Rest-Lens supports an interactive search process.

Furthermore, our system fits the needs of restaurant selection. We decided not to use the automatic ranking method for decision making. For this reason, Rest-Lens suggests rather than decides on the restaurant. Instead, it helps users explore candidate restaurants by showing how each restaurant aligns with their needs using search needs and review information.

Implication. We believe that our design is applicable to other search platforms that combine factual filters with review information, such as e-commerce, hotel booking, travel search, and local service platforms. In these settings, users often provide a human query input, such as "a durable backpack for travel" or "a quiet hotel for a business trip." Some needs in the input can be transformed to structured search conditions (e.g., `category`, `location`) while other needs require unstructured customer reviews, such as durability, quietness, service quality, or ease of use.

A similar system could be extended to other search tasks that use review information. The main implication is that LLMs can help make search more transparent by showing how user needs are interpreted, how search conditions are formed, and how candidate items match those needs through review information.

Limitations and Future Work. In our current system, each restaurant has its own review graph. We can explore how to visualize a review graph across all retrieved restaurants. A larger graph could help users compare restaurants by showing needs, aspects, opinions, and reviews across restaurants. However, this is challenging, as the same aspect may appear in reviews of different restaurants and carry subtly different implications. We cannot naively merge the same aspects into the same node across two restaurants, as they may sell different dishes, such as "Korean fried chicken" and "American southern fried chicken". The cuisine of restaurants and the searching context should be considered when merging aspects in the knowledge graph. Future work should design an algorithm to decide when and under which conditions the same aspect across restaurants can be merged.

Another direction for future work is to examine how the system should respond when the search conditions in collaborative user queries contain conflicts. In some cases, no results may be returned because the full set of criteria cannot be satisfied simultaneously. A knowledge graph could provide a more effective way to visualize such situations, particularly by revealing which constraints prevent the query from being fulfilled. For example, certain branches of the knowledge graph could be pruned to show how the available results change when one condition is relaxed or removed. This would make it possible to illustrate how many restaurants can still satisfy the remaining requirements after a specific criterion is excluded. Such a visualization could help users better understand trade-offs among conflicting preferences and support more flexible decision-making.

Long customer reviews often discuss multiple aspects and opinions, while providing only one overall rating. Future work could investigate how to infer aspect-level sub-ratings for each aspect–opinion pair and how these sub-ratings contribute to the overall rating. For example, a customer may appreciate a restaurant's ambience but be dissatisfied with its service. Separating these aspect-level evaluations would help capture the nuanced structure of customer feedback more accurately.

References

Aliannejadi, M., Zamani, H., Crestani, F., & Croft, W. B. (2019). Asking clarifying questions in open-domain information-seeking conversations. *SIGIR'19*, 475–484.

Banchs, R. E., & Kim, S. (2014). Spoken dialogue system for restaurant recommendation and reservation. *Proceedings of the 15th Annual Conference of the International Speech Communication Association (INTERSPEECH 2014)*, 1488–1489.

Boyatzis, R. E. (1998). *Transforming qualitative information: Thematic analysis and code development*. SAGE Publications.

Brown, T. B., Mann, B., Ryder, N., et al. (2020). Language models are few-shot learners. *NeurIPS*, *33*, 1877–1901.

Bu, J., Ren, L., Zheng, S., Yang, Y., Wang, J., Zhang, F., & Wu, W. (2021). ASAP: A Chinese review dataset towards aspect category sentiment analysis and rating prediction. *Proceedings of NAACL-HLT '21*, 2069–2079.

Cai, H., Xia, R., & Yu, J. (2021). Aspect-Category-Opinion-Sentiment quadruple extraction with implicit aspects and opinions. *Proceedings of ACL-IJCNLP '21*, 340–350.

Chebolu, S. U. S., Rosso, P., Kar, S., & Solorio, T. (2022). Survey on aspect category detection. *ACM Computing Surveys*, *55*(7), 1–37.

Chua, B.-L., Karim, S., Lee, S., & Han, H. (2020). Customer restaurant choice: An empirical analysis of restaurant types and eating-out occasions. *International Journal of Environmental Research and Public Health*, *17*(17), 6276.

Doan, K., Plaisant, C., & Shneiderman, B. (1996). Query previews in networked information systems. *Proceedings of the Third Forum on Research and Technology Advances in Digital Libraries (ADL '96)*, 120–129.

Ericsson, K. A., & Simon, H. A. (1984). *Protocol analysis: Verbal reports as data*. MIT Press.

Gao, T., Dontcheva, M., Adar, E., Liu, Z., & Karahalios, K. G. (2015). DataTone: Managing ambiguity in natural language interfaces for data visualization. *ACM UIST '15*, 489–500.

Glaeser, E. L., Kim, H., & Luca, M. (2022). Nowcasting the local economy: Using Yelp data to measure economic activity. In *Big data for twenty-first-century economic statistics* (pp. 249–273). University of Chicago Press.

Hearst, M. A. (2006). Design recommendations for hierarchical faceted search interfaces. *Proceedings of the SIGIR 2006 Workshop on Faceted Search*, 26–30.

Herman, I., Melançon, G., & Marshall, M. S. (2000). Graph visualization and navigation in information visualization: A survey. *IEEE TVCG*, *6*(1), 24–43.

Huang, J., Etzioni, O., Zettlemoyer, L., Clark, K., & Lee, C. (2012). RevMiner: An extractive interface for navigating reviews on a smartphone. *ACM UIST'12*, 3–12.

Li, H., & Hecht, B. (2021). 3 stars on Yelp, 4 stars on Google Maps: A cross-platform examination of restaurant ratings. *Proceedings of the ACM on Human-Computer Interaction*, *4*(CSCW3), 1–25.

Liu, S., Xu, J., Tjangnaka, W., Semnani, S., Yu, C., & Lam, M. (2024). SUQL: Conversational search over structured and unstructured data with large language models. *Findings of the Association for Computational Linguistics: NAACL 2024*, 4535–4555.

Maciejewski, J., Nikoletos, K., Papadakis, G., & Velegrakis, Y. (2025). Progressive entity matching: A design space exploration. *Proceedings of the ACM on Management of Data*, *3*(1), Article Article 65, 1–25.

Medynskiy, Y., Dontcheva, M., & Drucker, S. M. (2009). Exploring websites through contextual facets. *ACM CHI '09*, 2013–2022.

OpenAI. (2025, August). Introducing gpt-oss [Accessed: 2026-06-04]. https : / / openai . com/index/introducing-gpt-oss/

Panchendrarajan, R., Ahamed, N., Sivakumar, P., Murugaiah, B., Ranathunga, S., & Pemasiri, A. (2017). Eatery: A multi-aspect restaurant rating system. *Proceedings of the 28th ACM Conference on Hypertext and Social Media*, 225–234.

Plaisant, C., Shneiderman, B., Doan, K., & Bruns, T. (1999). Interface and data architecture for query preview in networked information systems. *TOIS*, *17*(3), 320–341.

Sun, K., Hu, Y., Ma, Y., Zhou, R. Z., & Zhu, Y. (2023). Conflating point of interest (POI) data: A systematic review of matching methods. *Computers, Environment and Urban Systems*, *103*, 101977.

Wen, T.-H., Vandyke, D., Mrkšić, N., Gašić, M., Rojas-Barahona, L. M., Su, P.-H., Ultes, S., & Young, S. (2017). A network-based end-to-end trainable task-oriented dialogue system. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 438–449.

Yatani, K., Novati, M., Trusty, A., & Truong, K. N. (2011). Review Spotlight: A user interface for summarizing user-generated reviews using adjective-noun word pairs. *ACM CHI '11*, 1541–1550.

Yelp Inc. (2026). Yelp open dataset [Accessed: 2026-05-24]. https://business.yelp.com/data/resources/open-dataset/

Zhang, W., Li, X., Deng, Y., Bing, L., & Lam, W. (2023). A survey on aspect-based sentiment analysis: Tasks, methods, and challenges. *TKDE*, *35*(11), 11019–11038.