Multi-Aspect Reviewed-Item Retrieval via LLM Query Decomposition and Aspect Fusion

Anton Korikov[1,*,†], George Saad[1,†], Ethan Baron[1], Mustafa Khan[1], Manav Shah[1] and Scott Sanner[1]

[1]University of Toronto, Toronto, Canada

Abstract

While user-generated product reviews often contain large quantities of information, their utility in addressing natural language product queries has been limited, with a key challenge being the need to aggregate information from multiple low-level sources (reviews) to a higher item level during retrieval. Existing methods for reviewed-item retrieval (RIR) typically take a late fusion (LF) approach which computes query-item scores by simply averaging the top-K query-review similarity scores for an item. However, we demonstrate that for *multi-aspect queries* and *multi-aspect items*, LF is highly sensitive to the distribution of aspects covered by reviews in terms of aspect frequency and the degree of aspect separation across reviews. To address these LF failures, we propose several novel aspect fusion (AF) strategies which include Large Language Model (LLM) query extraction and generative reranking. Our experiments show that for imbalanced review corpora, AF can improve over LF by a MAP@10 increase from 0.36 ± 0.04 to 0.52 ± 0.04, while achieving equivalent performance for balanced review corpora.

Keywords

Dense retrieval, query decomposition, multi-aspect retrieval, LLM reranking, late fusion,

1. Introduction

User-generated reviews are an abundant and rich source of data that has the potential to be used to improve the retrieval of reviewed-items such as products, services, or destinations. However, a challenge of using review data for retrieval is that information has to be aggregated across multiple (low-level) reviews to a (higher) item-level during retrieval. Recent work [1], defining this Reviewed-Item Retrieval setting as RIR, showed that state-of-the-art results could be achieved by using a bi-encoder to aggregate review information to an item-level in a process called late fusion (LF). As opposed to aggregating review information to an item-level *before* query-scoring (early fusion), LF first computes query-review similarity to avoid losing information before scoring, and then averages the top-K query-review similarity scores to get a query-item similarity score. Recently, LF has been implemented by retrieval augmented generation (RAG) driven conversational recommendation (ConvRec) systems for generative recommendation, explanation, and interactive question answering [2].

In this paper, we extend RIR to a multi-aspect retrieval setting, formulating what we call multi-aspect RIR (MA-RIR). In this problem, our goal is to retrieve relevant items for a multi-aspect query by using the reviews of multi-aspect items. Specifically, for an item with multiple aspects, we assume that each review describes at least one, and up to all, of the item's aspects.

As our primary contributions:

- We formulate the MA-RIR problem and identify failure modes of LF under imbalanced review-aspect distributions, considering imbalances due to both aspect frequency and the degree of aspect separation across reviews.
- We propose several novel aspect fusion strategies, which include LLM query extraction and reranking, to address failures of LF review-score aggregation on imbalanced multi-aspect review distributions.
- We leverage a recently released multi-aspect retrieval dataset, Recipe-MPR [3], with ground-truth query- and item- aspect labels to generate four multi-aspect review distributions with various aspect balance properties, and numerically evaluate the effect of review-aspect balance on MA-RIR.
- Our simulations show that for imbalanced data, Aspect Fusion can improve over LF by MAP@10 increase from 0.36 ± 0.04 to 0.52 ± 0.04 while achieving equivalent performance for balanced data.
- We show that LLM reranking in both cross-encoder and zero-shot (ZS) listwise reranking settings can provide some improvements when given a large enough number of reviews, but risk decreasing performance when not enough reviews are provided.

2. Background

2.1. Neural IR

Given a set of documents $\mathcal{D}$ and a query $q \in \mathcal{Q}$, an IR task $IR\langle \mathcal{D}, q \rangle$ is to assign a similarity score $S_{q,d} \in \mathbb{R}$ between the query and each document $d \in \mathcal{D}$ and return a ranked list of top scoring documents. The standard first-stage neural-IR method [4] for a large corpus is to first use a bi-encoder $g(\cdot) : \mathcal{Q} \cup \mathcal{D} \rightarrow \mathbb{R}^m$ to map a query q and document d to their respective embeddings $g(q) = \mathbf{z}^q$ and $g(d) = \mathbf{z}^d$. A similarity function $f(\cdot,\cdot) : \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}$, such as the dot product, is then used to compute a query-document score $S_{q,d} = f(\mathbf{z}^q, \mathbf{z}^d)$. For web-scale corpora, exact similarity search for the top query-document scores is typically impractical, so approximate similarity search algorithms [5] are used instead.

SIGIR'24 Workshop on Information Retrieval's Role in RAG Systems, July 18, 2024, Washington D.C., USA.

*Corresponding author.

†These authors contributed equally.

anton.korikov@mail.utoronto.ca (A. Korikov); g.saad@mail.utoronto.ca (G. Saad); mr.khan@mail.utoronto.ca (M. Khan)

0009-0003-4487-9504 (A. Korikov); 0009-0000-3549-9874 (G. Saad); 0009-0004-2461-5760 (E. Baron); 0009-0008-3622-7270 (M. Khan); 0009-0008-4728-0771 (M. Shah); 0000-0001-7984-8394 (S. Sanner)

2.2. Reviewed-Item Retrieval

2.2.1. Problem Formulation

Information retrieval across two-level data structures was previously studied by Zhang and Balog [6]. Specifically, Zhang and Balog define the *Object Retrieval* problem, where (high-level) objects are described by multiple (low-level) documents. Given a query, the task is to retrieve high-level objects by using information in the low-level documents.

To investigate a special case of object retrieval where the goal is retrieving items (e.g., products, destinations) based on their reviews, Abdollah Pour *et al.* [1] recently proposed the Reviewed-item Retrieval (RIR) problem. In the $RIR\langle \mathcal{I}, \mathcal{D}, q \rangle$ problem, there is a set of items $\mathcal{I}$, where every item i is a high-level object. Each item is described by a set of reviews (i.e., "low-level documents") $\mathcal{D}_i \subset \mathcal{D}$, and the r'th review of item i is $d_{i,r} \in \mathcal{D}_i$. The main difference between RIR and Object Retrieval is that in RIR a low-level document $d_{i,r}$ cannot describe more than one high-level object i, while Object Retrieval allows for more general two-level structures. Given a query $q \in \mathcal{Q}$ and a score $S_{q,i}$ between q and each item i, the goal of RIR is to retrieve a ranked list L^q of top-K_I scoring items:

$$L^q = (i_1, ..., i_{K_I}) \quad \text{s.t. } i_1 \in \arg\max_i \{S_{q,i}\}$$
$$S_{q,i_k} \geq S_{q,i_{k+1}}, \quad \forall i_k \in L^q.$$

2.2.2. Fusion

To get a query-item score $S_{q,i}$ using an item's review set $\mathcal{D}_i$, review information needs to be aggregated to an item level: this process is called *fusion*. Two alternatives exist for fusion [6]: if low-level information is aggregated *before* a query is used for scoring, it is called Early Fusion (EF) — in contrast, if the aggregation occurs *after* query-scoring, it is called Late Fusion (LF).

For EF in RIR, Abdollah Pour *et al.* [1] experiment with mean-pooling and contrastive learning methods to create an item embedding $\mathbf{z}^i \in \mathbb{R}^m$ from review embeddings $\{\mathbf{z}^d\}_{d \in \mathcal{D}_i}$. They then directly compute the similarity between $\mathbf{z}^i$ and a query embedding $\mathbf{z}^q$ as the query-item score $S_{q,i} = f(\mathbf{z}^q, \mathbf{z}^i)$.

For LF in RIR, these authors first compute query-review similarity scores $S_{q,d_{i,r}} = f(\mathbf{z}^q, \mathbf{z}^{d_{i,r}})$. They then aggregate these scores into a query-item score $S_{q,i}$ by averaging the top K_R query-review scores for each item:

$$S_{q,i} = \frac{1}{K_R} \sum_{r=1}^{K_R} S_{q,d_{i,r}}. \tag{1}$$

Numerical evaluations performed for EF and LF for RIR demonstrate that EF has significantly worse performance than LF [1], and Abdollah Pour *et al.* conjecture that EF performs worse because it loses fine-grained review information before query-scoring. In contrast, by delaying fusion, LF preserves review-level information during query-scoring. Due to these findings, we do not study EF for MA-RIR, rather, we focus on developing Aspect Fusion as an extension of LF, discussed next.

3. Multi-Aspect Reviewed Item Retrieval

3.1. Multi-Aspect Queries

This paper focuses on retrieving relevant items using their reviews for a multi-aspect query, such as "*Can I have a **meatball** recipe that **doesn't take too long**?*". We define a query aspect to be a sub-span of a multi-aspect query that represents a distinct topic (or facet) in the query, for instance the sub-spans "*"meatball"* and *"doesn't take too long"* in the previous sentence. While there is ambiguity in identifying which sub-spans, if any, in a query should be considered aspects, this sub-span based definition is a simple way to represent aspects and is conducive to overlap-based evaluations of aspect extraction such as intersection-over union (IOU). Formally, we denote the set of aspects in query q as $\mathcal{A}_q^{\text{query}}$, where the j'th query aspect is $a_{q,j}^{\text{query}} \in \mathcal{A}_q^{\text{query}}$. In this work, multi-aspect queries are assumed to be logical AND queries for all aspects, though an aspect itself can represent other logical operators such as XOR (e.g. a query aspect may be "*chicken or beef*"). Finally, we assume all query aspects are equally important — a further discussion of weighted multi-aspect retrieval can be found in Section 7.

3.2. Multi-Aspect Reviewed-Items

In addition to considering multi-aspect queries, we also consider multi-aspect items described by reviews. For instance, a multi-aspect item that is relevant to the multi-aspect query example above might be a recipe titled "*Beef **meatballs** cooked in canned soup, **ready in 25 minutes***". However,

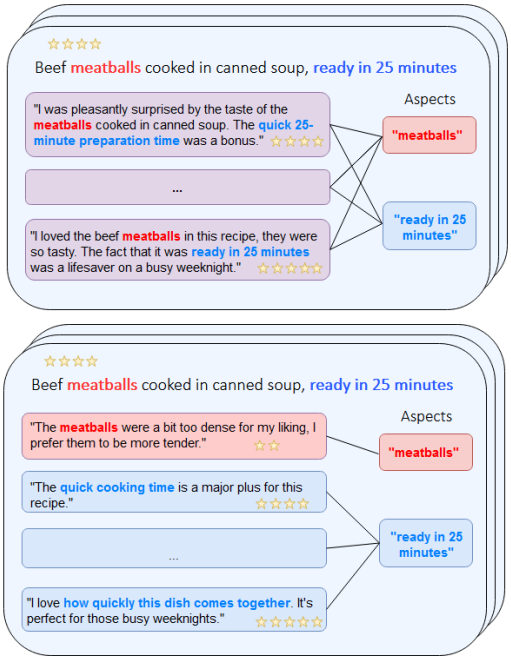

Figure 1: Two extremes of item aspect distributions, showing reviews for an item with aspects "meatballs" and "ready in 25 minutes": a) *Fully overlapping* (top) — Each review mentions all item aspects. b) *Fully disjoint with imbalanced aspect frequency* (bottom) — no review mentions more than one aspect, and some aspects are mentioned much more frequently than others.

since our goal is to isolate the properties of review-based retrieval, we assume that no such natural language (NL) item-level description is available. Instead, we assume that the item's aspects are described in reviews. Obviously, item-level descriptions (e.g. titles) *are* often available in practice, so a prime direction for future work is fusion across multiple levels of NL data during reviewed-item retrieval.

Examples of reviews describing the item in the previous paragraph, which has aspects "*meatballs*" and "*ready in 25 minutes*", are shown in Figure 1. In this paper, we assume that a review $d_{i,r}$ must mention at least one item aspect $a^{\text{item}}_{i,j} \in \mathscr{A}^{\text{item}}_i$ and could mention up to all item aspects. Formally, the distribution of item aspects across reviews can be defined with a bipartite aspect distribution graph $\mathscr{G} = \{\mathscr{D}, \mathscr{A}^{\text{item}}, \mathscr{E}\}$ where an edge $(d_{i,r}, a^{\text{item}}_{i,j}) \in \mathscr{E}$ exists if review $d_{i,r} \in \mathscr{D}_i$ mentions aspect $a^{\text{item}}_{i,j} \in \mathscr{A}^{\text{item}}_i$. We also let $\mathscr{A}^{\text{rel},q}_i \subseteq \mathscr{A}^{\text{item}}_i$ represent the set of item-aspects that are relevant to a query and should be considered during retrieval. We define the $MA-RIR\langle \mathscr{A}, \mathscr{E}, \mathscr{D}, q\rangle$ problem as the task of retrieving a ranked list of relevant multi-aspect items L^q for a multi-aspect query q, where $\mathscr{A} = \mathscr{A}^{\text{item}} \cup \mathscr{A}^{\text{query}}$.

3.3. Multi-Aspect Review Distributions

As we will demonstrate with numerical simulations on LLM-generated review data, understanding review distributions in terms of *aspect frequency* and *degree of aspect separation* between reviews is key to designing successful MA-RIR techniques. Figure 1 shows two extremes of aspect distributions that are among the distributions we explore in our experiments.

3.3.1. Fully Overlapping Distributions

Figure 1a) shows a *fully overlapping* aspect distribution where *each* review mentions *all* aspects — in this case, the bipartite graph $\mathscr{G}$ (see the RHS of Figure 1) is fully connected for item i_1. This is the most balanced review aspect distribution possible for an item, and, because of this "perfect" aspect balance, we postulate that aspect-agnostic retrieval approaches such as standard LF will perform competitively on such distributions.

3.3.2. Degree of Separation and Aspect Frequency

In contrast to the case of perfect review-aspect balance, Figure 1b) shows an extreme case of aspect imbalance. Firstly, one aspect is mentioned much more frequently than another — this is an aspect frequency imbalance. Secondly, *each* review mentions *only one* aspect — this is a maximal degree of separation of aspects across reviews (fully disjoint). Mathematically, $\mathscr{G}$ has $|\mathscr{A}^{\text{item}}_{i_1}|$ (disjoint) star components where some stars have a singificantly higher degree than others. In the next section, we discuss the negative effects of imbalanced review-aspect distributions on LF performance on MA-RIR, and propose aspect fusion as a method for mitigating these negative effects.

4. Aspect Fusion for MA-RIR

4.1. Desiderata of Aspect Fusion

Recall that LF computes a query-item similarity score by averaging the top K_R query-review similarity scores using

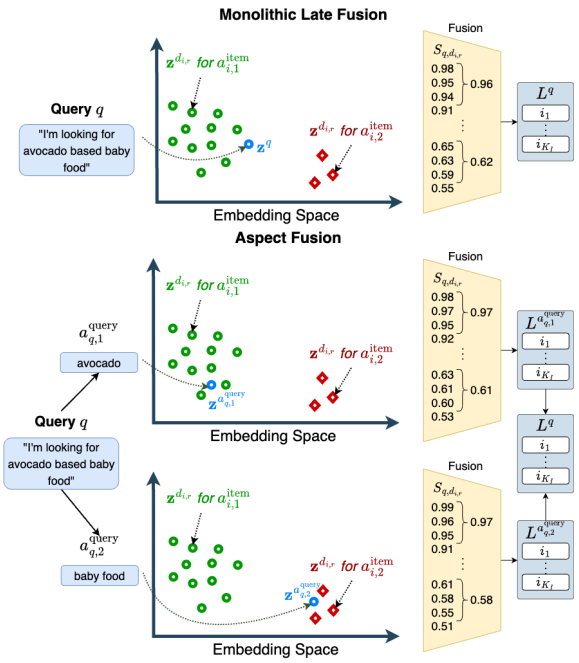

Figure 2: a) Top. In (Monolithic) LF, the full query is scored against all reviews, and the top K_R query-review scores are averaged for each item to produce a query-item score. b) Bottom. Aspect Fusion extracts aspects (i.e., query subspans) from a query, performs LF with each aspect, and aggregates the resulting top K_I item lists (i.e., one list per extracted aspect) to a final list.

Equation (1). For MA-RIR, we propose two desiderata for the aspect distribution in the top K_R reviews during fusion.

Desideratum 1: Since we assume multi-aspect queries are AND queries, if an item contains $\mathscr{A}^{\text{rel},q}_i$ relevant aspects for query q, the K_R reviews used for LF should mention all $\mathscr{A}^{\text{rel},q}_i$ of those relevant aspects.

Desideratum 2: As mentioned in Section 3.1, we also assume all query aspects are equally important, which implies that aspect frequency should be identical for all $\mathscr{A}^{\text{rel},q}_i$ aspects in the top K_R retrieved reviews.

In a fully overlapping distribution (Figure 1a) where *each* review mentions *each* aspect, both Desiderata 1 and 2 are guaranteed to be satisfied by any subset of item reviews. We thus argue that standard LF should be sufficient when reviews fully overlap in aspects, and focus on developing Aspect Fusion methods that address the failures of LF for imbalanced review-aspect distributions.

4.2. Failures of LF under Review-Aspect Imbalance

Standard LF will fail to achieve Desiderata 1 and 2 for review-aspect distributions with at least some degree of disjointedness and aspect frequency imbalance under the following assumptions.

Aspect Popularity Bias Aspects that are reviewed more frequently are more likely to be mentioned in the top K_R reviews.

Embedding Bias The non-isotropic nature of the embedding space [7] biases retrieval towards one aspect. Consider two equally sized and fully disjoint review subsets $\mathcal{D}_i^j \subset \mathcal{D}_i$ and $\mathcal{D}_i^k \subset \mathcal{D}_i$ in which reviews mention only a single aspect $a_{i,j}^{\text{rel}} \in \mathscr{A}_i^{\text{rel},q}$ or $a_{i,k}^{\text{rel}} \in \mathscr{A}_i^{\text{rel},q}$, respectively, for some item i. If query-review similarity scores tend to be higher when a review describes aspect $a_{i,j}^{\text{rel}}$ as opposed to aspect $a_{i,k}^{\text{rel}}$, LF will be more likely to select reviews from review set $\mathcal{D}_i^j$ for the top K_R fused reviews. For example, in Figure 1b), the reviews describing cooking time might be more likely to score higher with the full query than reviews describing "meatballs".

4.3. Aspect Fusion

To address these failures of LF on imbalanced data, we introduce several methods for *Aspect Fusion*, which explicitly utilizes the multi-aspect nature of reviews during fusion to address multi-aspect queries.

4.3.1. Aspect Extraction

To extract aspects from queries, we propose to use few-shot (FS) prompting with an LLM. Though the number of query-aspects is typically not known *a priori*, since we study multi-aspect queries, our proposed prompt (Figure 10 in the Appendix) asks that at least two non-overlapping subspans of the query be extracted as aspects. We represent the set of extracted query aspects for query q as $\mathscr{A}_q^{\text{ext}}$ and let $A_q^{\text{e}} = |\mathscr{A}_q^{\text{ext}}|$.

4.3.2. Aspect-Item Scoring

The key to Aspect Fusion is directly computing *aspect-review* similarity scores $S_{a,d_{i,r}}$, as opposed to similarity scores between reviews and a monolithic query, since the later can be negatively impacted by review-aspect distribution imbalance. Aspect similarity scores are computed by separately embedding each extracted aspect $a \in \mathscr{A}_q^{\text{ext}}$ as $\mathbf{z}^a = g(a)$ and calculating $S_{a,d_{i,r}} = f(\mathbf{z}^a, \mathbf{z}^{d_{i,r}})$. Then, aspect-item scores $S_{a,i} \in \mathbb{R}$ are obtained by aggregating the top K_R aspect-review scores via Eq. (1) with aspect-review scores instead of query-review scores. For each extracted aspect a, the top-K_I scoring items are ordered into a list

$$L^a = (i_1, ..., i_{K_I}) \quad \text{s.t. } i_1 \in \arg\max_i \{S_{a,i}\}$$
$$S_{a,i_k}, \geq S_{a,i_{k+1}}, \quad \forall i_k \in L^q.$$

Figure 2b) demonstrates aspect-item scoring and how it can alleviate the biases of standard LF. In this figure, the red and green points are embeddings of the reviews of item i describing aspect $a_{i,1}^{\text{item}}$ and $a_{i,2}^{\text{item}}$, respectively — both these aspects are assumed to be relevant to the query. Though the former aspect is more frequent, an equal number (K_R) of reviews for each aspect will be used during score fusion — as long as the aspect review embeddings are similar enough to the relevant query aspect embedding, and the total number of reviews for an aspect is at least K_R. In contrast, Figure 2a) shows how standard (monolithic) LF will take a biased review sample of the first aspect since it is more frequently mentioned by reviews and $\mathbf{z}^q$ happens to be closer to those review embeddings. To differentiate between LF for RIR proposed by Abdollah Pour *et al.*, and Aspect Fusion, we will refer to LF as *Monolithic LF* since it uses the full query.

4.3.3. Aspect-Item Score Fusion

After aspect-item scoring, we must aggregate the A_q^e top K_I item lists for each aspect $\{L^a\}_{a_\in \mathscr{A}_q^{\text{ext}}}$ into a single ranked list of top-K_I items for the query, L^q. We examine six aggregation strategies, which can be categorized as four score aggregation methods and two rank aggregation methods. The score-based variants convert the A_q^e aspect-item scores into a query-item score $S_{q,i}$ using

1. **AMean**: Arithmatic mean
2. **GMean**: Geometric mean
3. **HMean**: Harmonic mean
4. **Min**: Minimum

to return the final ranked list L^q. The two rank-based list aggregation methods include:

1. **Borda**: Borda count
2. **R-R**: Round-robin (interleaved) merge.

In Borda Count, the score for a given item i is calculated as follows: $\sum_{j=1}^{A_q^e}(K_I - \text{rank}_i^{L^{a_j}} + 1)$, where $\text{rank}_i^{L^{a_j}}$ is the rank of item i in list L^{a_j}. In a round-robin merge of A_q^e lists, elements from each list are merged in a cyclic order, and when a conflict arises with a particular item, that item is skipped and the merge continues from the same list.

4.4. LLM Reranking

In addition to Aspect Fusion, we also introduce an LLM reranking step for MA-RIR — to the best of our knowledge LLM reranking has not been previously studied in a reviewed-item setting. Our goal is to understand whether LLMs in cross-encoder (CE) or ZS listwise [8] settings can fuse reviews of multi-aspect items for effective reranking.

After a list L^q of top K_I items is returned from the first stage, K_R reviews for each item need to be given to the LLM for what we call *fusion-during-reranking*. For Monolithic LF, these K_R reviews are simply the K_R reviews used for LF. For Aspect Fusion, since K_R reviews were used for fusion *with each aspect,* we propose to perform a round-robin merge of the top K_R review lists for each aspect in order to preserve a balanced distribution of reviews across aspects.

For a CE, reviews are simply concatenated and cross-encoded with the query. For listwise reranking, our prompt provides the LLM with the query, initial ranked list of item IDs, reviews for each item, and instructions to order the items based on relevance to the query — the full listwise reranking prompt is in Figure 11 in the Appendix.

5. Experimental Method

We perform simulations on generated review data to study the effect of aspect balance across reviews and test our hypothesis that Aspect Fusion is more robust to aspect imbalances than Monolithic LF. While using synthetic data exposes our results to biases from the data generation process, we are able to generate synthetic review distributions with far greater control that would have been possible several years ago before the advent of LLMs. We specifically design experiments to study the performance of Aspect Fusion vs Monolithic LF under the presence of aspect imbalance, both in the form of disjointedness of aspects across reviews and imbalanced aspect frequencies.

In order to perform our experimentation, we need a dataset that has (a) multi-aspect queries and items, (b) GT aspect labels and (c) item reviews. To the best of our knowledge, there is no existing dataset with all of these properties. However, the recently-released Recipe-MPR dataset [3] includes properties (a) and (b). We leverage this dataset and generate item reviews using GPT-4.

Table 1
Distribution of ground truth (GT) and LLM-extracted aspects for Recipe-MPR queries and items

# of Aspects	1	2	3	4	5	6	7	8
Items (GT)	76	282	72	29	10	1	2	1
Queries (GT)	0	294	103	14	0	0	0	0
Queries (Extracted)	0	2	342	55	12	0	0	0

5.1. Data Generation

We create four datasets for our experiments based on the Recipe-MPR dataset and our new LLM-generated reviews.

Firstly, the *fully overlapping* dataset includes 20 reviews per item, which each mention all of the aspects of the item. Secondly, the *fully disjoint* dataset includes 10 reviews for each aspect of a given item. We also modify the fully disjoint dataset to create two datasets with imbalanced aspect frequencies. In the *one rare aspect* dataset, we remove all but one of the reviews for a randomly-selected aspect of each item. In the *one popular aspect* dataset, we keep all ten reviews for only one randomly-selected aspect of each item, and keep only one review for the other aspects.

In order to generate reviews, the GT aspects for each correct item in Recipe-MPR were used to prompt GPT-4. The total number of items for which there were GT aspects is 473. The distribution of the number of aspects per query and item is shown in Table 1. On average, each item has 2.2 aspects. The prompts we used to generate the reviews are included in the Appendix.

Recipe-MPR contains logical AND queries with ground truth (GT) labels for the query aspects. Refer to subsection 3.1 for an example of a query q and its GT aspects, $\mathcal{A}_q^{\text{query}}$. Since the focus of this paper is on MA-RIR, we only included the 411 queries whose associated correct item had at least two aspects. For each of these queries, we used two-shot examples to have GPT-4 extract "at least two non-overlapping spans" representing the relevant aspects in the query.

5.2. Experimental Details

For our query and review embeddings, we used TAS-B [9]. For the listwise reranking experiments, we used the gpt-3.5-turbo-16k model. For the CE reranking experiments, the model used was ms-marco-MiniLM-L-12-v2 [1].

6. Experimental Results

RQ1: Is Aspect Fusion helpful when item aspects are discussed disjointly across reviews?

Table 2 lists the mean absolute precision at 10 (MAP@10) and recall@10 (Re@10) of the stage 1 dense retrieval for various settings of K_R. The table is broken up according to whether the disjoint or overlapping reviews are used. Throughout this paper, we show results for $K_I = 10$. In our experiments we noticed that varying K_I led to minor changes in the results. For completeness, we report results for $K_I = 5$ in the Appendix.

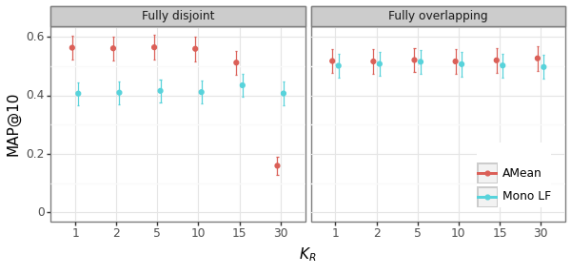

Figure 3: Monolithic LF versus Aspect Fusion with AMean aggregation. Both methods perform similarly on the fully overlapping dataset, but Aspect Fusion performs significantly better than Monolithic LF for the fully disjoint dataset and $K_R < 30$. For the fully disjoint dataset, Aspect Fusion drops in performance for $K_R > 10$ because when K_R exceeds the number of reviews per aspect, scoring is based on reviews that are irrelevant to the given aspect. This decline in performance does not apply in the fully overlapping case.

We see that for the fully overlapping dataset, Aspect Fusion is approximately equivalent to the Monolithic LF approach, while for the fully disjoint dataset, Aspect Fusion score aggregation approaches (arithmetic mean, harmonic mean, and geometric mean) offer a significant improvement in performance compared to the Monolithic LF approach. This pattern offers empirical evidence that Aspect Fusion is better suited to disjoint aspect distributions than Monolithic LF. More specifically, this suggests that Monolithic LF is not symmetrical across aspects, and fails to consider information from each of the aspects in a balanced way.

Additionally, for the fully disjoint dataset, the performance of the aspect-based approach suffers for $K_R > 10$. This can be explained by the fact that when K_R exceeds the number of disjoint reviews available for a given aspect (10 in this data), the aspect-based methods will score items based on reviews that are irrelevant to a given aspect. This could result in correct items receiving low scores for some aspects. We conclude that Aspect Fusion should use $K_R \leq R_i^{a,\min}$, where $R_i^{a,\min}$ is the smallest number of reviews for an item i for an aspect, in order to avoid this performance drop.

Furthermore, the fact that the score aggregation methods outperform the rank-based aggregation methods (R-R and Borda) offers evidence that the embedding similarity scores contain significant information about how well an item's reviews align with a given query aspect, above and beyond that item's rank relative to the other candidate items. Considering the simplicity and strong performance of AMean score aggregation, we focus on this Aspect Fusion method in the remaining results below.

RQ2: How does review aspect frequency imbalance affect Monlithic LF and Aspect Fusion?

Table 3 shows the performance of the stage 1 dense retrieval for the balanced frequency (fully disjoint) dataset and the two datasets with imbalance in the review aspect frequency. These results are also presented visually in Figure 4.

[1] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-12-v2

Table 2
LF versus Aspect Fusion with six various aggregation functions for both the Fully Disjoint and Fully Overlapping datasets with 95% error margins in parentheses.

	K_R	1		2		5		10		15		30	
Dataset		MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10
Fully disjoint	Mono LF	.41 (.04)	.67 (.05)	.41 (.04)	.69 (.04)	.42 (.04)	.69 (.04)	.41 (.04)	.70 (.04)	.43 (.04)	.70 (.04)	**.41 (.04)**	**.68 (.05)**
	AMean	.56 (.04)	**.77 (.04)**	.56 (.04)	**.77 (.04)**	.56 (.04)	**.78 (.04)**	**.56 (.04)**	.76 (.04)	.51 (.04)	**.75 (.04)**	.16 (.03)	.24 (.04)
	Borda	.38 (.04)	.56 (.05)	.39 (.04)	.59 (.05)	.38 (.04)	.58 (.05)	.37 (.04)	.57 (.05)	.35 (.04)	.53 (.05)	.13 (.03)	.21 (.04)
	GMean	**.57 (.04)**	**.77 (.04)**	.56 (.04)	**.77 (.04)**	.56 (.04)	**.78 (.04)**	**.56 (.04)**	**.77 (.04)**	.51 (.04)	**.75 (.04)**	.16 (.03)	.24 (.04)
	HMean	**.57 (.04)**	**.77 (.04)**	**.57 (.04)**	**.77 (.04)**	**.57 (.04)**	**.78 (.04)**	**.56 (.04)**	**.77 (.04)**	**.52 (.04)**	**.75 (.04)**	.16 (.03)	.24 (.04)
	Min	.43 (.04)	.62 (.05)	.01 (.01)	.03 (.02)	.01 (.01)	.03 (.02)	.01 (.01)	.03 (.02)	.01 (.01)	.03 (.02)	.01 (.01)	.03 (.02)
	R-R	.21 (.03)	.66 (.05)	.21 (.03)	.66 (.05)	.21 (.03)	.66 (.05)	.21 (.03)	.66 (.05)	.19 (.02)	.66 (.05)	.06 (.02)	.22 (.04)
Fully overlapping	Mono LF	.50 (.04)	.73 (.04)	.51 (.04)	.75 (.04)	.51 (.04)	.74 (.04)	.51 (.04)	.75 (.04)	.50 (.04)	.75 (.04)	.50 (.04)	.75 (.04)
	AMean	**.52 (.04)**	.74 (.04)	**.52 (.04)**	**.76 (.04)**	**.52 (.04)**	**.77 (.04)**	**.52 (.04)**	.75 (.04)	.52 (.04)	.75 (.04)	**.53 (.04)**	.74 (.04)
	Borda	.33 (.04)	.51 (.05)	.33 (.04)	.52 (.05)	.34 (.04)	.52 (.05)	.34 (.04)	.53 (.05)	.33 (.04)	.52 (.05)	.33 (.04)	.51 (.05)
	GMean	**.52 (.04)**	**.75 (.04)**	**.52 (.04)**	**.76 (.04)**	**.52 (.04)**	**.77 (.04)**	**.52 (.04)**	**.76 (.04)**	.52 (.04)	**.76 (.04)**	**.53 (.04)**	.75 (.04)
	HMean	**.52 (.04)**	**.75 (.04)**	**.52 (.04)**	**.76 (.04)**	**.52 (.04)**	.76 (.04)	**.52 (.04)**	**.76 (.04)**	**.53 (.04)**	**.76 (.04)**	.52 (.04)	**.76 (.04)**
	Min	.32 (.04)	.52 (.05)	.01 (.01)	.03 (.02)	.01 (.01)	.03 (.02)	.01 (.01)	.03 (.02)	.01 (.01)	.03 (.02)	.01 (.01)	.03 (.02)
	R-R	.17 (.02)	.61 (.05)	.17 (.02)	.60 (.05)	.17 (.02)	.61 (.05)	.18 (.02)	.63 (.05)	.18 (.02)	.63 (.05)	.17 (.02)	.62 (.05)

Table 3
Effect of aspect frequency imbalance on Monolithic LF and Aspect Fusion. "Balanced frequency" refers to the fully disjoint dataset where all item aspects have the same number of reviews. The values in parentheses indicate the 95% error margin.

Dataset	K_R	1		2		5		10		15		30	
		MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10
Balanced Frequency	AMean	**.56 (.04)**	**.77 (.04)**	**.56 (.04)**	**.77 (.04)**	**.56 (.04)**	**.78 (.04)**	**.56 (.04)**	**.76 (.04)**	**.51 (.04)**	**.75 (.04)**	.16 (.03)	.24 (.04)
	Mono LF	.41 (.04)	.67 (.05)	.41 (.04)	.69 (.04)	.42 (.04)	.69 (.04)	.41 (.04)	.70 (.04)	.43 (.04)	.70 (.04)	**.41 (.04)**	**.68 (.05)**
One Popular Aspect	AMean	**.52 (.04)**	**.73 (.04)**	**.43 (.04)**	**.68 (.05)**	**.33 (.04)**	**.58 (.05)**	**.27 (.04)**	**.52 (.05)**	.02 (.01)	.03 (.02)	.01 (.01)	.03 (.02)
	Mono LF	.36 (.04)	.62 (.05)	.32 (.04)	.60 (.05)	.28 (.04)	.53 (.05)	.26 (.04)	.49 (.05)	**.27 (.04)**	**.51 (.05)**	**.27 (.04)**	**.52 (.05)**
One Rare Aspect	AMean	**.52 (.04)**	**.74 (.04)**	**.45 (.04)**	**.67 (.05)**	**.36 (.04)**	**.58 (.05)**	**.34 (.04)**	**.54 (.05)**	.15 (.03)	.23 (.04)	.06 (.02)	.09 (.03)
	Mono LF	.39 (.04)	.65 (.05)	.36 (.04)	.64 (.05)	.32 (.04)	.57 (.05)	.30 (.04)	.53 (.05)	**.32 (.04)**	**.55 (.05)**	**.29 (.04)**	**.51 (.05)**

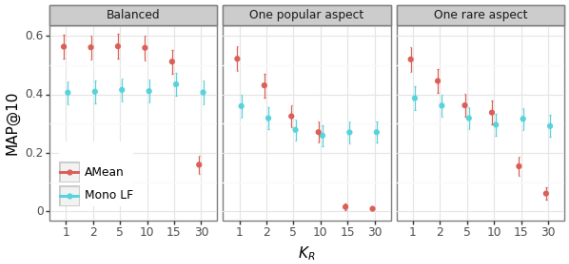

Figure 4: Effect of Aspect Frequency. Aspect Fusion performs better than Monolithic LF for low values of K_R, but suffers for higher values of K_R. This pattern is explained in the discussion of RQ1.

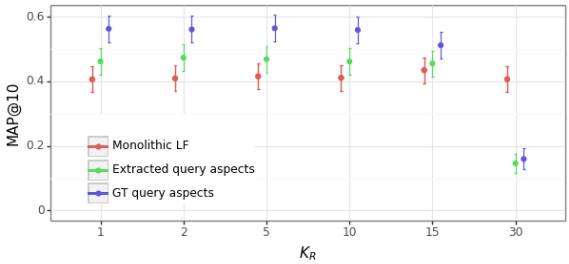

Figure 5: Aspect Fusion with GT vs extracted query aspects with fully disjoint reviews. Although GT query aspects perform better, Aspect Fusion still offers an improvement over Monolithic LF with extracted query aspects.

Note that this imbalance can only be analyzed for the case where the reviews cover disjoint, rather than overlapping, aspects.

Based on our conclusion above, we focus on the results for $K_R = 1$ in this section, since for the datasets with imbalanced review aspect frequency, $R_i^{a,\min} = 1$. We see that there is a significant decrease in performance for all methods when aspect frequency imbalance is introduced. This result suggests that balance in reviews across aspects is helpful for both Monolithic LF and Aspect Fusion.

Furthermore, for $K_R = 1$, the performance of Monolithic LF decreases more when aspect frequency imbalance is introduced, compared to for Aspect Fusion methods. For example, the MAP@10 of Monolithic LF decreased from 0.41 to 0.36 on the *one popular aspect* dataset, representing a 12% drop, compared to a 7% drop for the Aspect Fusion approach. This suggests Aspect Fusion methods may be more robust to aspect frequency imbalance.

Lastly, we note that the performance of Monolithic LF decreases as K_R grows large, which occurs because any relevant item aspects that are infrequently reviewed (there is only 1 review for rare aspects in these datasets) will contribute less and less to the query-item score with an increase in K_R.

Table 4
Effect of using LLM extracted query aspects vs. GT query aspects on Monolithic LF and Aspect Fusion. The values in parentheses indicate the 95% error margin.

	K_R	1		2		5		10		15		30	
		MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10
Extracted query aspects	AMean	**.46 (.04)**	**.70 (.04)**	**.47 (.04)**	**.72 (.04)**	**.47 (.04)**	**.72 (.04)**	**.46 (.04)**	**.71 (.04)**	**.46 (.04)**	**.71 (.04)**	.15 (.03)	.23 (.04)
	Mono LF	.41 (.04)	.67 (.05)	.41 (.04)	.69 (.04)	.42 (.04)	.69 (.04)	.41 (.04)	.70 (.04)	.43 (.04)	.70 (.04)	**.41 (.04)**	**.68 (.05)**
GT query aspects	AMean	**.56 (.04)**	**.77 (.04)**	**.56 (.04)**	**.77 (.04)**	**.56 (.04)**	**.78 (.04)**	**.56 (.04)**	**.76 (.04)**	**.51 (.04)**	**.75 (.04)**	.16 (.03)	.24 (.04)
	Mono LF	.41 (.04)	.67 (.05)	.41 (.04)	.69 (.04)	.42 (.04)	.69 (.04)	.41 (.04)	.70 (.04)	.43 (.04)	.70 (.04)	**.41 (.04)**	**.68 (.05)**

Table 5
Reranker performance of CE and LW LLMs for various K_R values. "No" refers to the case where no reranking is applied, and is equivalent to the stage 1 results. The values in parentheses indicate the 95% error margin.

		K_R	1		2		5		10		15		30	
			MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10	MAP@10	Re@10
Fully disjoint	AMean	CE	.36 (.04)	.77 (.04)	.51 (.04)	.77 (.04)	.53 (.04)	.78 (.04)	.53 (.04)	.76 (.04)	**.53 (.04)**	.75 (.04)	**.16 (.03)**	.24 (.04)
		LW	.40 (.04)	.77 (.04)	.45 (.04)	.77 (.04)	.53 (.04)	.78 (.04)	.55 (.04)	.76 (.04)	.52 (.04)	.75 (.04)	**.16 (.03)**	.24 (.04)
		No	**.56 (.04)**	.77 (.04)	**.56 (.04)**	.77 (.04)	**.56 (.04)**	.78 (.04)	**.56 (.04)**	.76 (.04)	.51 (.04)	.75 (.04)	**.16 (.03)**	.24 (.04)
	Mono LF	CE	.35 (.04)	.67 (.05)	.38 (.04)	.69 (.04)	.40 (.04)	.69 (.04)	**.44 (.04)**	.70 (.04)	**.47 (.04)**	.70 (.04)	**.47 (.04)**	.68 (.05)
		LW	.33 (.04)	.67 (.05)	.34 (.04)	.69 (.04)	.38 (.04)	.69 (.04)	.40 (.04)	.70 (.04)	.42 (.04)	.70 (.04)	.46 (.04)	.68 (.05)
		No	**.41 (.04)**	.67 (.05)	**.41 (.04)**	.69 (.04)	**.42 (.04)**	.69 (.04)	.41 (.04)	.70 (.04)	.43 (.04)	.70 (.04)	.41 (.04)	.68 (.05)
Fully overlapping	AMean	CE	.51 (.04)	.74 (.04)	**.52 (.04)**	.76 (.04)	.52 (.04)	.77 (.04)	.51 (.04)	.75 (.04)	.48 (.04)	.75 (.04)	.50 (.04)	.74 (.04)
		LW	.43 (.04)	.74 (.04)	.48 (.04)	.76 (.04)	**.55 (.04)**	.77 (.04)	**.53 (.04)**	.75 (.04)	**.53 (.04)**	.75 (.04)	.52 (.04)	.74 (.04)
		No	**.52 (.04)**	.74 (.04)	**.52 (.04)**	.76 (.04)	.52 (.04)	.77 (.04)	.52 (.04)	.75 (.04)	.52 (.04)	.75 (.04)	**.53 (.04)**	.74 (.04)
	Mono LF	CE	**.50 (.04)**	.73 (.04)	.50 (.04)	.75 (.04)	.50 (.04)	.74 (.04)	.50 (.04)	.75 (.04)	.48 (.04)	.75 (.04)	.50 (.04)	.75 (.04)
		LW	.45 (.04)	.73 (.04)	.47 (.04)	.75 (.04)	**.52 (.04)**	.74 (.04)	**.53 (.04)**	.75 (.04)	**.53 (.04)**	.75 (.04)	**.52 (.04)**	.75 (.04)
		No	**.50 (.04)**	.73 (.04)	**.51 (.04)**	.75 (.04)	.51 (.04)	.74 (.04)	.51 (.04)	.75 (.04)	.50 (.04)	.75 (.04)	.50 (.04)	.75 (.04)

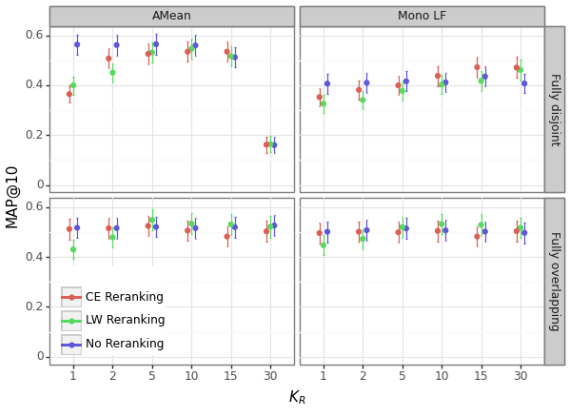

Figure 6: Comparison of reranking methods. Performance generally increases as more reviews are included in the LLM — using too few reviews can hurt performance.

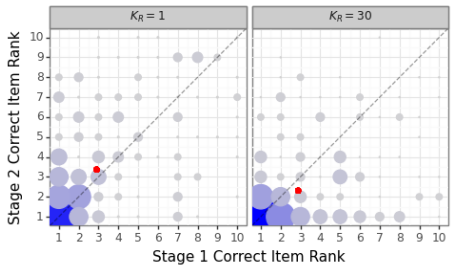

Figure 7: Ranks of correct items after Stage 1 Monolithic LF (x axis) and Stage 2 Cross-Encoder reranking (y axis) on the Fully Disjoint dataset. Circle size is proportional to position frequency, and the center of mass is shown in red. For $K_R = 1$, most of the mass lies above the diagonal line, meaning the reranker has worsened performance. On the other hand, for $K_R = 30$, most of the mass lies below the diagonal line, meaning that the reranker has improved the performance.

RQ3: How does the use of extracted query aspects instead of GT query aspects affect Aspect Fusion?

Table 4 shows the same results as Table 2 except for the case of the extracted query aspects. These results are also presented visually in Figure 5. At $K_R = 1$, while the MAP@10 of Aspect Fusion drops from 0.56 with GT aspects to 0.46 with extracted aspects, it remains higher than the 0.41 MAP@10 of Monolithic LF. This result implies that Aspect Fusion is useful even when GT query aspects are unknown.

RQ4: Are LLMs effective MA-RIR rerankers?

Table 5 summarizes the performance of the listwise[2] and cross-encoder rerankers. We see there is a beneficial effect to increasing the number of reviews K_R given to the language model for both CE and listwise reranking. Specifically, for reranking Monolithic LF on the fully disjoint dataset, listwise MAP@10 improves from 0.33 to 0.46, for $K_R = 1$ and $K_R = 30$, respectively. Similarly, CE MAP@10 improves from 0.35 MAP@10 at $K_R = 1$ to 0.47 at $K_R = 30$. We conjecture this large increase in MAP@10 with K_R is due to the quadratic nature of cross-attention across input text.

[2]Approximately 1% of queries had only 9 items returned by the listwise reranker instead of 10 — this was an error in generative retrieval.

Since Aspect Fusion did best with low K_R values, a possible reason that we did not observe any benefits of LLM reranking for Aspect Fusion is because K_R was not high enough. Also, while some reranking settings showed 2nd stage MAP@10 increases over 1st stage values (such as at $K_R = 30$ reranking of Monolithic LF for fully disjoint data), when too few reviews were given to the reranker, the second stage sometimes made performance worse, such as at $K_R = 1$.

Figure 7 shows a heatmap of the ranks assigned to the correct items by the stage 1 retriever and stage 2 reranker. An effective reranker would consistently improve the ranks for the correct item, and this would result in the center of mass lying below the anti-diagonal. We see that this is indeed the case for a high value of K_R, but is not the case for a low value of K_R. The raw values underlying this figure are provided in the Appendix.

7. Related Work

7.1. Multi-level Retrieval

The most relevant work to ours is that on RIR by Abdollah Pour *et al.* [1], which formulates the RIR problem and studies EF and LF approaches. In addition to LF with an off-the-shelf bi-encoder such as TAS-B, the authors also contrastively fine tune an encoder for LF and show performance improvements over off-the-shelf LF. Extending their contrastive learning approach to MA-RIR Aspect Fusion is a natural direction for future work. As mentioned in Section 2.2.1, Zhang and Balog [6] have previously studied the Object Fusion problem, which allows for more general two-level structures than RIR (in which a low-level document cannot describe more than one high-level object). However, they did not study neural techniques or multi-aspect retrieval, which are key to our work.

7.2. Multi-aspect Retrieval

In addition to releasing Recipe-MPR, which was used to generate review distributions in this work, Zhang *et al* [3] use the queries and items in Recipe-MPR in a multi-aspect question-answering setting, and find that FS GPT-3 listwise prompting achieves far superior accuracy to all other methods. However, it is computationally infeasible to use such listwise prompting methods for first stage retrieval. Kong *et al.* [10] consider multiple aspects when calculating relevance scores in dense retrieval, but assume documents and queries contain a fixed number of aspects from known categories. Similarly, the label aggregation method of Kang *et al.* [11] explicitly deals with multiple query aspects, but has fixed number of known categories.

Another methods called Multi-Aspect Dense Retrieval (MADRM) [10] learns early fusion embeddings of documents and queries by extracting and then aggregating their aspects, and report improvements over Monolithic LF baselines. DORIS-MAE [12] presents a dataset that deconstructs complex queries into hierarchies of aspects and sub-aspects. Unlike our aspect extraction approach, which extracts aspects from queries using few-shot prompting with an LLM, DORIS-MAE predefines these aspects and their corresponding topic hierarchy for both queries and document corpora.

Finally, some recent works study multi-aspect LLM-driven conversational recommendation [13], including work on preference elicitation over multiple aspects [14] and knowledge graph based topic-guided chatbots [15].

8. Conclusions

By extending reviewed-item-retrieval (RIR) to a setting with multi-aspect queries and items, we were able to both theoretically and empirically demonstrate the failure modes of Monolithic Late Fusion (LF) when there is an imbalance in how aspects are distributed across reviews. Specifically, since Monolithic LF is aspect-agnostic, it is subject to a frequency bias in its review selection towards more popular aspects. Furthermore, the disjointedness of aspects across reviews can induce a selection bias towards certain aspects if monolithic multi-aspect query embeddings are closer to review embeddings for those aspects.

To address these failure modes, we propose Aspect Fusion as a robust MA-RIR method for imbalanced review distributions. Using the recently released Recipe-MPR dataset, specifically designed to study multi-aspect retrieval, we design four generated datasets that allow us to empircally test the effects of review imbalances from aspect frequency and disjointess. Our experiments show that Aspect Fusion is much more robust to non-uniform review variations than Monolithic LF, outperforming the later with a 44% MAP@10 increase on some distributions.

References

[1] M. M. Abdollah Pour, P. Farinneya, A. Toroghi, A. Korikov, A. Pesaranghader, T. Sajed, M. Bharadwaj, B. Mavrin, S. Sanner, Self-supervised contrastive BERT fine-tuning for fusion-based reviewed-item retrieval, in: European Conference on Information Retrieval, Springer, 2023, pp. 3–17. doi:10.1007/978-3-031-28244-7_1.

[2] S. Kemper, J. Cui, K. Dicarlantonio, K. Lin, D. Tang, A. Korikov, S. Sanner, Retrieval-augmented conversational recommendation with prompt-based semi-structured natural language state tracking, in: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '24, Association for Computing Machinery, New York, NY, USA, 2024. doi:10.1145/3626772.3657670.

[3] H. Zhang, A. Korikov, P. Farinneya, M. M. Abdollah Pour, M. Bharadwaj, A. Pesaranghader, X. Y. Huang, Y. X. Lok, Z. Wang, N. Jones, S. Sanner, Recipe-MPR: A test collection for evaluating multi-aspect preference-based natural language retrieval, in: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 2744–2753. doi:10.1145/3539618.3591880.

[4] N. Reimers, I. Gurevych, Sentence-BERT: Sentence embeddings using Siamese BERT-networks, in: K. Inui, J. Jiang, V. Ng, X. Wan (Eds.), Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 3982–3992. doi:10.18653/v1/D19-1410.

[5] J. Johnson, M. Douze, H. Jégou, Billion-scale similarity search with GPUs, IEEE Transactions on Big Data 7 (2021) 535–547. doi:10.1109/TBDATA.2019.2921572.
[6] S. Zhang, K. Balog, Design patterns for fusion-based object retrieval, in: European Conference on Information Retrieval, Springer, 2017, pp. 684–690. doi:10.1007/978-3-319-56608-5_66.
[7] K. Ethayarajh, How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings, in: K. Inui, J. Jiang, V. Ng, X. Wan (Eds.), Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 55–65. URL: https://aclanthology.org/D19-1006. doi:10.18653/v1/D19-1006.
[8] X. Ma, X. Zhang, R. Pradeep, J. Lin, Zero-shot listwise document reranking with a large language model, 2023. arXiv:2305.02156.
[9] S. Hofstätter, S.-C. Lin, J.-H. Yang, J. Lin, A. Hanbury, Efficiently teaching an effective dense retriever with balanced topic aware sampling, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 113–122.
[10] W. Kong, S. Khadanga, C. Li, S. K. Gupta, M. Zhang, W. Xu, M. Bendersky, Multi-aspect dense retrieval, in: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 3178–3186. doi:10.1145/3534678.3539137.
[11] C. Kang, X. Wang, Y. Chang, B. Tseng, Learning to rank with multi-aspect relevance for vertical search, in: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, WSDM '12, Association for Computing Machinery, New York, NY, USA, 2012, p. 453–462. doi:10.1145/2124295.2124350.
[12] J. Wang, K. Wang, X. Wang, P. Naidu, L. Bergen, R. Paturi, DORIS-MAE: Scientific document retrieval using multi-level aspect-based queries, in: Proceedings of the 37th International Conference on Neural Information Processing Systems, NIPS '23, Curran Associates Inc., Red Hook, NY, USA, 2024. doi:10.5555/3666122.3667790.
[13] Y. Deldjoo, Z. He, J. McAuley, A. Korikov, S. Sanner, A. Ramisa, R. Vidal, M. Sathiamoorthy, A. Kasirzadeh, S. Milano, A review of modern recommender systems using generative models (gen-recsys), in: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain, 2024.
[14] D. E. Austin, A. Korikov, A. Toroghi, S. Sanner, Bayesian optimization with LLM-based acquisition functions for natural language preference elicitation, in: Proceedings of the 18th ACM Conference on Recommender Systems (RecSys'24), 2024.
[15] K. Zhou, Y. Zhou, W. X. Zhao, X. Wang, J.-R. Wen, Towards topic-guided conversational recommender system, arXiv preprint arXiv:2010.04125 (2020).

A. Appendix A

A.1. LLM Prompts

We provide the prompts userd for overlapping review generation, disjoint review generation, query aspect extraction, and listwise reranking in Figures 8, 9, 10, and 11 respectively.

```
Given the following description of a recipe,
generate {{num_reviews}} reviews for the recipe.
Make sure you mention all of the aspects in every
review. Do not include review numbers in the output.

Description: {{description}}
Aspects: {{aspects}}
Result (in a list in JSON format, with "reviews" as
the key):
```

Figure 8: Overlapping Review Generation Prompt Used with GPT-4

```
Given the following description of a recipe and a
specific aspect, generate {{num_reviews}} reviews
for the specific aspect provided without mentioning
anything that relates to the other aspects. Do not
mention anything in the "Do Not Mention" list.

Description: {{description}}
Aspect: {{aspect}}
Do Not Mention: {{aspects}}
Result (in JSON format, with "reviews" as the key):
```

Figure 9: Disjoint Review Generation Prompt Used with GPT-4

```
Given a multi-aspect query, extract at least two
non-overlapping spans to represent the query
aspects.

<few shot examples>

Query: {{query}}
Result:
```

Figure 10: Query Aspect Extraction Prompt Used with GPT-4

```
Given a query, an ordered list of items, and the top
reviews for each item, return a better ordered list
of items.

Query: {{query}}

Initial Ordered Item List: {{ranked_list}}

The top reviews for each item, in order of predicted
relevance, are: {{top_review_dict_list}}

You can now take a second look at the order of the
items, and if necessary, return a better order given
the query, keeping list length the same, and using
each item ID exactly once in the returned list. If
an item is more likely to be relevant to the query,
it should appear sooner in the list. Use the query
and the reviews from each item for your reasoning.

Respond exactly in the following format.

The JSON format of an ordered list for the response
is:
[{"ID": <1st most relevant item ID>}, {"ID": <2nd
most relevant item ID>},...]
Response:
```

Figure 11: Generic Listwise Reranking Prompt Used with GPT-3.5

Table 6
Stage 1 retriever performance for various aggregation functions and settings of K_R, with $K_I = 5$. All methods except Mono LF include Aspect Fusion. The values in parentheses indicate the 95% error margin.

	K_R	1		2		5		10		15		30	
Dataset		MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5
Fully disjoint	AMean	.55 (.04)	.71 (.04)	.55 (.04)	.70 (.04)	.55 (.04)	.70 (.04)	.55 (.04)	.69 (.04)	.50 (.04)	.67 (.05)	.16 (.03)	.21 (.04)
	Borda	.37 (.04)	.49 (.05)	.38 (.04)	.49 (.05)	.37 (.04)	.48 (.05)	.36 (.04)	.45 (.05)	.34 (.04)	.44 (.05)	.13 (.03)	.18 (.04)
	GMean	.56 (.04)	.71 (.04)	.55 (.04)	.71 (.04)	.55 (.04)	.71 (.04)	.55 (.04)	.69 (.04)	.50 (.04)	.67 (.05)	.16 (.03)	.21 (.04)
	HMean	.56 (.04)	.71 (.04)	.56 (.04)	.71 (.04)	.56 (.04)	.71 (.04)	.55 (.04)	.69 (.04)	.51 (.04)	.68 (.05)	.16 (.03)	.21 (.04)
	Min	.42 (.04)	.54 (.05)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)
	Mono LF	.39 (.04)	.56 (.05)	.39 (.04)	.57 (.05)	.40 (.04)	.57 (.05)	.40 (.04)	.58 (.05)	.42 (.04)	.61 (.05)	.39 (.04)	.58 (.05)
	R-R	.25 (.03)	.53 (.05)	.25 (.03)	.53 (.05)	.25 (.03)	.54 (.05)	.26 (.03)	.54 (.05)	.22 (.03)	.47 (.05)	.07 (.02)	.15 (.03)
Fully overlapping	AMean	.51 (.04)	.67 (.05)	.51 (.04)	.68 (.05)	.51 (.04)	.67 (.05)	.50 (.04)	.66 (.05)	.51 (.04)	.66 (.05)	.52 (.04)	.66 (.05)
	Borda	.32 (.04)	.45 (.05)	.32 (.04)	.45 (.05)	.33 (.04)	.44 (.05)	.33 (.04)	.44 (.05)	.32 (.04)	.43 (.05)	.32 (.04)	.44 (.05)
	GMean	.51 (.04)	.67 (.05)	.51 (.04)	.68 (.05)	.51 (.04)	.66 (.05)	.50 (.04)	.65 (.05)	.51 (.04)	.66 (.05)	.51 (.04)	.66 (.05)
	HMean	.51 (.04)	.66 (.05)	.51 (.04)	.68 (.05)	.51 (.04)	.66 (.05)	.51 (.04)	.66 (.05)	.52 (.04)	.66 (.05)	.51 (.04)	.66 (.05)
	Min	.31 (.04)	.44 (.05)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)	.01 (.01)
	Mono LF	.49 (.04)	.66 (.05)	.50 (.04)	.66 (.05)	.50 (.04)	.66 (.05)	.50 (.04)	.67 (.05)	.49 (.04)	.65 (.05)	.48 (.04)	.65 (.05)
	R-R	.20 (.03)	.50 (.05)	.22 (.03)	.50 (.05)	.22 (.03)	.50 (.05)	.21 (.03)	.50 (.05)	.21 (.03)	.48 (.05)	.22 (.03)	.49 (.05)

Table 7
Stage 1 retriever performance by review aspect frequency and settings of K_R, with $K_i = 5$. "Balanced" refers to the fully disjoint dataset where all item aspects have the same number of reviews. The values in parentheses indicate the 95% error margin.

	K_R	1		2		5		10		15		30	
Dataset		MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5
Balanced Frequency	AMean	.55 (.04)	.71 (.04)	.55 (.04)	.70 (.04)	.55 (.04)	.70 (.04)	.55 (.04)	.69 (.04)	.50 (.04)	.67 (.05)	.16 (.03)	.21 (.04)
	Mono LF	.39 (.04)	.56 (.05)	.39 (.04)	.57 (.05)	.40 (.04)	.57 (.05)	.40 (.04)	.58 (.05)	.42 (.04)	.61 (.05)	.39 (.04)	.58 (.05)
One Popular Aspect	AMean	.51 (.04)	.65 (.05)	.42 (.04)	.58 (.05)	.31 (.04)	.45 (.05)	.25 (.04)	.39 (.05)	.01 (.01)	.02 (.01)	.01 (.01)	.02 (.01)
	Mono LF	.35 (.04)	.50 (.05)	.30 (.04)	.46 (.05)	.26 (.04)	.40 (.05)	.24 (.04)	.38 (.05)	.25 (.04)	.38 (.05)	.25 (.04)	.38 (.05)
One Rare Aspect	AMean	.51 (.04)	.65 (.05)	.43 (.04)	.59 (.05)	.35 (.04)	.48 (.05)	.33 (.04)	.45 (.05)	.15 (.03)	.20 (.04)	.06 (.02)	.08 (.03)
	Mono LF	.37 (.04)	.53 (.05)	.34 (.04)	.51 (.05)	.30 (.04)	.45 (.05)	.28 (.04)	.43 (.05)	.30 (.04)	.45 (.05)	.28 (.04)	.40 (.05)

A.2. Results for $K_I = 5$

In the main body we showed various results of experiments where K_I was set to 10. We found that varying K_I within this order of magnitude had a very small effect on the results, and therefore did not include findings for any other settings of K_I above. For completeness, in this section we duplicate the preceding tables but use $K_I = 5$ instead of $K_I = 10$. See Tables 6, 7, 8, and 9 for these results.

A.3. Data for Figure 7

In Figure 7, we show the number of queries for which the correct item was ranked in a certain position by the stage 1 retriever and stage 2 reranker. The underlying data for this figure is shown in Table 10.

Table 8
Stage 1 retriever performance by whether labelled GT or extracted query aspects are used, with $K_I = 5$. The values in parentheses indicate the 95% error margin.

	K_R	1		2		5		10		15		30	
		MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5
Extracted query aspects	AMean	.45 (.04)	.60 (.05)	.46 (.04)	.63 (.05)	.46 (.04)	.62 (.05)	.45 (.04)	.61 (.05)	.44 (.04)	.62 (.05)	.14 (.03)	.20 (.04)
	Mono LF	.39 (.04)	.56 (.05)	.39 (.04)	.57 (.05)	.40 (.04)	.57 (.05)	.40 (.04)	.58 (.05)	.42 (.04)	.61 (.05)	.39 (.04)	.58 (.05)
GT query aspects	AMean	.55 (.04)	.71 (.04)	.55 (.04)	.70 (.04)	.55 (.04)	.70 (.04)	.55 (.04)	.69 (.04)	.50 (.04)	.67 (.05)	.16 (.03)	.21 (.04)
	Mono LF	.39 (.04)	.56 (.05)	.39 (.04)	.57 (.05)	.40 (.04)	.57 (.05)	.40 (.04)	.58 (.05)	.42 (.04)	.61 (.05)	.39 (.04)	.58 (.05)

Table 9
Stage 2 reranker performance by reranking method and setting of K_R, with $K_I = 5$. "No" refers to the case where no reranking is applied, and is equivalent to the stage 1 results. The values in parentheses indicate the 95% error margin.

		K_R	1		2		5		10		15		30	
			MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5	MAP@5	Re@5
Fully disjoint	AMean	CE	.41 (.04)	.71 (.04)	.52 (.04)	.70 (.04)	.53 (.04)	.70 (.04)	.53 (.04)	.69 (.04)	.53 (.04)	.67 (.05)	.17 (.03)	.21 (.04)
		LW	.44 (.04)	.71 (.04)	.48 (.04)	.70 (.04)	.53 (.04)	.70 (.04)	.55 (.04)	.69 (.04)	.53 (.04)	.67 (.05)	.16 (.03)	.21 (.04)
		No	.55 (.04)	.71 (.04)	.55 (.04)	.70 (.04)	.55 (.04)	.70 (.04)	.55 (.04)	.69 (.04)	.50 (.04)	.67 (.05)	.16 (.03)	.21 (.04)
	Mono LF	CE	.35 (.04)	.56 (.05)	.37 (.04)	.57 (.05)	.38 (.04)	.57 (.05)	.41 (.04)	.58 (.05)	.46 (.04)	.61 (.05)	.46 (.04)	.58 (.05)
		LW	.33 (.04)	.56 (.05)	.34 (.04)	.57 (.05)	.37 (.04)	.57 (.05)	.37 (.04)	.58 (.05)	.44 (.04)	.61 (.05)	.43 (.04)	.58 (.05)
		No	.39 (.04)	.56 (.05)	.39 (.04)	.57 (.05)	.40 (.04)	.57 (.05)	.40 (.04)	.58 (.05)	.42 (.04)	.61 (.05)	.39 (.04)	.58 (.05)
Fully overlapping	AMean	CE	.51 (.04)	.67 (.05)	.51 (.04)	.68 (.05)	.51 (.04)	.67 (.05)	.50 (.04)	.66 (.05)	.49 (.04)	.66 (.05)	.51 (.04)	.66 (.05)
		LW	.48 (.04)	.67 (.05)	.50 (.04)	.68 (.05)	.52 (.04)	.67 (.05)	.51 (.04)	.66 (.05)	.53 (.04)	.66 (.05)	.53 (.04)	.66 (.05)
		No	.51 (.04)	.67 (.05)	.51 (.04)	.68 (.05)	.51 (.04)	.67 (.05)	.50 (.04)	.66 (.05)	.51 (.04)	.66 (.05)	.52 (.04)	.66 (.05)
	Mono LF	CE	.50 (.04)	.66 (.05)	.50 (.04)	.66 (.05)	.50 (.04)	.66 (.05)	.50 (.04)	.67 (.05)	.48 (.04)	.65 (.05)	.49 (.04)	.65 (.05)
		LW	.47 (.04)	.66 (.05)	.48 (.04)	.66 (.05)	.52 (.04)	.66 (.05)	.53 (.04)	.67 (.05)	.51 (.04)	.65 (.05)	.51 (.04)	.65 (.05)
		No	.49 (.04)	.66 (.05)	.50 (.04)	.66 (.05)	.50 (.04)	.66 (.05)	.50 (.04)	.67 (.05)	.49 (.04)	.65 (.05)	.48 (.04)	.65 (.05)

Table 10
Ranks assigned to the correct items for stage 1 retriever and stage 2 CE reranker with AMean aggregation Aspect Fusion.

		Stage 2 Correct Item Rank									
	Stage 1 Correct Item Rank	1	2	3	4	5	6	7	8	9	10
$K_R = 1$	1	68	19	12	9	2	2	4	2	0	0
	2	12	20	8	1	3	4	1	3	0	0
	3	5	3	8	5	2	2	2	1	1	1
	4	1	2	2	4	1	4	2	0	1	0
	5	1	1	1	2	3	0	2	2	1	1
	6	0	0	0	1	0	0	1	1	1	1
	7	3	3	2	2	1	3	1	1	3	0
	8	1	0	2	0	1	0	1	0	4	0
	9	0	0	0	0	1	0	1	0	2	0
	10	0	1	0	1	1	1	2	1	0	1
$K_R = 30$	1	83	21	5	5	0	2	1	0	0	0
	2	29	12	2	1	2	2	3	1	1	0
	3	12	5	3	3	2	0	1	2	0	0
	4	7	2	1	1	1	3	1	0	0	0
	5	7	2	7	5	0	0	1	1	0	0
	6	5	1	3	1	0	2	2	0	0	0
	7	3	1	1	1	0	1	0	0	0	1
	8	4	0	0	1	0	2	1	0	0	0
	9	0	2	1	0	1	1	0	1	0	0
	10	1	2	1	0	0	0	1	1	0	0