PROBES : Performance and Relevance Observation for BEtter Search

Sejal Jain
Amazon
sejaljn@amazon.com

Cyrus DSouza
Amazon
dsocyrus@amazon.com

Jitenkumar Rana
Amazon
jitenkra@amazon.com

Aniket Joshi
Amazon
anikjosh@amazon.com

Promod Yenigalla
Amazon
promy@amazon.com

Abstract

High-quality search is essential for the success of online platforms, spanning e-commerce, social media, shopping-focused applications, and broader search systems such as content discovery and enterprise web search. To ensure optimal user experience and drive business growth, continuous evaluation and improvement of search systems is crucial. This paper introduces PROBES, a novel multi-task system powered by Large Language Models (LLMs) designed for end-to-end evaluation of semantic search systems. PROBES identifies context-aware relevance using a fine-grained scale (exact, substitute, complement, irrelevant) by leveraging the query category, feature-level intent, and category-aware feature importance, enabling more precise and consistent judgments than relying solely on raw query text. This allows PROBES to provide differentiated relevance assessment across a diverse range of query categories. PROBES then dives deeper to understand the reason behind irrelevant results (Precision issues) by checking product content conflicts and inaccuracies. It also analyzes Missed Recall by leveraging retrieval and relevance models to determine whether a missed recall was due to a selection issue or a ranking/retrieval system issue. To evaluate PROBES, we introduce a new metric, the Actionable Error Rate (AER), defined as the proportion of actionable errors over all flagged errors. We observe that PROBES operates at an AER of 76%, generating actionable insights across 100 product categories.

1 Introduction

Search is the primary entry point for user interaction on online platforms, helping users explore products and express their intent. The quality of search algorithms is crucial for ensuring a smooth user experience, enabling users to find desired products with minimal interaction. An effective search algorithm must address three fundamental challenges: (1) understanding user intent, (2) retrieving the most relevant products that fulfill the user's session intent, and (3) ranking and displaying results with the most relevant items appearing at the top.

A search system represents a complex AI application requiring a cascade of models to interpret user intent, retrieve relevant results, and rank them from most to least relevant. Large-scale evaluation of search systems is essential for providing timely feedback to search algorithms, ensuring high-quality user experiences, and driving overall system effectiveness. However, search system evaluation presents several complexities: the intricate, multi-model cascade architecture of search systems; the variety of ways users express search intent through queries; and the requirement to retrieve items from extensive collections within milliseconds.

Online A/B testing is a standard for evaluating search model performance but has limitations: it requires significant user feedback, complicates control group management when multiple tests run concurrently, and risks degrading user experience if a model under performs. For example, in e-commerce, search directly affects product discovery and purchasing decisions; in delivery platforms, it influences item availability and fulfillment choices; and in social media, it shapes content discovery and user engagement. These challenges highlight the need for scalable offline evaluation. Manual audits, though reliable, are slow and resource-intensive, evaluating 50,000 queries with 20 results each would demand over 6,000 hours. While crowd sourcing offers scale, it's costly and demands multiple annotators to ensure accuracy.

In this paper, we introduce PROBES, an LLM-based system for offline search quality evaluation. PROBES detects Precision (irrelevant products that were displayed in top results) and Missed Recall (relevant products that were not retrieved) issues for a search query and its results. It further inves-

tigates these issues to determine the root causes (poor index feature quality, retrieval/ranking issue or selection issue). To achieve this, PROBES leverages LLMs to: (1) understand user intent and query category, (2) determine query category aware product relevance to query intent, and (3) evaluate product index feature quality. Additionally, it employs a keyword-product similarity model trained on a novel relevance aware loss to retrieve the top-k most relevant products to determine root cause of missed recall issues. We further propose a new metric, "Actionable Error Rate" (AER), to measure the quality of PROBES by assessing the human-PROBES agreement rate for defect identification.

The key practical and scientific contributions of this paper are as follows: (1) We propose PROBES, an LLM-powered framework that stitches together multiple components to enable end-to-end evaluation of complex semantic search systems. (2) We demonstrate that incorporating feature-level intent and category-aware feature importance leads to more accurate relevance assessment than relying solely on raw query text (Section 5.2). Table 2 highlights the performance gains resulting from this design. (3) We introduce the Actionable Error Rate (AER) as a new metric to measure the effectiveness of PROBES, and show that it achieves an AER of 76% across 100 product categories, producing actionable insights for improving search algorithms. (4) We present a relevance-aware loss function to train the retrieval model, and show in Section 5.4 that it provides clear improvements over relevance-agnostic training objectives.

This paper is organized as follows. Section 3 describes the dataset used throughout the paper. Section 4 introduces the architecture of PROBES. Section 5 dives deep into each component of PROBES including the introduction of our new metric "Actionable Error Rate" (AER) to evaluate the performance of such a complex evaluation system. We summarize our contributions and future directions in Section 6, and discuss limitations in Section 7.

2 Related Work

Traditional search evaluation relied heavily on human judgments, as outlined by (Voorhees, 2001). To improve scalability, crowd-sourcing methods were introduced (Alonso and Baeza-Yates, 2011; Blanco et al., 2011), though they struggled with consistency and cost. The use of behavioral signals marked a major shift—click data and engagement metrics were shown to enhance relevance assessment in operational systems (Huang et al., 2013; Liu et al., 2017; Wang et al., 2018). Deep learning brought further improvements, enabling better semantic understanding of queries and products (Zhang et al., 2016; Li et al., 2019; Yang et al., 2019), with contextual embeddings advancing performance even further (MacAvaney et al., 2019; Dai and Callan, 2020). Most recently, LLMs have shown promise in automating relevance evaluation (Mehrdad et al., 2024), multimodal assessment (Hosseini et al., 2024), missed recall detection (Wu et al., 2024), post-ranking evaluation (Yan et al., 2024), and graded relevance scoring (Liu et al., 2024). PROBES builds on these advances by offering a scalable, end-to-end semantic search evaluation framework

3 Master Dataset

All PROBES experiments use the Shopping Queries Dataset (Reddy et al., 2022), which reflects real-world behavior through user queries, product listings, and ESCI-based relevance labels. We focus on English (US) queries and analyze product content - titles, descriptions and bullets. Relevance labels include:

- **Exact (E):** Fully matches the query and its specifications (e.g., "plastic water bottle 24oz").
- **Substitute (S):** Functionally similar but misses some aspects (e.g., red shirt for "green shirt").
- **Complement (C):** Used with a matching item (e.g., track pants for "running shoes").
- **Irrelevant (I):** Unrelated or unsuitable (e.g., socks for "telescope")

4 PROBES

Figure 1 demonstrates the PROBES architecture. (1) PROBES begins by identifying the query category and extracting feature-level intent with category-aware feature importance from the search query using **QIC-LLM**. (2) If the query category corresponds to a product-search intent, PROBES uses **QRR-LLM** to measure context-aware relevance between the query intent and the retrieved results using a fine-grained scale: *exact, substitute, complement, irrelevant*. (3) When a **Precision issue** (i.e., at least one irrelevant result)

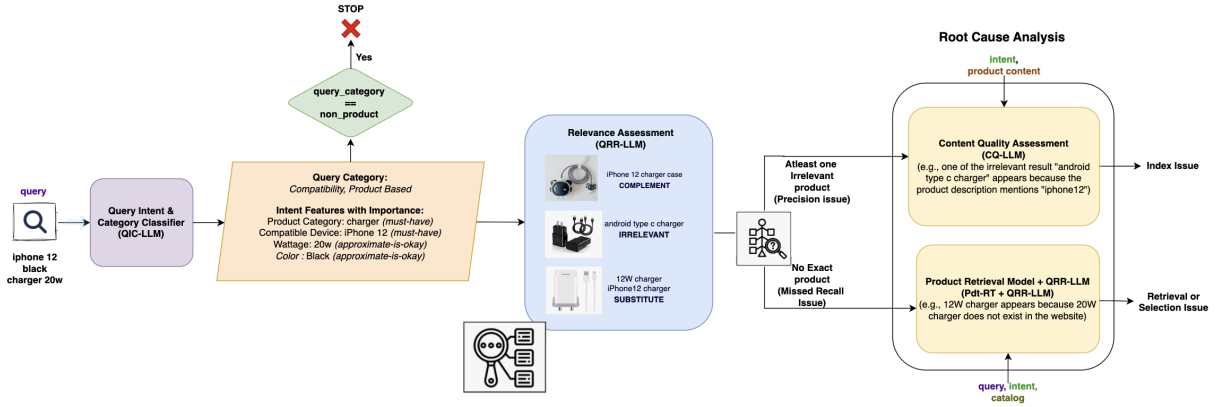

Figure 1: PROBES workflow

occurs, PROBES invokes **CQ-LLM** to find the **root cause**, primarily inspecting index features for feature-value conflicts or inaccuracies — the most common source of irrelevance. (4) If the search results contain a **Missed Recall issue** (i.e., no Exact products found in Step 2), PROBES uses the **Pdt-RT model** to retrieve the top-k most relevant products from the collection. It then applies **QRR-LLM** to evaluate the relevance of these retrieved products to the query intent. If at least one Exact product is present among the top-k retrieved items, PROBES features the Missed Recall issue to ranking/retrieval errors; otherwise, it concludes the issue arises from missing selection in the collection.

5 PROBES Components

5.1 Query Intent & Category Classifier (QIC-LLM)

Query Intent and Category Classifier (QIC-LLM) is a multi-task LLM component designed to uncover hierarchical user intent from search queries. Queries range from exact product searches (e.g., "brand WH-1000XM5 headphones") to subjective needs (e.g., "best laptop") or even non-product intents (e.g., "return policy"). Understanding this intent is essential, particularly for complex or ambiguous queries that often lead to irrelevant retrieval and incorrect relevance labeling. Symptomatic queries such as "stains on carpets," for instance, express a problem to be solved, yet systems frequently return items related to carpets or stains rather than the intended solution (e.g., stain removers).

To address such challenges, QIC-LLM organizes user intent at both semantic and feature levels. Empirical analysis of 10000 queries across 100 product categories shows that users consistently follow one of 15 recurring patterns, including (i) *Negation queries* (e.g., "chairs without wheels"), (ii) *Specification queries* (e.g., "iPhone 15 Pro Max 256GB"), and (iii) *Compatibility queries* (e.g., "charger for iPhone 16"). Additional examples appear in Table 6. These observations motivate the need for structured, category-aware interpretation of user queries. QIC-LLM performs two key functions: (1) **Query category classification**: Identifies the semantic type of the query—e.g., (2) **feature-level intent extraction with importance**: Extracts features such as *product_category*, *compatible devices*, and *color*, assigning each a corresponding importance ("must_have" vs. "approximate_is_okay") inferred from the query category.

QIC LLM benefits PROBES in two ways: (i) It improves query–result relevance measurement, yielding an average gain of ∼4.2% over raw-query–based relevance (Table 2). (ii) It enables effective index-feature conflict checks in CQ-LLM, for diagnosing Precision issues.

QIC-LLM Output Example

Query: black charger for iPhone 12
Output:

```
{
  "query_category": "compatibility",
  "features_with_importance": {
    "product category": {"value": "charger",
                         "importance": "must_have"},
    "compatible_device": {"value": "iPhone",
                         "importance": "must_have"},
    "color": {"value": "black",
           "importance": "approximate_is_okay"}
  }
}
```

We evaluated several LLMs on two core tasks—query category classification and feature-

	Query category		Feature-level intent	
Model	Precision	Recall	Precision	Recall
Claude-4-Sonnet	0.92	0.91	0.95	0.93
DS-R1-Qwen-14B	0.92	0.89	0.91	0.89
Mistral Nemo	0.85	0.83	0.88	0.86
Mixtral-8x7B	0.87	0.85	0.90	0.88

Abbreviations: DS-R1-Qwen-14B = DeepSeek-R1-Distill-Qwen-14B

Table 1: QIC-LLM Evaluation

level intent extraction (Table 1). For each model, we performed dedicated prompt engineering (8–10 variants per LLM) and selected the best prompt using a validation set (as per F1 metric) of 1,000 manually annotated queries sampled from the master dataset (Section 3). The final prompts were then evaluated on a 10,000-query test set (100 queries for each of 100 product categories) using a human-in-the-loop setup. Claude 4 Sonnet achieved the highest precision and recall, with DeepSeek-R1-Distill-Qwen-14B performing comparably and showing strong results on both tasks. Mixtral remained competitive given its smaller architecture, while Mistral Nemo performed reliably on structured, text-driven inputs. This consistency reflects that structured, text-only intent understanding aligns well with the strengths of modern LLMs, making lighter open-weight models viable for production use. Although Claude 4 achieved the best accuracy, we selected DeepSeek-R1-Distill-Qwen-14B for production due to its 2.2× cost efficiency enabled by inference optimizations, with only marginal performance loss. We also evaluated its category-wise precision and recall (Table 5).

5.2 Query-Result Relevance (QRR-LLM)

In this section, we provide the details of query-result relevance task. Classifying each product shown in response to a user query as being relevant or not may not always the most appropriate. For example, for the query "iPhone": would an iPhone charger be relevant, irrelevant, or somewhere in between? In practice, many users issue such broad queries expecting the search engine to infer their true intent, such as purchasing accessories rather than the phone itself. To address this issue in relevance evaluation we have adopted the ESCI labeling scheme, categorizing query-result pairs into four classes: *Exact (E)*, *Substitute (S)*, *Complement (C)*, and *Irrelevant (I)*. This offers a more granular alternative to binary relevance.

Further, we propose to use query category and feature level intent with importance values instead of raw query as input to LLM for relevance measurement task. Motivation to do this is the following: This allows PROBES to focus only on queries that express clear product-based intent. Query category classification filters out non-product or ambiguous queries (e.g., "return policy", "120cm"), which would otherwise introduce arbitrary ESCI labels and degrade evaluation quality.

Second, for meaningful product based queries, QIC-LLM assigns a query category (e.g., compatibility, feature-based, subjective), which provides essential context to interpret relevance accurately. This becomes especially critical when determining whether a product is a valid Substitute or truly Irrelevant. While ESCI offers clearer definitions for labels like Exact and Complement, the line between Substitute and Irrelevant often depends on the user's core intent, something not always obvious from surface-level matching. For example, showing a blue bag instead of a black bag may still be acceptable in a feature-based query, as color is often a flexible preference, and the blue bag can be considered a reasonable Substitute. However, in a compatibility-based query like "20W charger for iPhone", relevance hinges on the product's compatibility. If the result is a charger for Android (C-type), it fails to satisfy the primary intent and must be labeled Irrelevant, regardless of matching color or product category. To identify what matters most in each query and determine whether a returned product truly meets the user's intent, we rely on QIC-LLM. By classifying the query and extracting structured features along with their importance (e.g., "must-have" vs. "approximate"), QIC-LLM helps isolate the critical features that define relevance, enabling more precise and intent-aligned ESCI labeling.

Third, for some queries that are highly specific, relevance assessment works differently. For example, in a query such as "iPhone 15 Pro Max 256GB Black," the product collection might contain only one exact match. Other near matches (e.g., 128GB versions) may be wrongly labeled Irrelevant when they should be considered valid Substitutes. Recognizing the specificity of such queries helps avoid penalizing the search engine unfairly.

We evaluate multiple LLMs for the relevance measurement task. Our initial step is to invest in prompt-engineering (8–10 iterations per LLM) to establish strong baselines for open-source models. We then run ablation studies using three input configurations: (i) raw query, (ii) feature-level

	Raw query			Attr-level intent (w/o QC importance)			Attr-level intent w/ QC importance		
Model	P	R	F1	P	R	F1	P	R	F1
Claude-4-Sonnet	0.93	0.91	0.92	0.95	0.92	0.94	0.97	0.94	0.96
DS-R1-Qwen-14B	0.92	0.90	0.91	0.94	0.91	0.93	0.96	0.93	0.95
Mixtral-8x7B	0.91	0.89	0.90	0.92	0.89	0.91	0.94	0.91	0.93
Mistral Nemo	0.89	0.87	0.88	0.91	0.87	0.90	0.93	0.89	0.92

Abbreviations: QC = Query Category. P = Precision, F1 = F1 Score, R = Recall, DS-R1 = DeepSeek-R1-Distill-Qwen-32B.

Table 2: QRR-LLM Evaluation across different input types and models with Precision (P), Recall (R), and F1 scores

intent, and (iii) feature-level intent with query-category–based importance. A validation set of 1,000 (query, intent, product content, ESCI label) tuples from the master dataset is used to select the best prompt per model. Final evaluation is performed on a 30,000-sample test set (300 per product category). Table 2 reports detailed results across LLMs and input variants.

We find that using feature-level intent with category-aware importance consistently outperforms other inputs, as it provides QRR-LLM with structured guidance on which features matter most. While Claude-4-Sonnet delivers the best overall performance, DeepSeek-R1-Distill-Qwen-14B outperforms other open-source models of comparable size—even some larger ones—likely due to stronger reasoning capabilities, which are critical for relevance assessment.

5.3 Content-Quality (CQ-LLM)

Following an "Irrelevant" classification from QRR-LLM, PROBES invokes a Content Quality assessment module to diagnose potential root causes within the product listing data. An irrelevant result appears in search result if product content has inaccurate or conflicting feature values. For each identified feature (e.g., color, device compatibility, by QIC-LLM from a query like "red case for iPhone 13"), CQ-LLM performs a multi-faceted analysis of the product content page. We leverage the framework and model architecture from (Joshi et al., 2025), to identify discrepancies in the product information. This analysis encompasses (i) Verifying the factual accuracy of feature values against the product information presented across various modalities, including product title and detailed description. (ii) Checking for contradictions or conflicting information regarding the feature across different sections of the product content. Discrepancies identified (Refer to 7 for examples) during this inaccuracy and conflicts evaluation are flagged as potential contributing factors to the item's irrelevant retrieval. Given a query, irrelevant results from master dataset (section 3) and feature level intent from QIC-LLM, we leverage Claude-4 to detect inaccuracies or conflicts for each feature identified by QIC-LLM.

Validation set contains 1,000 examples and Test set contains 10,000 samples (100 per product category). We chose the best prompt for each model based on Precision on validation set.

Finally, we evaluate model performance on test set by computing Actionable Error Rate (AER eq 1) - the proportion of system-flagged errors that are validated by human reviewers as both correctly identified and operationally fixable. Note that, AER is a precision oriented metric. We don't focus on recall for this task since output of this task is used for driving fix treatments (generally manual). In practice, fix capacity is lesser compared to the volume of issues identified for fixing . Hence, AER (a precision oriented metric) is more useful for efficient downstream consumption.

$$\text{AER} = \frac{\text{Human Validated, Fixable Errors}}{\text{Total Errors Flagged by the System}} \quad (1)$$

We observe that smaller LLMs (<20B params) don't perform well since this is a complex reasoning task. Again, we consistently observe that Claude-4-Sonnet performs better than rest of the open-source models. Among open-source models, DeepSeek-R1-Distill-Qwen-32B performs best. Again, we conjecture that these may be due to better reasoning capabilities of the models and task also requires reasoning.

5.4 Product Retrieval Model (Pdt-RT)

We analyze missed recall issues by focusing on queries that yield no exact matches. The Product Retrieval Model plays a crucial role in diagnosing ranking and selection issues within the search system. This model aims to independently identify the top-k most relevant products from the entire product collection for a given query.

Model	AER%		Params
	Conflicts	Inaccuracy	
Claude 4 Sonnet	88.4	87.9	~400B
DS-R1-Qwen-32B	82.1	83.5	32B
Mixtral-8x7B	80.5	82.0	46.7B
LLaMA 2-34B	78.2	80.0	34B

Abbreviations: DS-R1-Qwen-32B = DeepSeek-R1-Distill-Qwen-32B; Params : number of parameters

Table 3: Content Quality Evaluation

- **Ranking or Retrieval Issue:** For queries with no "Exact" matches in actual results, we apply QRR-LLM to the top-*k* retrieved products. If any of them come out to be "Exact," it suggests a ranking or retrieval issue - the original search algorithm failed to surface the correct product despite its presence in the product collection. Note that, top-k products serve as a practical proxy for the entire product collection, balancing accuracy and efficiency, as running the QRR-LLM on millions of products would be computationally infeasible.

- **Selection Issue:** If neither the actual results nor the top-*k* retrieved products contain "Exact" matches, it points to a selection issue - either relevant items are missing from the product collection or described in a way that prevents them from being identified as relevant to the query.

Data Preparation and Architecture: We use the master dataset (section 3), containing ESCI labels for (query, product) pairs, to build training and evaluation sets. The model uses a Siamese two-tower architecture that generates separate embeddings for queries and products to learn similarity. We use SentenceBERT (Reimers and Gurevych, 2019) as the embedding model.

Training with Customized Triplet Loss: Model training is driven by a customized triplet loss function. A standard triplet loss aims to minimize the distance between an anchor (the query) and a positive example (a relevant product) while maximizing the distance between the anchor and a negative example (an irrelevant product). We extend this concept to incorporate the hierarchical nature of the ESCI labels. Our modified triplet loss function enforces the following distance relationship in the embedding space: $D(Q,E) < D(Q,S) < D(Q,C) < D(Q,I)$, where D is the Distance function, Q is the search query, and E, S, C, I are Exact, Substitute, Complement and Irrelevant product respectively, by carefully selecting triplets from products sampled with different ESCI labels during training. This nuanced loss function (Equation 2) encourages the model to learn a fine-grained representation of relevance, capturing the subtle distinctions between the ESCI categories. The loss function is described below:

$$\begin{aligned} L = &\max(0, m_1 + d(q,p^+) - d(q,p_1^-)) \\ &+ \max(0, m_2 + d(q,p^+) - d(q,p_2^-)) \\ &+ \max(0, m_3 + d(q,p^+) - d(q,p_3^-)) \end{aligned} \quad (2)$$

where, q is the query embedding, p^+ are positive samples (Exact products), p_1^- (Substitute), p_2^- (Complement), p_3^- (Irrelevant) are negative samples, $m_1 < m_2 < m_3$ are margins and d is the distance function (cosine distance).

We use total $200k$ queries and 4 pairs of (query,class) for every query, one for pair per each ESCI class as training data. We trained two models: one with relevance-aware loss (Eq. 2) and the other with an equal-margin, relevance-agnostic loss function, using the same training data. Typically, retrieval models are evaluated on NDCG metric, however, we evaluate performance on AER (1) metric since our objective is to identify missed recall issues with higher action-ability. We have performed online evaluation by taking the missed recall issues surfaced by PROBES over a period a month and validate manually for a sample of ~ 2000 issues. We observe that the model trained on relevance aware loss performs at 76% AER, whereas the model trained on relevance agnostic loss performs at 69% AER.

6 System evaluation and conclusion

In this paper, we introduced PROBES, an LLM based automated evaluation and modular system for online search systems. We introduced a new metric AER to measure effectiveness of such systems. PROBES performance metrics are as follows: (i) 76% AER (3 out of 4 issues surfaced are actionable), (ii) $\sim 3\%$ issue detection rate (300 issues found per 10000 queries), (iii) 80% reduction in insight generation time (5 days earlier to 1 day), (iv) 92% reduction in manual hours (~ 30 manual validation hours compared to ~ 400 hours earlier). Further, we also propose a novel approach for relevance measurement that leverages feature-level intent and query-category awareness, which shows substantial improvement over using only the raw

query for the task. We also introduce a relevance-label-aware loss function for the retrieval model, which helps achieve an 8% absolute improvement in AER over a generic loss function.

7 Limitations

PROBES presents a powerful, scalable approach to diagnosing search system issues by identifying precision and recall failures and tracing them to root causes like content quality, retrieval, ranking errors or selection gaps. Its ability to automate traditional manual evaluations drastically reduces human effort while providing actionable insights. However, PROBES still faces several limitations. It currently relies on static data and lacks user reviews and ratings, valuable for subjective queries like "best chair." Ambiguous queries will benefit from contextual cues or interactive disambiguation, such as leveraging recent user activity (e.g., browsing history or session patterns) or prompting follow-up clarifying questions to refine intent. We plan to incorporate image data for evaluation as well as expand to different languages as a future work. Finally, while PROBES effectively identifies and diagnoses search failures, it stops short of prescribing solutions - incorporating a recommendation module to suggest content corrections, retrieval adjustments, or ranking improvements will evolve PROBES into a proactive, end-to-end search optimization system.

References

Omar Alonso and Ricardo Baeza-Yates. 2011. Design and implementation of relevance assessments using crowdsourcing. In *European Conference on Information Retrieval*, pages 153–164.

Roi Blanco, Harry Halpin, Daniel M Herzig, Peter Mika, Jeffrey Pound, Henry S Thompson, and Thanh Tran Duc Tran. 2011. Repeatable and reliable search system evaluation using crowd-sourcing. In *Proceedings of the 34th international ACM SIGIR conference*, pages 923–932.

Zhuyun Dai and Jamie Callan. 2020. Context-aware document term weighting for ad-hoc search. In *The Web Conference 2020*, pages 1897–1907.

Kasra Hosseini, Thomas Kober, Josip Krapac, Roland Vollgraf, Weiwei Cheng, and Ana Peleteiro Ramallo. 2024. Retrieve, annotate, evaluate, repeat: Leveraging multimodal llms for large-scale product retrieval evaluation. In *Proceedings of the ACM Web Conference 2024*.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338.

Aniket Joshi, Cyrus Andre DSouza, Sejal Jain, Jitenkumar Babubhai Rana, and Promod Yenigalla. 2025. I-SEE: An instruction-tuned, SOP-enhanced quality evaluator for product content. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1379–1388, Suzhou (China). Association for Computational Linguistics.

Mingming Li, Shuai Liu, Ke Li, Jun Xu, and Huiming Wang. 2019. Semantic representation learning for e-commerce search. In *Proceedings of the SIGIR 2019 Workshop on eCommerce*.

Qi Liu, Atul Singh, Jingbo Liu, Cun Mu, and Zheng Yan. 2024. Towards more relevant product search ranking via large language models: An empirical study. In *Proceedings of the first workshop on Generative AI for E-Commerce 2024*.

Shichen Liu, Fei Xiao, Wenwu Ou, and Luo Si. 2017. Cascade ranking for operational e-commerce search. In *Proceedings of the 23rd ACM SIGKDD International Conference*, pages 1557–1565.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. Cedr: Contextualized embeddings for document ranking. In *Proceedings of the 42nd International ACM SIGIR Conference*, pages 1101–1104.

Navid Mehrdad, Hrushikesh Mohapatra, Mossaab Bagdouri, Prijith Chandran, Alessandro Magnani, Xunfan Cai, Ajit Puthenputhussery, Sachin Yadav, Tony Lee, ChengXiang Zhai, and Ciya Liao. 2024. Large language models for relevance judgment in product search. In *Proceedings of the 47th International ACM SIGIR Conference*.

Chandan K. Reddy, Lluís Màrquez, Fran Valero, Nikhil Rao, Hugo Zaragoza, Sambaran Bandyopadhyay, Arnab Biswas, Anlu Xing, and Karthik Subbian. 2022. Shopping queries dataset: A large-scale ESCI benchmark for improving product search.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084.

Ellen M Voorhees. 2001. The philosophy of information retrieval evaluation. *Workshop of the Cross-Language Evaluation Forum for European Languages*, pages 355–370.

Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. An empirical study of learning to rank techniques for e-commerce search. *ACM Transactions on Information Systems*, 37(1):1–30.

Shengnan Wu, Yongxiang Hu, Yingchuan Wang, Jiazhen Gu, Jin Meng, Liujie Fan, Zhongshi Luan, Xin Wang, and Yangfan Zhou. 2024. Combating missed recalls in e-commerce search: A cot-prompting testing approach. In *Companion Proceedings of the ACM Web Conference 2024*.

Yang Yan, Yihao Wang, Chi Zhang, Wenyuan Hou, Kang Pan, Xingkai Ren, Zelun Wu, Zhixin Zhai, Enyun Yu, Wenwu Ou, and Yang Song. 2024. Llm4pr: Improving post-ranking in search engine with large language models. In *Conference acronym 'XX*.

Priyanka Yang, Yin Bing, Choon Hui Teo, et al. 2019. Semantic product search. In *Proceedings of the 25th*

ACM SIGKDD International Conference, pages 2876–2885.

Ye Zhang, Daryl Wallace, and Matthew Patrick. 2016. Deep learning for search result relevance. In *Proceedings of the 39th International ACM SIGIR conference*, pages 1233–1234.

A Appendix

QRR LLM Output Example

Query: black charger for iPhone 12

QRR-LLM Input:

feature level intent with query category based importance

```
{
  "query_category": "compatibility",
  "feature_extraction": {
    "product category": {"value": "charger",
                  "importance": "must_have"},
    "compatible_device": {"value": "iPhone",
                  "importance": "must_have"},
    "color": {"value": "black",
          "importance": "approximate_is_okay"}
  }
}
```

Product Information
Black USB-C charger compatible with Samsung Galaxy devices

QRR-LLM Output:

```
{
 "reason": "This is a compatibility query where
  the charger must work with iPhone 12.
  Although the color matches (black),
  the product is only compatible with Samsung
  and not iPhone",
  "ESCI tag": "Irrelevant"
}
```

Query	Product Info	ESCI
55 inch tv wall mount	Wall mount for 37–70 inch TVs	E
black long sleeve shirt	Black Long Sleeve Shirt	E
laptop	laptop case	C
green cotton socks	red cotton socks	S
wire for iOS device	wire for Android	I
no calorie snacks	Chocolate Brownie	I

Table 4: Example output of QRR-LLM

Query Category	Precision	Recall
Feature search	0.94	0.90
Non-Product search	0.96	0.92
Product category search	0.99	0.97
Thematic search	0.93	0.90
Brand search	0.98	0.98
Exact search	0.99	0.99
Symptom search	0.88	0.84
Compatibility search	0.95	0.92
Relational search	0.93	0.88
Subjective search	0.85	0.82
No product category	0.84	0.80
Natural Language search	0.90	0.87
Slang or spelling error search	0.80	0.76
Negation search	0.86	0.83
Generic search	0.88	0.85

Table 5: Precision and Recall by Query Category (DeepSeek-R1-Distill-Qwen-32B)

Query Category	Examples
Feature search	leather jacket
Non-Product search	my orders, my refunds
Product type search	sandals, tv
Thematic search	Christmas decorations
Brand search	starbucks
Exact search	Zebronics XE Mouse
Symptom search	dry cough
Compatibility search	apple carplay adapter
Relational search	ronaldo jersey kids
Subjective search	best hair mask
No product type	124 cm, 4K
Slang search	chlr, nke
Negation search	chair without wheel
Generic search	home essentials

Table 6: Query Categories with Examples

Issue Type	Product Category	Model Explanation
Conflict	Chair	The product information states the item weight as 8.84 pounds. However, the bullet points mention the weight as 7.27 lbs, which is conflicting with the other source.
Conflict	Television	The product information states the display size is 65.0 inches, but the product product detail mentions the size as "83 Inch". This is a significant conflict in the display size specification.
Inaccuracy	Chair	The size value listed as 999 seems anomalous and inaccurate for a chair product.
Inaccuracy	Chair	The special feature listed as \"Toy\" seems anomalous for an outdoor chair intended for adults and children up to 250 lbs.

Table 7: Content quality model output examples

Language	Total			Train			Public Test		
	# Queries	# Judgements	Avg. Depth	# Queries	# Judgements	Avg. Depth	# Queries	# Judgements	Avg. Depth
English (US)	97,345	1,819,105	18.7	68,139	1,272,626	18.7	14,602	274,261	18.8
Spanish (ES)	15,180	356,578	23.5	10,624	249,721	23.5	2,277	53,494	23.5
Japanese (JP)	18,127	446,055	24.6	12,687	312,397	24.6	2,719	66,612	24.5
Overall	130,652	2,621,738	20.1	91,450	1,834,744	20.1	19,598	394,367	20.1

Table 8: Summary of the Shopping queries dataset for the tasks 2 and 3 (large version): the number of unique queries, the number of judgements, and the average number of judgements per query (Avg. Depth).

A.1 Prompt Templates

QIC Prompt Template

In an e-commerce website, customer intent is captured through search queries, such as:

```
<product_category>{pt}</product_category>
<search_query>{search_query}</search_query>
```

Task Overview

Your task is two-fold:

1. Classify the extracted information into one of the categories listed below, based on the definitions provided <list of query categories>
2. Perform feature extraction for each search query and its importance (`approximate_is_okay` or `must_have`) inferred from query category (1)

Rules

- Carefully review the definitions of each category before making a classification.
- If a search query does not clearly fit into any existing category, you are allowed to define and assign a new category that better represents the user's intent.
- Use the query type to determine the importance of the features, examples :
- For **thematic** queries, the `theme` is `must_have`.
- For **relational** queries, the related entity is important (e.g., "Messi shoes").
- For **negation** queries, the feature being negated is important (e.g., "headphones without wire" — here `without wire` is must_have).
- Make sure you do not infer, assume, or imply anything out of context that is not mentioned.
- Handle variations in case, singular/ plural forms, and spelling mistakes.

Categories

```
<categories>
<definitions and example of each category>
</categories>
```

Example

<2 examples for few short prompting>

Output Schema

```
{
  "searched_keyword": searched keyword,
  "reason": reason for classification,
  "category": type of search query,
  "feature_level_intent_with_importance" : feature_level_intent_with_importance
}
```

QRR Prompt Template

In an e-commerce website, the customer search is given as a search query, along with information about the products in the search results.
Your task is to classify these into four categories: **Exact**, **Complementary**, **Substitute**, and **Irrelevant**.

Input Schema

```
<search_query>{search_query}</search_query>
<product_information>{product_data}</product_information>
<Query Category and intent features with importance>{QIC output}
</Query Category and intent features with importance>
```

Categories

Exact: The product information is an exact match to the search query — all extracted features match exactly.

Substitute: The product is a substitute — the `approximate_is_okay` features may differ, but the core need is met.

Complementary: The product is a complementary item — such as an accessory or add-on to the main product in the query.

Irrelevant: The product is completely unrelated to the query — does not fulfill the intent or relevant features.

Example

<2 examples for few short prompting>

Output Schema

```
{
  "searched_keyword": searched keyword,
  "reason": reason for the category,
  "category": type of search of the search_query
}
```