Bridging Language and Items for Retrieval and Recommendation: Benchmarking LLMs as Semantic Encoders

Yupeng Hou[1†], **Jiacheng Li**[1†], **Xiangjun Fu**[1†],
Zhankui He[1], **An Yan**[1], **Xiusi Chen**[2], **Julian McAuley**[1]

[1]UC San Diego, [2]UC Los Angeles
{yphou,j9li,xif001,zhh004,ayan,jmcauley}@ucsd.edu xchen@cs.ucla.edu

Amazon Reviews 2023 dataset: https://amazon-reviews-2023.github.io
Benchmarking toolkit: https://github.com/hyp1231/BLaIR-Bench

Abstract

Feature engineering has long been central to recommender systems, yet effectively leveraging textual item features remains challenging. Recent advances in large language models (LLMs) have enabled their use as semantic encoders for recommendation, but their roles and behaviors in this setting are still not well understood. Prior studies often rely on general-purpose embedding benchmarks (*e.g.,* MTEB) when selecting LLMs, overlooking the unique characteristics of recommendation tasks. To address this gap, we introduce BLAIR, a comprehensive benchmark for evaluating LLMs as semantic encoders in recommendation scenarios. We contribute (1) a new large-scale Amazon Reviews 2023 dataset with over 570 million reviews and 48 million items, (2) a unified benchmark covering sequential recommendation, collaborative filtering, and product search, and (3) a new complex-query product search task featuring both semi-synthetic and real-world evaluation datasets. Experiments with 11 leading LLMs show that their rankings on BLAIR show little correlation with MTEB, highlighting the unique challenges of semantic encoding in recommendation.

1 Introduction

Feature engineering has long been central to modern recommender systems (Rendle, 2010; Chen et al., 2012; Cheng et al., 2016). However, effectively leveraging textual information (*e.g.,* item descriptions) has remained a challenge. While text features are rich in semantics, they are often noisy, unstructured, and difficult to integrate directly into recommendation models. Early approaches relied on keyword tagging (Mooney and Roy, 2000; Wang and Blei, 2011; Tang et al., 2012), which were largely heuristic and discarded substantial semantic information. Moreover, textual features were typically modeled using shallow statistical techniques, lacking the broader common-sense knowledge inherent in natural language.

With the rapid advancement of large language models (LLMs) (Hoffmann et al., 2022; Rae et al., 2021; Chowdhery et al., 2023; Touvron et al., 2023; Zhao et al., 2023), researchers have begun to employ them as semantic encoders for recommendation. Pretrained on large-scale corpora, LLMs capture extensive world knowledge and demonstrate strong capabilities in understanding and representing natural language. Consequently, an increasing number of studies has explored using LLM-encoded text representations as inputs to train recommendation models, a direction referred to as modality-based recommendation (Hou et al., 2022; Yuan et al., 2023). This paradigm has shown promising results across various tasks, including sequential recommendation (Hou et al., 2022; Yuan et al., 2023), collaborative filtering (Ren et al., 2024; Sheng et al., 2024), and product search (Li et al., 2021; Reddy et al., 2022).

Despite this progress, the role of LLMs as semantic encoders in recommendation remains not well understood. Existing studies typically adopt popular or recently released LLMs, or rely on general-purpose sentence embedding benchmarks such as MTEB (Muennighoff et al., 2023; Enevoldsen et al., 2025) for model selection. However, the requirements of recommendation differ substantially from those of generic text embedding tasks. (1) In recommendation, encoded representations are usually used as input features to train downstream models such as Transformer decoders for sequential recommendation, whereas in text embedding tasks, representations are more often consumed directly for similarity search or simple classifiers. (2) In recommendation, text inputs are often short and noisy (*e.g.,* item titles), requiring semantic encoders to provide strong disambiguation based on world knowledge, while text embedding benchmarks typically evaluate well-formed

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 3251–3265
July 2-7, 2026

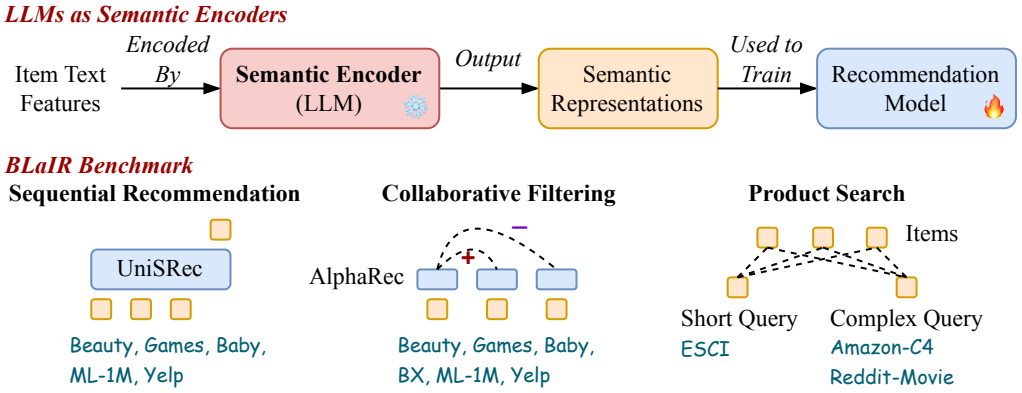

Figure 1: Overview of BLAIR benchmark. We evaluate LLMs as semantic encoders to convert item text features into semantic representations, which then train downstream recommendation models. The benchmark covers three scenarios: (i) sequential recommendation (*e.g.,* UniSRec (Hou et al., 2022)); (ii) collaborative filtering (*e.g.,* AlphaRec (Sheng et al., 2024)); and (iii) product search with short queries and complex queries.

sentences or paragraphs.

To address this gap, we present BLAIR, which stands for **B**ridging **La**nguage and **I**tems for **R**etrieval and **R**ecommendation, a comprehensive benchmark designed to evaluate LLMs as semantic encoders in recommendation scenarios (Figure 1). Our contributions are as follows:

• **A new large-scale e-commerce dataset: Amazon Reviews 2023.** Previous Amazon Reviews datasets (McAuley et al., 2015; He and McAuley, 2016; Ni et al., 2019) are widely used for benchmarking but were released several years ago (last released in 2018) and contain noisy metadata. We collect a new dataset with cleaned metadata, containing over 570 million reviews and 48 million items, with a cutoff date of September 2023.

• **A comprehensive benchmark: BLAIR.** We evaluate three representative recommendation scenarios: sequential recommendation, collaborative filtering, and product search. For each, we implement strong baselines and conduct extensive experiments with 11 top-performing LLMs on MTEB.

• **A new subtask: complex-query product search.** Existing product search datasets typically involve short queries. To align with emerging applications that require understanding long, complex, and ambiguous queries, we introduce a new subtask, complex-query product search. We construct two datasets: one semi-synthetic dataset derived from real user reviews and another curated from real-world forum posts.

Our results yield several notable insights: (1) LLM rankings on BLAIR are not positively correlated with their rankings on MTEB, highlighting that recommendation poses distinct challenges compared to general embedding benchmarks. (2) Even when downstream models have the same number of parameters, performance still scales with the size of the semantic encoder, though the effect weakens as task complexity grows. (3) We observe a strong correlation between results on semi-synthetic and real-world datasets for complex query product search tasks, implying that semi-synthetic data can be a cost-effective proxy for evaluation.

2 Related Work

Semantic Encoders in Recommendation. Researchers have long sought to better leverage semantically rich textual features such as titles and descriptions. Early works in content-based recommendation (Mooney and Roy, 2000; Gopalan et al., 2014) extracted keywords and learned their collaborative relations between users and items (Rendle, 2010; Wang and Blei, 2011; Chen et al., 2012; Wang et al., 2015). With advances in natural language processing (NLP), later studies began directly encoding text features using pretrained models to better capture language semantics (Chen et al., 2017; Hou et al., 2022; Yuan et al., 2023). However, such models often suffer from domain gaps between their pretraining data and recommendation scenarios (Hou et al., 2023). Recently, the emergence of LLMs with superior generalization capabilities has inspired researchers to explore using LLMs as semantic encoders in recommendation (Wei et al., 2024; Ren et al., 2024; Ren and Huang, 2024; Sheng et al., 2024). Nevertheless, questions remain about how the choice of LLM impacts the downstream performance.

Table 1: Statistics of different versions of Amazon Reviews datasets. #Tokens denotes the total number of text tokens obtained using the `cl100k_base` tokenizer.

Version	Amazon'13	Amazon'14	Amazon'18	Amazon'23
#Categories	28	24	29	**33**
Customer Reviews				
#Reviews	34,686,771	82,456,877	233,055,327	**571,544,897**
#Users	6,643,669	21,128,805	43,531,850	**54,514,264**
#Items	2,441,053	9,857,241	15,167,257	**48,185,153**
#Tokens	5.9B	9.1B	15.7B	**30.1B**
Min Time	Jun 1995	Jun 1996	Jun 1996	Jun 1996
Max Time	Mar 2013	July 2014	Oct 2018	**Sep 2023**
Item Metadata				
#Tokens	-	4.1B	7.9B	**30.7B**
# Meta	-	9,430,088	14,741,571	**35,393,189**

Benchmarking Text Embeddings. To comprehensively evaluate text embedding models across domains and tasks, researchers have developed benchmarks that aggregate multiple public datasets. An early example is BEIR (Thakur et al., 2021), which evaluates text embedding models on information retrieval tasks. MTEB (Muennighoff et al., 2023) and MMTEB (Enevoldsen et al., 2025) further expand the evaluation scope beyond retrieval to broader embedding applications such as classification and clustering. Recently, BRIGHT (Su et al., 2025) focuses on retrieval tasks that require reasoning to identify relevant documents. In this work, we aim to benchmark text embedding models as semantic encoders in recommendation scenarios. This is not merely a domain-specific adaptation but rather reflects that, as semantic encoders, LLMs require different capabilities.

Benchmarking LLMs for Recommendation. Existing efforts to benchmark LLMs for recommendation can be grouped into three categories: (1) benchmarking their real-world web interaction capabilities as shopping agents (Yao et al., 2022; Wang et al., 2025); (2) benchmarking their shopping-related knowledge and understanding of user intent (Jin et al., 2024); and (3) benchmarking LLMs themselves as recommendation models (Liu et al., 2023, 2024; Jiang et al., 2025; Liu et al., 2025). In contrast, to the best of our knowledge, we are the first to benchmark LLMs for encoding textual features in recommendation scenarios.

3 A Large-Scale E-Commerce Dataset: Amazon Reviews 2023

The Amazon Reviews datasets are among the most representative publicly available resources for evaluating recommendation models (McAuley et al., 2015; He and McAuley, 2016; Ni et al., 2019), offering rich item features. Given the wide use of the Amazon Reviews datasets, we aim to include them in our benchmark. However, existing versions of the Amazon Reviews datasets are outdated. The most recent one was last updated in 2018.

Table 2: Statistics of representative large-scale recommendation datasets.

Dataset	#Items	#Users	#Interactions
Netflix (Bennett et al., 2007)	17,770	480,189	100,480,507
Amazon'18 (Ni et al., 2019)	15,167,257	43,531,850	233,055,327
Google Local (Yan et al., 2023)	4,963,111	113,643,107	666,324,103
Tenrec (Yuan et al., 2022)	3,753,436	5,022,750	142,321,193
MicroLens (Ni et al., 2023)	1,142,528	34,492,051	1,006,528,709
Amazon Reviews 2023	48,185,153	54,514,264	571,544,897

To enhance the quality of our proposed benchmark, we collect a new version of the dataset, named *Amazon Reviews 2023*[1]. Table 1 presents key statistics comparing Amazon Reviews 2023 with earlier versions, while Table 2 compares Amazon Reviews 2023 with other representative large-scale datasets. The new dataset offers several notable improvements:

- *Larger scale*: Amazon Reviews 2023 is substantially larger than previous versions across all dimensions. Specifically, it contains $3.18\times$ more items and $2.58\times$ more text tokens in reviews and item metadata than the 2018 version.
- *More recent interactions*: The dataset contains more recent reviews from Amazon, extending the previous version with new data ranging from Oct 2018 to Sep 2023.
- *Richer and cleaner metadata*: We re-parsed the original HTML product pages into structured JSON format, resulting in metadata with more descriptive fields (*e.g.,* item descriptions and features) and multi-modal information (*e.g.,* product videos and images at multiple resolutions).
- *Finer-grained timestamps*: Previous datasets provided timestamps only at the day level, which can introduce inaccuracies for time-sensitive tasks such as sequential recommendation. In contrast, Amazon Reviews 2023 offers timestamps with millisecond precision.

4 BLAIR Benchmark

In this section, we present the structure of the proposed benchmark, BLAIR. The benchmark comprises three recommendation-related scenarios: sequential recommendation, collaborative filtering,

[1] https://amazon-reviews-2023.github.io/

Table 3: Data statistics of the BLAIR benchmark. Each row is a dataset. Cells that do not apply are marked *n/a*. "Real" indicates real-world data, while "Sync" indicates semi-synthetic data. More details about data split strategies (by timestamp, by ratio, or leave-last-out) can be found in Section C.1.

Dataset	Data Source / Subtask	Type	#Items	#Interactions			Split	Avg. #Items Per User	Avg. Item Metadata #Chars	Avg. Query #Chars
				Train	Val	Test				
Sequential Recommendation										
Beauty	Amazon'23	Real	43,978	82,800	14,268	7,698	By Timestamp (08/11/2021, 07/16/2022)	2.45	118.83	*n/a*
Games			115,813	2,127,563	189,268	215,809		3.41	85.40	
Baby			179,128	2,930,822	304,617	347,884		3.53	107.26	
ML-1M	ML-1M		3,043	983,412	6,040	6,040	Leave-Last-Out	164.82	23.78	
Yelp	Yelp		20,033	255,492	30,431	30,431		10.01	604.60	
Collaborative Filtering										
Beauty	Amazon'23	Real	43,978	41,906	31,429	31,431	By Ratio (4:3:3)	2.45	118.83	*n/a*
Games			115,813	1,013,056	759,792	759,792		3.41	85.40	
Baby			179,128	1,433,329	1,074,996	1,074,998		3.53	107.26	
BX	BX		5,335	101,183	75,804	75,862		40.31	27.57	
ML-1M	ML-1M		3,043	398,184	298,631	298,677		164.82	23.78	
Yelp	Yelp		20,033	126,542	94,906	94,906		10.01	604.60	
Product Search										
ESCI	Short	Real	1,367,729	*n/a*	*n/a*	27,643	*n/a*	*n/a*	96.57	22.46
Amazon-C4	Complex	Sync	1,058,417			21,223			99.81	229.89
R-Movies	Complex	Real	51,203			1,627			19.63	435.85

and product search. We first introduce the definitions and used datasets of each task in Section 4.1. Then, we describe in detail a newly introduced subtask under the product search scenario, namely complex-query product search, which contains long and ambiguous user queries (Section 4.2).

4.1 Task Definitions and Datasets

Given an item v with textual features x_v (such as title), the first step in using LLMs as semantic encoders is to transform these text features into dense representations: $\boldsymbol{e}_v = \text{LLM}(x_v)$. Here, $\text{LLM}(\cdot)$ denotes a pretrained and frozen LLM, and $\boldsymbol{e}_v \in \mathbb{R}^d$ is the resulting semantic representation for item v. The encoded representations $\{\boldsymbol{e}_v\}$ can then be used as inputs to train downstream recommendation models or directly for retrieval tasks. We consider three representative scenarios, each with its own task definition and dataset. The statistics of all datasets used in BLAIR are summarized in Table 3.

Sequential Recommendation. In the sequential recommendation task, a user's interaction history is represented as a time-ordered sequence of items $S_{t-1} = [v_1, v_2, \ldots, v_{t-1}]$. The goal is to predict the next item v_t that the user will interact with. Following common practice (Hou et al., 2022; Yuan et al., 2023; Sheng et al., 2024), we first add adapter layers to align the semantic embedding space with the recommendation space. Specifically, each item embedding is projected as $\boldsymbol{e}'_v = \text{Proj}_{\text{seq}}(\boldsymbol{e}_v) \in \mathbb{R}^{d'}$, where $\text{Proj}_{\text{seq}}(\cdot)$ denotes a learnable projection layer. This design ensures that the downstream recommendation model maintains a consistent number of parameters, regardless of the dimensionality of the underlying LLM. We then adopt the UniSRec (Hou et al., 2022) architecture to implement the sequential recommendation model. Formally, the next-item probability is defined as:

$$P(v_t|S_{t-1}) \propto \text{Trm}(\boldsymbol{e}'_{v_1}, \boldsymbol{e}'_{v_2}, \ldots, \boldsymbol{e}'_{v_{t-1}}) \cdot \boldsymbol{e}'_{v_t},$$

where $\text{Trm}(\cdot)$ is a Transformer decoder (Vaswani et al., 2017). The model is trained by maximizing the likelihood of the next items using a cross-entropy loss.

For sequential recommendation tasks, we select three categories from the Amazon Reviews 2023 dataset: All Beauty (*Beauty*), Video Games (*Games*), and Baby Products (*Baby*). These categories differ in both scale (0.7M, 4.5M, and 6.0M interactions, respectively) and domain (fashion, entertainment, and childcare). In addition, we include two widely used public datasets, Movielens-1M (*ML-1M*) (Harper and Konstan, 2015) and *Yelp* (Yelp Inc., 2020).

Collaborative Filtering. Given historical user-item interactions, the goal of collaborative filtering is to predict which item a user u is most likely to interact with next. Similar to the sequential recommendation setup, we add an adapter layer to obtain projected item representations $\boldsymbol{e}'_v = \text{Proj}_{\text{cf}}(\boldsymbol{e}_v) \in$

$\mathbb{R}^{d'}$. Following AlphaRec (Sheng et al., 2024), the user representation $\boldsymbol{e}'_u$ is computed as the average of the projected representations of items that the user has interacted with. The prediction probability is then defined as:

$$P(v|u) \propto \cos(\boldsymbol{W}\boldsymbol{e}'_u, \boldsymbol{W}\boldsymbol{e}'_v)/\tau,$$

where $\boldsymbol{W} \in \mathbb{R}^{d'' \times d'}$ is a linear layer, $\cos(\cdot,\cdot)$ denotes the cosine similarity function, and τ is a temperature hyperparameter. The model is trained using the InfoNCE loss with in-batch negative sampling (Chen et al., 2020).

For collaborative filtering tasks, we use the same five datasets as in the sequential recommendation scenario: *Beauty*, *Games*, and *Baby* from Amazon Reviews 2023, along with *ML-1M* and *Yelp*. Additionally, we include the widely used Book-Crossing dataset (*BX*) (Ziegler et al., 2005). We exclude Book-Crossing from the sequential recommendation scenario because it does not provide timestamps to order user interactions.

Product Search. In the product search task, given a user query q, the goal is to retrieve the most relevant items from a large corpus. The query is encoded using the same LLM as for items: $\boldsymbol{e}_q = \text{LLM}(q) \in \mathbb{R}^d$. For this task, we do not apply any adapter layers and directly use the raw LLM-encoded representations for retrieval. The relevance score between a query q and an item v is computed as $\boldsymbol{e}_q \cdot \boldsymbol{e}_v$ in a zero-shot manner.

We consider two subtasks under the product search scenario: (1) *short-query product search*, the conventional setting where user queries are short and specific, typically containing concise keywords; and (2) *complex-query product search*, where user queries are longer, more descriptive, and often ambiguous. For short-query product search, we use the *ESCI* dataset (Reddy et al., 2022). The latter subtask is newly introduced in this work, with detailed task definitions and evaluation datasets presented in Section 4.2.

4.2 New Subtask: Complex-Query Product Search

As the natural language understanding capabilities of large language models continue to grow, users are no longer limited to issuing short, keyword-based queries when searching for items. Instead, they can describe their needs in more natural and expressive ways, which can be long, ambiguous, or require reasoning to identify relevant items (*e.g.*, Buy it in ChatGPT[2] and Amazon Rufus[3]). To reflect this emerging trend, we introduce this subtask into our benchmark.

However, there are few public datasets that support evaluation in this setting. Collecting real paired complex user queries and their corresponding items is infeasible, as user chat histories are and should remain private. Therefore, we construct two datasets for this task: a semi-synthetic dataset derived from real user reviews in the Amazon Reviews 2023 (collected in Section 3), and a real-world dataset curated from Reddit forum posts.

Semi-Synthetic Complex Queries: Amazon-C4. The first dataset we introduce is *Amazon-C4*[4], short for **C**omplex **C**ontexts **C**reated by **C**hatGPT. The key idea is that user reviews often contain detailed intentions and rich contextual information, which can be repurposed into complex user queries. However, directly using reviews as queries is problematic. Reviews are usually written in a reflective tone, not in the style of a user query. In addition, reviews may include information about the target item that causes data leakage. To address this, we use large language models to rephrase user reviews into natural, query-like forms, creating the semi-

Table 4: Comparison of user queries for the same item in the ESCI and Amazon-C4 datasets.

Item ASIN/ID	B07RJZNY5C
Query (ESCI)	salt gun
Query (Amazon-C4)	I want a gun that I can use while gardening to get rid of stink bugs, ants, flies, and spiders in my house. It needs to be amazing and help me feel less scared.
Metadata	BUG-A-SALT 3.0 Black Fly Edition.

Table 5: An example from the Reddit-Movie dataset.

IMDb ID	tt2582802
Query (Reddit-Movie)	Movies with a character that is very determined and goes through a vigorous training to be very good/the best at something? It doesn't necessarily have to be a training, it could be the character learning by trial and error, experience and doing the thing himself. Pretty much any genre is okay, action, sci fi, thriller, horror, ...
Movie	Whiplash (2014)

[2] https://openai.com/index/buy-it-in-chatgpt/
[3] https://www.amazon.com/rufus/
[4] https://huggingface.co/datasets/McAuley-Lab/Amazon-C4

Table 6: Performance comparison of LLMs as semantic encoders across different recommendation scenarios. "Seq. Rec.", "Col. Fil.", "Short", and "Complex" stand for sequential recommendation (5 datasets), collaborative filtering (6 datasets), short-query product search (1 dataset), and complex-query product search (2 datasets), respectively. The best results in each column are highlighted in **bold**.

Model	Rank (Borda)*	Avg. (Overall)	Avg. (Task)	Seq. Rec.	Col. Fil.	Product Search	
						Short	Complex
Open-Source Models (< 1B parameters)							
`FacebookAI/roberta-large`	11 (15.0)	0.0263	0.0190	0.0393	0.0269	0.0096	0.0001
`Qwen/Qwen3-Embedding-0.6B`	10 (35.5)	0.0507	0.0829	0.0415	0.0274	0.1876	0.0750
`princeton-nlp/sup-simcse-roberta-large`	9 (37.0)	0.0397	0.0536	0.0426	0.0265	0.1063	0.0389
`sentence-transformers/sentence-t5-large`	8 (42.5)	0.0513	0.0801	0.0418	0.0304	0.1691	0.0790
Open-Source Models ($\geq$ 1B parameters)							
`Qwen/Qwen3-Embedding-4B`	7 (69.5)	0.0620	0.1036	0.0416	0.0350	0.2258	0.1120
`Qwen/Qwen3-Embedding-8B`	6 (54.0)	0.0637	0.1069	0.0415	0.0362	0.2328	0.1172
`Salesforce/SFR-Embedding-Mistral`	4 (98)	0.0679	0.1160	0.0433	0.0372	**0.2560**	0.1273
`intfloat/e5-mistral-7b-instruct`	3 (101)	0.0666	0.1120	0.0434	0.0377	0.2437	0.1232
`GritLM/GritLM-7B`	2 (105)	**0.0685**	**0.1161**	0.0434	**0.0385**	0.2537	**0.1290**
Proprietary Models							
`gemini-embedding-001` (Google)	5 (96.5)	0.0629	0.1040	0.0434	0.0355	0.2233	0.1140
`text-embedding-3-large` (OpenAI)	**1 (116)**	0.0665	0.1112	**0.0440**	0.0366	0.2366	0.1278

* Rank (↓) is determined by the Borda Count score. For the Borda score and all other metrics, higher values are better (↑).

synthetic dataset.

Prompt for complex query synthesizing in Amazon-C4.

```
Given a review from Amazon, can you rephrase
it with a first-person tone as if you are the
customer looking for a product? Give me the
rephrased output without saying anything else.
Note that the name of the product must not
show in the output since you are looking for
it. You should ignore irrelevant information
that doesn't help with the rewriting.
```

Specifically, we uniformly sample more than 20,000 5-star reviews from the Amazon Reviews 2023 dataset with at least 100 characters per review to ensure sufficient detail. We use ChatGPT to rephrase each review into a first-person query expressing the intent to find a suitable item (see Prompt 1 for the prompt). We then filter out cases where ChatGPT fails to perform the conversion. A comparison between a user query in Amazon-C4 and ESCI for the same target item is shown in Table 4.

Real-World Complex Queries: Reddit-Movies. The second dataset, Reddit-Movies (*R-Movies*), is curated from real forum discussions. As Amazon-C4 is semi-synthetic, one may question whether the generated queries are realistic enough for model evaluation. To complement it, we build Reddit-Movies based on the `reddit_movie_large_v1` dataset (He et al., 2023), which collects real Reddit discussions about movies and links each movie to its IMDb ID. Specifically, we keep only posts where users explicitly ask for movie suggestions (with the label `is_seeker=True`) that have at least one valid response with more than 20 upvotes, which indicates a believable answer. For item metadata, we use the movie title and release year. The final dataset contains more than 20,000 query-item pairs, split into training, validation, and test sets following the original data split (He et al., 2023). Note that only the test split is used for evaluation in our benchmark.

5 Experiments

5.1 Experimental Setup

We evaluate 11 top-performing LLMs from the widely used general text embedding benchmark MTEB (Muennighoff et al., 2023; Enevoldsen et al., 2025): (1) open-source LLMs with $< 1B$ parameters (RoBERTa (Liu et al., 2019), SimCSE (Gao et al., 2021), Sentence-T5 (Ni et al., 2022)); (2) open-source LLMs with $\geq 1B$ parameters (E5 (Wang et al., 2024), SFT-Embedding (Meng et al., 2024), GritLM (Muennighoff et al., 2025)); and (3) proprietary text embedding models from Google (`gemini-embedding-001`) and OpenAI (`text-embedding-3-large`). Among these, we further benchmark Qwen3-Embedding models of varying sizes (0.6B, 4B, and 8B) (Zhang et al., 2025) to better understand the scaling behavior of

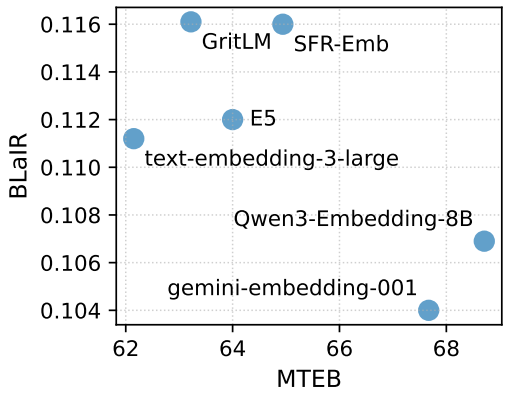

Figure 2: Performance correlation between MTEB (eng, v2) and BLaIR (avg. per task).

LLM-based semantic encoders. Please refer to Section C for more implementation details.

5.2 Main Results

We present the benchmarking results in Table 6, with detailed per-dataset results in Section D. Several key observations emerge:

(1) The ranking on BLAIR shows little correlation with MTEB. By analyzing the Spearman correlation between the per-task average scores on BLAIR and MTEB (eng, v2) (Enevoldsen et al., 2025), we observe a correlation coefficient of -0.476 with $p = 0.233$ (see Figure 2). This indicates that the performance of LLMs on general text embedding benchmarks does not necessarily reflect their effectiveness as semantic encoders for recommendation tasks. We attribute this to three possible factors. First, MTEB includes tasks such as clustering and summarization that are not directly aligned with the objectives of recommendation systems. Second, representations in recommendation tasks are expected to be highly discriminative (Hou et al., 2022; Sheng et al., 2024), a property not emphasized in MTEB. Third, as competition on MTEB intensifies, models may become overfitted to its specific tasks, resulting in reduced generalization to other domains.

(2) The scaling behavior of semantic encoders weakens when downstream tasks become complex. According to scaling laws, larger models typically achieve better performance (Kaplan et al., 2020), yet this trend is not consistently observed in our benchmark. We ensure a fair comparison by controlling the downstream model parameters across different LLM-based semantic encoders, using adaptors before feeding the representations into the downstream models. In tasks with simple downstream architectures, such as collaborative filtering (a single linear layer; see Section 4.1), larger LLMs generally yield higher performance. However, in more complex settings like sequential recommendation, where the downstream model involves a Transformer decoder, the scaling trend becomes less pronounced. For instance, models in the Qwen3-Embedding family show comparable performance across different sizes on the sequential recommendation task. A discussion on this counterintuitive observation is provided in Section E.

(3) The semi-synthetic dataset Amazon-C4 exhibits a strong linear correlation with the real-world dataset Reddit-Movie on the complex-query product search task. By computing the Pearson correlation coefficient between the NDCG@100 scores of all evaluated models on Amazon-C4 and Reddit-Movie (Table 10), we find a strong linearly positive correlation of 0.94 ($p < 0.01$). This indicates that Amazon-C4, despite being semi-synthetic and covering domains different from that in real-world complex queries, effectively captures essential characteristics of such queries. These results demonstrate that Amazon-C4 serves as a reliable proxy for evaluating (and potentially training) LLMs on complex-query product search tasks.

(4) `text-emb-3-large` demonstrates unexpectedly strong generalization on BLAIR. Developed by a leading company, `text-emb-3-large` is one of the most widely used text embedding models. Despite ranking only 42nd (as of Oct 6, AoE) on MTEB (English, v2), which is relatively low compared to other models evaluated in this study, raising doubts about its overall capability, it exhibits remarkably good performance across diverse datasets and tasks in BLAIR. Although its average performance is not the highest, its Borda Count ranking reveals consistently strong results across different scenarios. These observations suggest that `text-emb-3-large` possesses notable generalization ability and further indicate that evaluating LLM generalization remains an open and non-trivial challenge.

5.3 Performance w.r.t. Item Metadata

Although numerous item metadata attributes are available for use as item features in the collected Amazon Reviews 2023 dataset, in BLAIR we include only the item title, as it is the most universally available attribute, providing high informa-

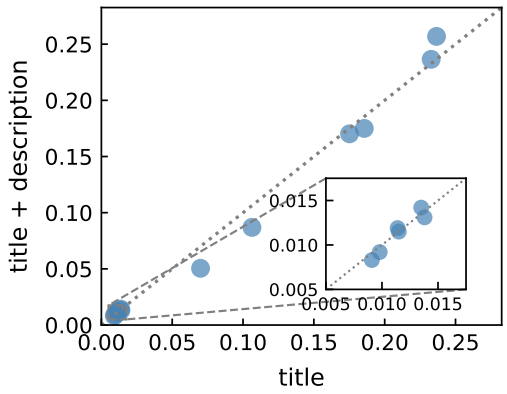

Figure 3: Comparison of performance when using different item metadata. Each point represents a model on a specific dataset.

Table 7: Comparison between adaptors using matryoshka representation learning (MRL) and principal component analysis (PCA).

Model	Adaptor	Seq. Rec.	Col. Fil.
`Qwen3-Embedding-8B`	PCA	**0.0415**	0.0362
	MRL	0.0359	**0.0392**
`gemini-embedding-001`	PCA	**0.0434**	**0.0355**
	MRL	0.0384	0.0313
`text-embedding-3-large`	PCA	**0.0440**	0.0366
	MRL	0.0383	**0.0379**

tion density in a concise and easily identifiable form. In this section, we investigate whether incorporating additional metadata can further enhance the performance of LLM-based semantic encoders. Specifically, we conduct experiments using both the item title and description as input features, and compare the results with those obtained using the title alone. We evaluate three representative models (`text-emb-3-large`, `Qwen3-Emb-8B`, and `SimCSE`) across four datasets: Games (sequential recommendation), Games (collaborative filtering), Amazon-C4, and ESCI.

The results in Figure 3 show that incorporating both the title and description does not consistently improve performance relative to using only the title. We attribute this to two factors. First, longer text inputs may introduce additional noise, making it harder for the model to extract relevant information. Second, LLM world knowledge may already capture some of the information provided in item descriptions. These observations suggest that effectively leveraging rich, heterogeneous, and noisy item metadata in semantic encoders remains a challenging open problem.

5.4 Performance w.r.t. Adaptor Design

In our experiments, to ensure fair comparisons across different LLM-based semantic encoders, we employ adaptors to align their representation dimensions. We use principal component analysis (PCA) with whitening as the default adaptor due to its simplicity and efficiency (Huang et al., 2021; Su et al., 2021). Meanwhile, several LLMs are capable of directly producing low-dimensional representations through matryoshka representation learning (MRL) (Kusupati et al., 2022). In this section, we compare the performance of PCA and MRL on three representative models (`Qwen3-Embedding-8B`, `gemini-embedding-001`, and `text-embedding-3-large`) across all recommendation datasets that require an adaptor.

As shown in Table 7, for tasks involving complex downstream models such as sequential recommendation, PCA outperforms MRL, likely because whitening encourages more distinguishable representations. In contrast, for tasks with simpler architectures like collaborative filtering, MRL tends to achieve better results, possibly due to its ability to preserve task-relevant information in low-dimensional spaces. These findings suggest that the choice of adaptor should be guided by the complexity of the downstream task and model architecture.

6 Conclusion

In this work, we introduced BLAIR, a comprehensive benchmark for evaluating LLMs as semantic encoders in recommendation scenarios. We contributed (1) a new large-scale Amazon Reviews 2023 dataset, (2) a unified benchmark spanning sequential recommendation, collaborative filtering, and product search, and (3) a novel complex-query product search task with both semi-synthetic and real-world evaluation datasets. Our experiments with 11 leading LLMs revealed several key insights: (1) LLM rankings on BLAIR show little correlation with general text embedding benchmarks like MTEB, indicating that the role of semantic encoders poses unique challenges. (2) Scaling laws apply to semantic encoders even when downstream models are parameter-controlled, though the effect weakens with task complexity increases. (3) Our semi-synthetic Amazon-C4 dataset shows strong correlation with real-world complex queries, indicating that it can serve as a reliable evaluation proxy and may also benefit the training of LLMs for complex-query tasks.

Acknowledgement

This work is partially supported by NSF IIS-2432486.

Limitations

While BLAIR provides a comprehensive evaluation framework for LLMs as semantic encoders in recommendation, several limitations should be acknowledged. First, our current benchmark focuses exclusively on English-language data. Given the global nature of e-commerce and recommender systems, multilingual evaluation would be valuable for assessing the cross-lingual capabilities of semantic encoders. Second, due to computational budget constraints, our experiments are limited to 11 LLMs and a subset of categories from the Amazon Reviews 2023 dataset. A more extensive evaluation covering additional state-of-the-art models and a broader range of categories would provide deeper insights into the generalizability of our findings.

Ethical Considerations

Our work prioritizes user privacy and data ethics in several ways. For the Amazon Reviews 2023 dataset, we exclusively collect publicly available information that users have explicitly chosen to share. We do not include any content that users have opted to keep private or restricted. Importantly, our dataset does not contain user metadata to prevent potential misuse for user profiling or privacy violations. The dataset focuses solely on publicly visible product reviews and item metadata, which are essential for recommendation research while minimizing privacy risks.

Regarding the BLAIR benchmark, all evaluation tasks are constructed using publicly released datasets that have been widely adopted by the research community. We ensure that our benchmark adheres to the original data collection and usage policies of these datasets. While recommender systems can potentially reinforce biases or create filter bubbles, our benchmark is designed to evaluate semantic encoding capabilities rather than deployment-ready recommendation systems. We encourage researchers using BLAIR to consider fairness, transparency, and potential societal impacts when developing real-world applications based on insights from this benchmark.

References

James Bennett, Stan Lanning, and 1 others. 2007. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. New York.

Tianqi Chen, Weinan Zhang, Qiuxia Lu, Kailong Chen, Zhao Zheng, and Yong Yu. 2012. Svdfeature: a toolkit for feature-based collaborative filtering. *JMLR*, 13(1):3619–3622.

Ting Chen, Liangjie Hong, Yue Shi, and Yizhou Sun. 2017. Joint text embedding for personalized content-based recommendation. *arXiv preprint arXiv:1706.01084*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607.

Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, and 1 others. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pages 7–10.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and 1 others. 2023. Palm: Scaling language modeling with pathways. *JMLR*, 24(240):1–113.

Pierre Colombo, Nathan Noiry, Ekhine Irurozki, and Stéphan Clémençon. 2022. What are the best systems? new perspectives on NLP benchmarking. In *NeurIPS*.

Kenneth Enevoldsen, Isaac Chung, Imene Kerboua, Márton Kardos, Ashwin Mathur, David Stap, Jay Gala, Wissam Siblini, Dominik Krzemiński, Genta Indra Winata, and 1 others. 2025. Mmteb: Massive multilingual text embedding benchmark. In *ICLR*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *EMNLP*, pages 6894–6910.

Prem Gopalan, Laurent Charlin, and David M Blei. 2014. Content-based recommendations with poisson factorization. In *NIPS*, volume 27.

F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW*, pages 507–517.

Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Majumder, Nathan Kallus, and Julian McAuley. 2023. Large language models as zero-shot conversational recommenders. In *CIKM*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, and 1 others. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Yupeng Hou, Zhankui He, Julian McAuley, and Wayne Xin Zhao. 2023. Learning vector-quantized item representation for transferable sequential recommenders. In *WWW*.

Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2022. Towards universal sequence representation learning for recommender systems. In *KDD*.

Junjie Huang, Duyu Tang, Wanjun Zhong, Shuai Lu, Linjun Shou, Ming Gong, Daxin Jiang, and Nan Duan. 2021. Whiteningbert: An easy unsupervised sentence embedding approach. In *Findings of EMNLP*, pages 238–244.

Chumeng Jiang, Jiayin Wang, Weizhi Ma, Charles L. A. Clarke, Shuai Wang, Chuhan Wu, and Min Zhang. 2025. Beyond utility: Evaluating LLM as recommender. In *WWW*, pages 3850–3862.

Yilun Jin, Zheng Li, Chenwei Zhang, Tianyu Cao, Yifan Gao, Pratik Jayarao, Mao Li, Xin Liu, Ritesh Sarkhel, Xianfeng Tang, Haodong Wang, Zhengyang Wang, Wenju Xu, Jingfeng Yang, Qingyu Yin, Xian Li, Priyanka Nigam, Yi Xu, Kai Chen, and 3 others. 2024. Shopping MMLU: A massive multi-task online shopping benchmark for large language models. In *NeurIPS*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*, abs/2001.08361.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham M. Kakade, Prateek Jain, and Ali Farhadi. 2022. Matryoshka representation learning. In *NeurIPS*.

Sen Li, Fuyu Lv, Taiwei Jin, Guli Lin, Keping Yang, Xiaoyi Zeng, Xiao-Ming Wu, and Qianli Ma. 2021. Embedding-based product retrieval in taobao search. In *KDD*, pages 3181–3189.

Jiao Liu, Zhu Sun, Shanshan Feng, and Yew-Soon Ong. 2024. Language model evolutionary algorithms for recommender systems: Benchmarks and algorithm comparisons. *CoRR*, abs/2411.10697.

Junling Liu, Chao Liu, Peilin Zhou, Qichen Ye, Dading Chong, Kang Zhou, Yueqi Xie, Yuwei Cao, Shoujin Wang, Chenyu You, and Philip S. Yu. 2023. Llmrec: Benchmarking large language models on recommendation task. *CoRR*, abs/2308.12241.

Qijiong Liu, Jieming Zhu, Lu Fan, Kun Wang, Hengchang Hu, Wei Guo, Yong Liu, and Xiao-Ming Wu. 2025. Benchmarking llms in recommendation tasks: A comparative evaluation with conventional recommenders. *CoRR*, abs/2503.05493.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *SIGIR*, pages 43–52.

Rui Meng, Ye Liu, Shafiq Rayhan Joty, Caiming Xiong, Yingbo Zhou, and Semih Yavuz. 2024. SFR-Embedding-Mistral:Enhance Text Retrieval with Transfer Learning. https://www.salesforce.com/blog/sfr-embedding/.

Raymond J Mooney and Loriene Roy. 2000. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 195–204.

Niklas Muennighoff, Hongjin Su, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. 2025. Generative representational instruction tuning. In *ICLR*.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. Mteb: Massive text embedding benchmark. In *EACL*, pages 2014–2037.

Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B. Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of ACL*, pages 1864–1874.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *EMNLP*, pages 188–197.

Yongxin Ni, Yu Cheng, Xiangyan Liu, Junchen Fu, Youhua Li, Xiangnan He, Yongfeng Zhang, and Fajie Yuan. 2023. A content-driven micro-video recommendation dataset at scale. *arXiv preprint arXiv:2309.15379*.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, and 1 others. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

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. *arXiv preprint arXiv:2206.06588*.

Xubin Ren and Chao Huang. 2024. Easyrec: Simple yet effective language models for recommendation. *arXiv preprint arXiv:2408.08821*.

Xubin Ren, Wei Wei, Lianghao Xia, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Representation learning with large language models for recommendation. In *WWW*, pages 3464–3475.

Steffen Rendle. 2010. Factorization machines. In *ICDM*, pages 995–1000. IEEE.

Leheng Sheng, An Zhang, Yi Zhang, Yuxin Chen, Xiang Wang, and Tat-Seng Chua. 2024. Language models encode collaborative signals in recommendation. In *ICLR*.

Hongjin Su, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han-yu Wang, Haisu Liu, Quan Shi, Zachary S. Siegel, Michael Tang, Ruoxi Sun, Jinsung Yoon, Sercan Ö. Arik, Danqi Chen, and Tao Yu. 2025. BRIGHT: A realistic and challenging benchmark for reasoning-intensive retrieval. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening sentence representations for better semantics and faster retrieval. *CoRR*, abs/2103.15316.

Jie Tang, Sen Wu, Jimeng Sun, and Hang Su. 2012. Cross-domain collaboration recommendation. In *KDD*, pages 1285–1293.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, volume 30.

Chong Wang and David M Blei. 2011. Collaborative topic modeling for recommending scientific articles. In *KDD*, pages 448–456.

Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *KDD*, pages 1235–1244.

Jiangyuan Wang, Kejun Xiao, Qi Sun, Huaipeng Zhao, Tao Luo, Jiandong Zhang, and Xiaoyi Zeng. 2025. Shoppingbench: A real-world intent-grounded shopping benchmark for llm-based agents. *CoRR*, abs/2508.04266.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving text embeddings with large language models. In *ACL*, pages 11897–11916.

Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Llmrec: Large language models with graph augmentation for recommendation. In *WSDM*, pages 806–815.

An Yan, Zhankui He, Jiacheng Li, Tianyang Zhang, and Julian McAuley. 2023. Personalized showcases: Generating multi-modal explanations for recommendations. In *SIGIR*, pages 2251–2255.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. In *NeurIPS*.

Yelp Inc. 2020. Yelp Dataset. https://business.yelp.com/data/resources/open-dataset/.

Guanghu Yuan, Fajie Yuan, Yudong Li, Beibei Kong, Shujie Li, Lei Chen, Min Yang, Chenyun Yu, Bo Hu, Zang Li, and 1 others. 2022. Tenrec: A large-scale multipurpose benchmark dataset for recommender systems. In *NeurIPS*, volume 35, pages 11480–11493.

Zheng Yuan, Fajie Yuan, Yu Song, Youhua Li, Junchen Fu, Fei Yang, Yunzhu Pan, and Yongxin Ni. 2023. Where to go next for recommender systems? id-vs. modality-based recommender models revisited. In *SIGIR*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *CoRR*, abs/2506.05176.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, and 3 others. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. In *WWW*, pages 22–32.

Appendices

A Benchmarking Toolkit

To facilitate reproducibility and simplify the integration of new models and datasets, we have developed a benchmarking toolkit available at https://github.com/hyp1231/BLaIR-Bench. The toolkit includes implementations of all datasets, tasks, and evaluation metrics used in this study, along with scripts for model training and evaluation. We welcome community contributions to further extend the benchmark.

B Dataset Collection Details

The Amazon Reviews 2023 dataset is collected using a large-scale, user-centric pipeline. We first sample user identifiers from public Amazon pages and then iteratively collect all review pages associated with each user. This user-level design ensures completeness of individual review histories, which is particularly beneficial for recommendation and user modeling research.

However, the review coverage for a given item may be incomplete if some users are not included in our user pool. Additionally, due to the temporal span of the collecting process, recently posted reviews might be missing even before the dataset cut-off date.

C Experimental Setup

C.1 Data Split and Preprocessing

Sequential Recommendation. We group all user-item interactions by user ID and sort them by timestamp. For each user, we truncate their interaction history to the most recent 50 items (200 for ML-1M). Dataset-specific preprocessing steps are as follows:

- **Amazon Reviews 2023.** To preserve the natural distribution, we do not filter out any users or items based on interaction counts. Since sequential recommendation is time-sensitive, we split the dataset by time. To be specific, we determined two cut-off timestamps `1628643414042` and `1658002729837` that split all the reviews in Amazon Reviews 2023 in a ratio of $8:1:1$. These timestamps are shared across the three categories: All Beauty, Video Games, and Baby Products.
- **Movielens-1M.** We iteratively remove users and items with fewer than 20 interactions until convergence, then apply the leave-last-out strategy: the last two interactions per user are used for validation and testing, while all previous interactions are used for training.
- **Yelp.** We iteratively remove users and items with fewer than 5 interactions until convergence and apply the same leave-last-out strategy as ML-1M. Using the 2020 competition dataset, we keep only interactions from 2019.

Collaborative Filtering. Following Sheng et al. (2024), we discard timestamps and split user-item interactions into training, validation, and test sets with a 4:3:3 ratio. For ML-1M and Book-Crossing, users and items with fewer than 20 interactions are iteratively removed. For Yelp, the threshold is 5. No filtering is applied to the three Amazon Reviews 2023 categories.

Short-Query Product Search (ESCI). We use the ESCI dataset (Reddy et al., 2022) and retain only *<query, item>* pairs with the label "Exact". We adopt the small version due to its higher-quality queries, filter out non-English queries, and keep only those whose timestamps fall within our predefined test period (after `1658002729837`). Each item ID (`ASIN`) is linked to its metadata in Amazon Reviews 2023. We construct a large multi-category candidate pool by sampling 50 in-domain items (from the same category) for each ground-truth pair. Ranking is then performed over all multi-domain candidates for each query.

Complex-Query Product Search.

- **Amazon-C4.** The data synthesizing process is described in Section 4.2. After that, we follow the same multi-category candidate sampling strategy as ESCI, sampling 50 in-domain items for each ground-truth pair. Ranking is computed over both in-domain and cross-domain candidates.
- **Reddit-Movie.** We use the `reddit_movie_large_v1` dataset (He et al., 2023), which contains multi-turn conversations where users seek movie recommendations. We retain only conversations with the label `is_seeker=True` (indicating the user is seeking movie recommendations) and apply several filtering steps: (1) contexts

Table 8: Sequential Recommendation (NDCG@10) detailed results. Best per column in **bold**, second best underlined.

Model / Datasets	All_Beauty	Video_Games	Baby_Products	ML1M	Yelp
Open-Source Models (< 1B parameters)					
FacebookAI/roberta-large	0.0177	0.0113	0.0070	0.1215	0.0391
Qwen/Qwen3-Embedding-0.6B	0.0224	0.0126	0.0072	0.1266	0.0385
princeton-nlp/sup-simcse-roberta-large	0.0232	0.0114	0.0069	0.1312	0.0404
sentence-transformers/sentence-t5-large	0.0212	0.0125	0.0075	0.1275	0.0405
Open-Source Models ($\geq$ 1B parameters)					
Qwen/Qwen3-Embedding-4B	0.0212	0.0127	0.0075	0.1276	0.0392
Qwen/Qwen3-Embedding-8B	0.0216	**0.0138**	0.0072	0.1248	0.0400
Salesforce/SFR-Embedding-Mistral	0.0231	0.0131	0.0076	0.1311	0.0416
intfloat/e5-mistral-7b-instruct	0.0238	0.0136	0.0072	0.1324	0.0398
GritLM/GritLM-7B	0.0216	0.0133	0.0074	0.1321	**0.0426**
Proprietary Models					
gemini-embedding-001 (Google)	**0.0241**	0.0136	0.0073	0.1297	0.0421
text-embedding-3-large (OpenAI)	0.0237	0.0135	**0.0078**	**0.1330**	0.0418

Table 9: Collaborative Filtering (NDCG@20) detailed results. Best per column in **bold**, second best underlined.

Model / Datasets	All_Beauty	Video_Games	Baby_Products	Book_Crossing	ML1M	Yelp
Open-Source Models (< 1B parameters)						
FacebookAI/roberta-large	0.0183	0.0087	0.0037	0.0338	0.0658	**0.0310**
Qwen/Qwen3-Embedding-0.6B	0.0220	0.0090	0.0047	0.0296	0.0713	0.0280
princeton-nlp/sup-simcse-roberta-large	0.0205	0.0091	0.0040	0.0285	0.0676	0.0295
sentence-transformers/sentence-t5-large	0.0202	0.0101	0.0043	0.0329	0.0869	0.0280
Open-Source Models ($\geq$ 1B parameters)						
Qwen/Qwen3-Embedding-4B	0.0219	0.0099	0.0041	0.0406	0.1047	0.0285
Qwen/Qwen3-Embedding-8B	0.0226	0.0098	0.0042	0.0427	0.1097	0.0284
Salesforce/SFR-Embedding-Mistral	0.0236	0.0107	0.0049	0.0416	0.1142	0.0284
intfloat/e5-mistral-7b-instruct	0.0232	0.0110	0.0049	0.0421	0.1162	0.0287
GritLM/GritLM-7B	0.0226	0.0110	0.0046	0.0425	**0.1216**	0.0287
Proprietary Models						
gemini-embedding-001 (Google)	**0.0242**	0.0108	**0.0051**	0.0371	0.1054	0.0304
text-embedding-3-large (OpenAI)	0.0238	**0.0115**	0.0050	**0.0436**	0.1071	0.0286

Table 10: Product Search (NDCG@100) detailed results. Best per column in **bold**, second best underlined.

Model / Datasets	Amazon-C4	ESCI	Reddit-Movies
Open-Source Models (< 1B parameters)			
FacebookAI/roberta-large	0.0001	0.0096	0.0000
Qwen/Qwen3-Embedding-0.6B	0.1288	0.1876	0.0211
princeton-nlp/sup-simcse-roberta-large	0.0700	0.1063	0.0078
sentence-transformers/sentence-t5-large	0.1219	0.1691	0.0360
Open-Source Models ($\geq$ 1B parameters)			
Qwen/Qwen3-Embedding-4B	0.1663	0.2258	0.0576
Qwen/Qwen3-Embedding-8B	0.1752	0.2328	0.0591
Salesforce/SFR-Embedding-Mistral	0.1836	**0.2560**	0.0710
intfloat/e5-mistral-7b-instruct	0.1759	0.2437	0.0705
GritLM/GritLM-7B	0.1845	0.2537	**0.0734**
Proprietary Models			
gemini-embedding-001 (Google)	0.1701	0.2233	0.0578
text-embedding-3-large (OpenAI)	**0.1856**	0.2366	0.0699

User query	"Over the Top Bonkers Action Movies?. I recently saw the new *Suicide Squad* movie and loved it. It had all the qualities I like in an action movie: violence, shock factor, comedy, and overall bizarreness. What are some other action movies with these qualities? *Deadpool*, *Kick-Ass*, *Turbo Kid*, *Tropic Thunder*, and *Drive Angry* are some of my other favorites."
Relevant items	*Nobody* (2021), *Hardcore Henry* (2015), *Slow Moe* (2010), *Dredd* (2012)
Top-3 retrieved by GritLM-7B	*Ninjas vs. Zombies* (2008), *Cowboys vs. Zombies* (2014), *Cowboys vs. Vampires* (2010)
Analysis	The retrieved movies appear to match the query only at a superficial level. For example, they contain unusual or fantastical entities such as zombies, ninjas, and vampires, which may loosely correlate with "bizarreness." However, they fail to capture the core intent of the query, namely over-the-top action movies that combine stylized violence, dark humor, and mainstream appeal. This example suggests that current semantic encoders still struggle to model complex intent beyond shallow lexical or thematic similarity.

Table 11: A failure case of the top-performing semantic encoder GritLM-7B on complex-query product search in the Reddit-Movies dataset.

must include at least one valid seeker turn; (2) recommendations must have at least 20 upvotes; (3) duplicate entries are removed using MD5 hashes of concatenated text and context; (4) samples must contain valid movie IDs matched with metadata. The dataset is split following the original partition, and each <*query, item*> pair represents a complex query and a recommended movie.

C.2 Item Metadata Processing

- **Amazon Reviews 2023.** For all Amazon-related datasets (Beauty, Games, Baby, Amazon-C4, and ESCI), we use product titles as item metadata. Although we conducted a small-scale experiment using concatenated titles and descriptions (Section 5.3), the inclusion of descriptions did not consistently improve performance. Therefore, we use only titles for simplicity and consistency.

- **Movielens-1M.** Movie titles and release years are used as metadata.

- **Yelp.** We concatenate each business's 'name', 'categories', 'stars', 'review_count', 'city', 'state', 'attributes', and 'hours' fields.

- **Book-Crossing.** We use book titles as metadata.

- **Reddit-Movie.** We use movie titles and release years as metadata.

C.3 Evaluation Metrics

For each dataset, we use the normalized discounted cumulative gain (NDCG) as the primary metric, with task-specific cutoffs: NDCG@10 for sequential recommendation, NDCG@20 for collaborative filtering, and NDCG@100 for product search.

To summarize results across datasets, we report three aggregated metrics in Table 6. First, to alleviate the bias in the number of datasets per task and the differences in metric scales, we follow MMTEB (Enevoldsen et al., 2025) and adopt the Borda Count (Colombo et al., 2022) as the primary ranking criterion. Second, for intuitive comparison, we also report the average across all datasets (Avg. Overall) and the average after first averaging within each (sub)task (Avg. per Task).

C.4 Evaluation Pipeline

For tasks requiring model training (sequential recommendation and collaborative filtering), we search over three learning rates ($\{1 \times 10^{-3}, 3 \times 10^{-4}, 1 \times 10^{-4}\}$) and select the checkpoint with the best validation performance for final testing.

For product search, evaluation is conducted in a zero-shot manner without training, following each LLM's recommended encoding procedure; for models that suggest a prompt when encoding queries, we use "*Given a user query describing a need, retrieve relevant entities (such as products, services, or media items) that satisfy the intent and constraints.*".

D Detailed Benchmarking Results

The results for each task are summarized in Table 6. Below, we provide detailed results for all evaluated LLMs across individual datasets. The results for sequential recommendation, collaborative filtering, and product search are presented in Table 8, Table 9, and Table 10, respectively.

E Discussion

Why is the correlation between BLaIR and MTEB results weak? We hypothesize that building a strong semantic encoder requires three key capabilities that may not be simultaneously emphasized in existing embedding benchmarks:

- The ability to accurately model semantic similarity, which is essential for tasks such as collaborative filtering and short-query product search.
- The ability to produce distinguishable and informative feature representations, which is important for tasks like sequential recommendation.
- Strong instruction-following and reasoning abilities, which are crucial for handling complex-query product search.

Why scaling semantic encoders yields limited gains on complex tasks? As observed in Section 5.2, in tasks with complex downstream architectures, such as sequential recommendation (a Transformer decoder), the scaling trend becomes less pronounced than tasks with simple downstream model architectures like collaborative filtering (a simple linear layer). Our current hypotheses are:

- *Differences in the nature of the tasks*: The representational capabilities required differ between tasks. Collaborative filtering is primarily about encoding similarity. A stronger semantic encoder can represent item similarities in a richer and more accurate manner, which directly translates into performance gains in collaborative filtering tasks as the encoder scales up. For example, Sheng et al. (2024) found that stronger semantic encoders yield representations whose distribution is more closely aligned with those learned from pure interaction data. Sequential recommendation is essentially a sequence modeling problem that requires memorizing and reproducing behavior patterns. To achieve this, item representations may require strong discriminative capability (akin to functioning as unique indices). Representations produced by even a smaller semantic encoder might already be sufficiently distinguishable for a complex downstream Transformer model. Consequently, scaling the semantic encoder does not significantly enhance discriminability from the perspective of the downstream model, leading to smaller gains compared to collaborative filtering. A similar observation was reported by Hou et al. (2022), who showed that treatments such as whitening enhance the discriminative power of item representations and improve sequential recommendation performance.
- *Sequentially dependent neural network systems and scaling behaviors*: Our setup functions as a two-stage system: a first-stage semantic encoder whose output is consumed by a second-stage model (a linear layer for collaborative filtering, or a Transformer decoder for sequential recommendation). We hypothesize that in such multi-stage systems, scaling the later-stage modules may diminish the impact of scaling earlier-stage modules. While there is limited systematic work on scaling laws in sequentially dependent systems, our findings may suggest that where we allocate additional parameters may matter as much as how many parameters we add.

F Case Study: Failure on Complex-Query Product Search

Table 11 presents a failure case from the Reddit-Movies dataset, which requires understanding a complex user query with multiple implicit preferences. This example highlights headroom for improvement in complex-query product search. In particular, even the best-performing semantic encoder on this dataset, GritLM-7B, only achieves an NDCG@100 of 0.0734 over a candidate set of 50k movies, suggesting that surface-level semantic matching is far from sufficient. Future improvements may require stronger intent modeling, explicit reasoning, or the incorporation of additional ranking signals such as popularity.