COSMO: A Large-Scale E-commerce Common Sense Knowledge Generation and Serving System at Amazon

Changlong Yu[1,2♣], Xin Liu[1,2♣], Jefferson Maia[1], Tianyu Cao[1], Yang Li[1], Yifan Gao[1], Yangqiu Song[1,2], Rahul Goutam[1], Haiyang Zhang[1], Bing Yin[1], Zheng Li[1♣*]
♣ Core Contributors. Corresponding author: Zheng Li.
[1]Amazon.com Inc, Palo Alto, USA [2]HKUST, Hong Kong SAR, China
{changlyu, xliucr, jdmaia, caoty, limyng, yifangao, gyangqiu }@amazon.com
{rgoutam, hhaiz, alexbyin, amzzhe}@amazon.com

ABSTRACT

Applications of large-scale knowledge graphs (KG) in the e-commerce platforms can improve shopping experience for their customers. While existing e-commerce KGs integrate a large volume of concepts or product attributes, they fail to discover user intentions, leaving the gap with how people think, behave, and interact with surrounding world. In this work, we present COSMO, a scalable system to mine user-centric commonsense knowledge from massive behaviors and construct industry-scale knowledge graphs to empower diverse online services. In particular, we describe a pipeline for collecting high-quality seed knowledge assertions that are distilled from large language models (LLMs) and further refined by critic classifiers trained over human-in-the-loop annotated data. Since those generations may not always align with human preferences and contain noises, we then describe how we adopt instruction tuning to finetune an efficient language model (COSMO-LM) for faithful e-commerce commonsense knowledge generation at scale. COSMO-LM effectively expands our knowledge graph to 18 major categories at Amazon, producing millions of high-quality knowledge with only 30k annotated instructions. Finally COSMO has been deployed in various Amazon search applications including search relevance, session-based recommendation and search navigation. Both offline and online A/B experiments demonstrate our proposed system achieves significant improvement. Furthermore, these experiments highlight the immense potential of commonsense knowledge extracted from instruction-finetuned large language models.

CCS CONCEPTS

• **Large Language Model, Knowledge Base**;

1 INTRODUCTION

Understanding users' intentions behind massive noisy behaviors in online e-commerce platforms can be beneficial for many downstream applications such as recommendations and product search, etc [9, 16]. From the view of cognitive science, intentions are mental states where humans can commit themselves to action, and behaviors result from intentions [27]. For example, "to attend a wedding party, we need to buy normal clothes" where the *intention*, i.e., "attend a wedding party" is used to rationalize and explain the user *behavior* i.e., "buy normal clothes". In online shopping scenarios, e-commerce platforms can be more intelligent and user-friendly to provide explainable recommendations and personalized search experiences if they can precisely capture users' intentions. However, such intentions are not explicitly expressed by human beings, which requires commonsense to understand and thus makes it challenging for machines to extract in a scalable way.

Figure 1: An example of mining implicit commonsense knowledge from e-commerce user behaviors.

Recently Yu et al. [45] propose to leverage a significant amount of knowledge implicitly stored in large language models like GPT3 [2] or OPT [48] and generate user intentions by "asking" the reason why users *purchase* or *co-purchase* products. One example is shown in Figure 1 and e-commerce commonsense knowledge can be discovered from user behaviors. Then human-in-the-loop annotations are involved in collecting the judgments and providing human feedback of automatic generations. Classifiers trained on small-scale annotated data are used to filter low-quality knowledge. Such distillation methods have been demonstrated effective in extracting high-precision commonsense knowledge at lower annotation cost [41, 45]. However, those methods generate knowledge candidates from language models that are not well aligned with human preferences. For example, we observe LLMs can generate generic intentions that are neither faithful nor helpful, like "customers bought them together because they like them" or "customers bought an

*The research for this project was conducted during Changlong and Xin's internship at Amazon. Prof. Yangqiu Song is a visiting academic scholar at Amazon.

Table 1: Comparison among existing commonsense knowledge graphs. 'Rel' represents relation types. Our new KG covers more nodes and edges in more domains compared to existing e-commerce related KGs for intention understanding.

KG	# Nodes	# Edges	# Rels	Source	Node Type	E-commerce	Intention	User Behavior
ConceptNet [30]	8M	21M	36	Crowdsource[1]	concept	✗	✓	✗
ATOMIC [25]	300K	870K	9	Crowdsource	daily situation, event	✗	✓	✗
AliCoCo [13, 14]	163K	813K	91	Extraction	concept	✓	✗	search logs
AliCG [47]	5M	13.5M	1	Extraction	concept, entity	✗	✗	search logs
FolkScope [45]	1.2M	12M	19	LLM Generation	product, intention	2 domains	✓	co-buy
COSMO (Ours)	6.3M	29M	15	LLM Generation	product, query, intention	18 domains	✓	co-buy&search-buy

Apple watch because it is a type of watch". The desired generation should be typical to explain e-commerce behaviors. Making language models better follow users' instructions becomes crucial to improve the helpfulness [1, 18], truthfulness [15] and transparency [26] of LLMs.

On the other hand, such distillation method still suffers from major challenges caused by scalability issues of industry-level data. First, [45] only explores co-purchasing intentions based on thousands of co-purchase item pairs within two categories. In the real production environment, millions of users produce complicated and noisy behaviors every day, which also potentially entail enormous and diverse intentions, such as *search-buy* behaviors. Thus, it is crucial to select representative user behaviors for diverse intention generations. Second, [45] performs fine-grained annotation by separately labeling plausibility and typicality scores. As we aim to fully support more scenarios in e-commerce, the annotation cost is significantly increasing with more categories and more user behavior types. Third, when applying FolkScope to downstream tasks, inference overhead might becomes the bottleneck since knowledge generation for new user behaviors has to go through the pipeline of LLM generation and classifier scoring. LLMs like `OPT-30b` used in FolkScope require huge computation cost and are not feasible for online serving.

In this work, motivated by recent advancements in instruction-following language models [4, 24, 38, 40], we directly align language models with human feedback via instruction tuning for e-commerce commonsense knowledge extraction. Instruction-finetuned language models over a large collection of datasets have demonstrated remarkable zero-shot abilities [40]. How to collect high-quality and diverse instruction data becomes important and challenging. Starting from the annotation data across two domains of *co-purchasing* behavior in [45], we scale up the data collection in terms of intention *knowledge resources* (i.e., user behaviors), *product domains*, and *relation types* shown in Figure 4. For user behaviors, we also adopt industry-scale *query-item* interactions to generate ambiguous and evolving intentions. Different from straightforward intentions behind co-purchasing behaviors [45], query intentions can help reduce the semantic gaps between what a user truly needs and how the product information is presented in the e-commerce system. Generated intentions can help refine the broad query to specific users' needs and improve the query understanding abilities. In addition, we sample millions of two user behavior data among 18 popular domains (*product categories*) for knowledge candidate generation (§3.2). Before human labeling, we create a branch of heuristic rules to filter out low-quality knowledge and design careful sampling strategies for annotated data selection (§3.3). Following Yu et al. [45], we collect two evaluation metrics named *plausibility* and *typicality* as human feedback (§3.3.2). To fuse language models with human judgments, we select *typical* knowledge examples as the demonstrations of desired model outputs for the commonsense generation task while annotated labels as the desired model outputs for label prediction tasks such as typicality prediction etc. (§3.4). The resulting LMs are capable of generating typical knowledge and judging knowledge quality as well. Compared with vanilla LLMs, our instruction-finetuned LM can significantly reduce inference time and support extensive applications at scale. We successfully deploy COSMO in various Amazon search applications and achieve significant offline performance improvement and online revenue increases.

The contributions of our work can be summarized as follows.

- We are the first industry-scale knowledge system that adopts large language models to construct high-quality knowledge graphs and serve online applications.
- We adopt instruction tuning for effective e-commerce commonsense knowledge generation to better align with human preferences.
- We scale up e-commerce intention knowledge to millions of user behaviors and achieve high-quality instruction data generation with fewer annotation efforts.
- We apply generated intention knowledge to three real-world e-commerce tasks, and promising experimental results show great potential for more e-commerce scenarios.

2 RELATED WORK

E-commerce Commonsense Knowledge. Existing e-commerce knowledge graphs [5, 8, 13, 14, 46, 47] are mainly based on factual knowledge concerning product attributes such as *isA* or *authorOf* relations, and are not well connected with commonsense knowledge about user intentions like "apple product fans" or "attend weddings" etc. There is still a gap between collecting factual knowledge about products and modeling users' purchasing intention, which we list the detailed comparison in Table 1. In contrast, Yu et al. [45] proposed a framework named `FolkScope` to distill intention knowledge from massive user behaviors by prompting large language models. Instead of directly authoring knowledge assertions, human beings only label a small number of automatic generations as high-level supervision. Then classifiers trained on labeled data are used to filter out low-quality generations. Although `FolkScope` achieves high-precision extraction with low annotation cost, it covers limited domains and ignores abundant types of user interaction data

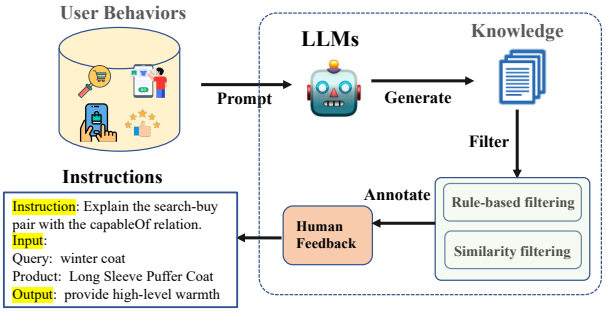


Figure 2: Overall framework of generating high-quality instruction data from massive user behaviors and large language models.

that entail complex intention knowledge in e-commerce scenarios. To improve the generalization of e-commerce commonsense extraction, we extend `FolkScope`, including scaling up to 18 popular domains and introducing millions of search query behavior data. Scaling up also presents challenges in terms of inference efficiency when distilling knowledge from LLMs. We solve them by effective finetuning.

Instruction-followed Language Models. Language models pretrained on web-scale corpus often generate unfaithful, biased, or unhelpful contexts. This is because the training objective of most vanilla LMs, i.e., predicting the next token, is not aligned with human preferences. Recently a series of works demonstrate that finetuning a language model with natural language instructions can teach LMs to have desired model behaviors [18, 40]. Instruction-finetuned LMs have substantially improved their zero-shot and few-shot performance on unseen tasks [4, 24, 38]. The quality and diversity of instruction data have large impacts on the instruction-following abilities of LMs. As collecting human-written instructions is time-consuming and costly, Wang et al. [37] proposed *self-instruct* to iteratively generate instructions and their outputs from GPT3 [2] based on a small seed set of tasks. Followup works [3, 19, 32] directly use machine-generated instruction-following data from ChatGPT or GPT4 for LLM finetuning. However, they focus more on general-purpose language models and instruction-finetuned LMs on specific domains such as *e-commerce* remain unexplored. Our work aims at efficient e-commerce instruction data collection and finetuning LLMs to generate helpful and typical commonsense knowledge. None of the above KGs are related to products or purchasing intention. We are the first to propose a effective KG construction pipeline from LLMs and massive user intentional behaviors. Our pipeline can be efficient for online serving of industry-scale applications.

3 PROPOSED FRAMEWORK

3.1 Preliminary

In this section, we present the formal definition for terms in Figure 2 and the overview of offline COSMO knowledge generation pipeline.

User Behaviors. Millions of users interact with online e-commerce platforms every day and produce massive behavior logs. E-commerce systems mine the intentions behind those behaviors to provide a better online shopping experience. We choose two typical user behaviors with strong potential intentions, i.e., *search-buy* and *co-buy*. Formally, we define the *search-buy* behavior as the query-product pair (q, p) that customers click the query q and finally purchase the product p within short sessions. Similarly, we use the co-purchased product pair (p_1, p_2) to represent the *co-buy* behavior. Each product p can be categorized into one major domain $d \in \mathcal{D}$ (all domains are shown in Table 3 and Figure 4).

Commonsense Knowledge. Following [45], we leverage relation-aware prompts for LLMs to explain the user behavior h as knowledge candidates, which we represent the knowledge as the triple (h, r, t) where r and t represent relation and tail respectively. For example, "customers bought camera case and screen protector glass together because they are capable of providing protection for camera", "provide protection for camera" is the tail under the relation *capableOf*.

Different from previous work [45] aligning commonsense relations from ConceptNet [30] for thousands of data, we can not simply adopt for millions of user behavior pairs due to computation constraints. Hence we propose data-driven relation discovery from large-scale generations to satisfy e-commerce scenarios. The basic idea is to start from four seed relations (i.e., *usedFor*, *capableOf*, *isA*, *cause*) that tend to generate diverse/high-quality knowledge according to the previous work [45] and mine the frequent predicate patterns to manually summarize the relations. The most common pattern is "the product is capable of being used [Prep]", where `[Prep]` means prepositions. Generations with different prepositions represent different tail types, which can be further canonicalized. By doing so, we can also make generated knowledge structured. We summarize our mined knowledge relation types and corresponding tail types as well as examples in Table 2. Either relation type or tail type is more e-commerce specific and strongly related to daily scenarios, which might require commonsense.

Instruction Data. We denote $\{I_t\}$ as a set of instructions, which each defines a task t in natural language. One example in Fig 2 can be "generate explanations for the search-buy behavior in the domain d using the *capableOf* relation". Each task includes l_t input-output pair instances. For the commonsense generation task, the input can be a user behavior pair (p_1, p_2) or (q, p), and the output is the typical knowledge tail t. Note the quality of knowledge (h, r, t) can be measured by *plausibility* and *typicality* scores labeled by human annotators [20, 45]. For the sake of usability and helpfulness, we select knowledge with high-typicality scores as desired model outputs. To further improve the e-commerce aware abilities of instruction-finetuned models, we also add several auxiliary tasks and train a language model for knowledge generalization and online serving as well (more details in §3.4)

3.2 Knowledge Generation

In this section, we first describe how we efficient sample representative user behaviors as inputs of LLMs. Then we introduce the question-answering based prompts to harvest large-scale knowledge candidates from general LLMs.

3.2.1 User Behavior Sampling. Millions of users interact with online e-commerce platforms everyday and produce massive behavior

Table 2: Mined e-commerce commonsense relations for the COSMO KG.

Relation Type	Tail Type	Example
USED_FOR_FUNC	Function / Usage	dry face
USED_FOR_EVE	Event / Activity	walk the dog
USED_FOR_AUD	Audience	daycare worker
CAPABLE_OF	Function / Usage	hold snacks
USED_TO	Function / Usage	build a fence
USED_AS	Concept / Product Type	smart watch
IS_A	Concept / Product Type	normal suit
USED_ON	Time / Season / Event	late winter
USED_IN_LOC	Location / Facility	bedroom
USED_IN_BODY	Body Part	sensitive skin
USED_WITH	Complementary	surface cover
USED_BY	Audience	cat owner
XINTERSTED_IN	Interest	herbal medicine
XIS_A	Audience	pregnant women
XWANT	Activity	play tennis

logs. E-commerce systems mine the intentions behind those behaviors to provide better online shopping experience.

In our work, we choose two typical user-behaviors with strong potential intentions, i.e., *search-buy* and *co-buy* as described in § 3.1. Huge-volume behaviors contain noises or are non intentional random ones. In order to generate diverse and high-quality knowledge, we conduct fine-grained sampling, which starts from product sampling followed by behavior pair sampling. For the product sampling, we cover most common popular categories (also known as *browse nodes*[2]) at Amazon and select top-tier products that have relatively larger behavior interactions. Besides category labels, we also adopt *product type* labels for sampling that define more than a thousand classes and describe what the products essentially are, such as *umbrella*, *chair* etc.

For the *co-buy* pair sampling, each co-buy edge should cover at least one from the selected product set and we cross-check with the *product type* of sampled pairs to remove random co-purchases and avoid duplicated sampling from the abstract level. Also some heuristic rules are applied such as the products co-purchased by different types of products are likely to be randomly selected. For the *search-buy* pair sampling, we empirically set thresholds for both *purchase rate* and *click rate* to sample queries as well as purchased products. One crucial consideration is the *specificity* of query, which indicates whether the query is a broad or specific one. As our goal is to make up the semantic gap between the search query and the product, generating knowledge for the broad or ambiguous query are of more values to narrow down clear needs. So we use one in-house service from Amazon Search to compute the *specificity* score of the query and sample broad queries associated with purchased products. For most search queries with high engagement, search engines can understand their intentions well. We also sample queries with lower engagement and less *purchase rate* to directly probe knowledge from LLMs themselves. To take all the above strategies into consideration, we finally sample several millions of behavior pairs. The statistics of sampled behavior pairs are shown in Table 3 and there exist 1.40 million *product type* pairs among 3.14 million co-purchased product pairs, which also demonstrates the diversity of our sampling.

3.2.2 QA-Prompted Generation. LLMs have been shown to encode a significant amount of knowledge in their parameters. Specific-designed prompts can enable autoregressive LLMs to continue generation on condition of verbalized prompts. For example, given a purchase behavior "A customer bought an iPhone because it has", LLMs can generate the intention knowledge related to function or property of "iPhone". In our work, we find that LLMs are more skillful at answering contextualized questions given a well-described scenarios or task instructions, which align with specific user behaviors. So we verbalize the user behaviors by providing a Question-Answering (QA) context. Take the following *search-buy* prompt as example in Figure 3.

> Task: Please provide typical explanation for the following search-purchase behavior and complete the answer.
> Search Query: {Query}
> Product: {Product Title}
> Question: what is the product capable of, which exactly match the intention of the search query?
> Answer: the query means customers want the product that is capable of
> 1.

Figure 3: Prompts used for generating knowledge candidates.

Adding the number character "1" at the end is a useful prompt engineering trick to generate a list of knowledge candidates. we first provide a task description like "The following search query caused the following product purchases", then follow the specific query and product information. For general LLMs, we append one question and partial answer so that LLMs can follow the given instructions in convenience of parsing generation. In our work, we use both `OPT175b` and `OPT30b` [48] hosted on 16 A100 GPUs to conduct generation inference[3]. Some generation examples for each domain are shown in Table 9 of the Appendix.

3.3 Knowledge Refinement

Though well-designed knowledge generation and relation-specific parsing, vanilla LLMs can generate generic or unfaithful knowledge. To encourage diversity and helpfulness, we use the following steps to filter the generations.

[2] https://www.browsenodes.com/

[3] We do not choose to query powerful ChatGPT or GPT4 APIs due to private data access and privacy constraints.

Table 3: Statistics of COSMO knowledge graph including the sampled user behavior pairs, annotated knowledge candidates, and remaining edges after knowledge refinement.

	Co-buy			Search-buy		
Category	# Behavior Pairs	# Annotations	# Edges	# Behavior Pairs	# Annotations	# Edges
Clothing, Shoes & Jewelry	233,989	1303	2,147,605	176,018	597	887,130
Sports & Outdoors	251,713	1302	2,140,491	126,130	970	556,233
Home & Kitchen	426,070	1991	3,380,502	225,377	3798	1,054,764
Patio, Lawn & Garden	117,871	542	908,158	56,754	263	280,932
Tools & Home Improvement	258,480	1184	1,988,346	122,613	585	629,004
Musical Instruments	24,206	84	174,238	9,385	24	33,786
Industrial & Scientific	385,990	1820	3,002,352	177,400	1317	814,266
Automotive	166,234	782	1,330,580	55,201	456	258,340
Electronics	178,938	777	1,316,937	119,764	768	549,716
Baby Products	111,204	430	721,727	30,156	38	135,702
Arts, Crafts & Sewing	13,1131	616	1,095,531	62,135	232	274,015
Health & Household	233,945	1198	1,906,447	215,349	67	930,307
Toys & Games	148,455	646	1,165,692	73,512	536	291,107
Video Games	16,436	60	106,449	10,306	30	29,681
Grocery & Gourmet Food	99,660	504	775,016	116,765	2123	577,986
Office Products	136,519	650	1,086,735	79,470	2063	364,767
Pet Supplies	43,541	206	302,839	51,807	1122	219,143
Others	182,738	905	1,351,257	160,189	11	648,765
Total	3,147,120	15,000	24,900,902	1,868,331	15,000	5,093,795

3.3.1 Coarse-grained Filtering. In this step, we aim at filtering incomplete generations with the help of linguistic analysis and general knowledge that apply for any behavior.

Rule-based Filtering. We first use the sentence segmentation tool from `nltk` to extract the first sentence from generation. Then we calculate the *perplexity* score based on the `GPT-2` language model and tune the threshold to remove incomplete sentences. We also directly filter the generations that are exactly the same as *query*, *product type* or *product title* (or edit distance less than the threshold). For the general knowledge like "used for the same reason", or "used with clothes", we identity those cases by combining frequency and entropy since they co-occur with many products or queries rather than specific ones.

Similarity Filtering. To handle the semantic-similar cases that can not easily be handled in the last step, we use the in-house language model, which was pretrained on the e-commerce corpus including *query*, *product information* etc, to obtain the embeddings for generate knowledge tails, query and product themselves. The similarity between the knowledge embedding and the context embedding (the original query or product embedding) is computed by their cosine similarity:

$$d(k, c) = \cos(\mathbf{E}(k), \mathbf{E}(c)). \tag{1}$$

We find that filtered generations are essentially paraphrases of original user behavior contexts with syntactic transformations. By two coarse-grained filtering steps, we are able to remove quite a large amount of noise and keep typical knowledge as much as possible.

3.3.2 Human-in-the-loop Annotation. The annotation step aims at providing human feedback for knowledge candidates and collecting diverse instruction data. The biggest challenge is still the balance between huge-volume knowledge candidates and cost. We expect models trained over annotated data can generalize well among multiple categories shown in Table 3. Uniform sampling might hurt the prediction performance on long-tail knowledge. Instead we combine the log of knowledge frequency and popularity of product or query for re-weighting:

$$w_{(q,p),t} = \frac{\log(f(t))}{\text{pop(q)} \times \text{pop(p)}}, \tag{2}$$

where $f(t)$ is the frequency of generated knowledge and the function of popularity is defined by the degree of query in the query-product interaction graph or the degree of product in the product co-buy graph. The more popular the product is, the more likely the generated knowledge is common. For both two user behaviors, we sample 15 thousand knowledge candidates for annotation and the distribution is also shown in Table 3.

Due to data privacy issues, we employ professional data annotation vendor company to conduct the high-quality annotation followed by strict and careful internal auditing process. Previous work [45] measures the quality of generated knowledge by two-step annotation i.e., *plausibility* (how the knowledge is plausible) and *typicality* (how representative the knowledge is regarding the typical shopping behavior). One example is that more typical intention why customers bought apple watches is that they are intelligent watches instead of being used for telling the time. To reduce the cognitive burden of annotators and the potential disagreement rate of commonsense, we decompose the two measurements' judgments into five clear questions: 1). Is the explanation a *complete* sentence? 2). Is the explanation *relevant*? 3). Is the explanation *informative*? 4).

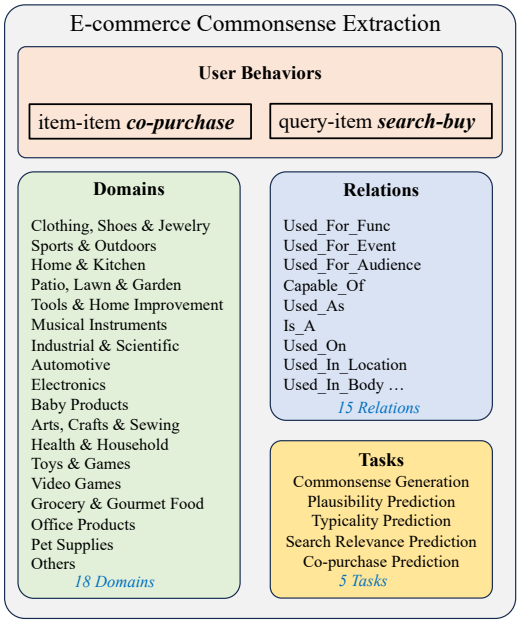


Figure 4: Illustration of finetuning COSMO-LM to generate e-commonsense knowledge from two typical user behaviors. We scale up product domains, relation types and tasks.

Is the explanation *plausible*? 5). Is the explanation *typical*?, where each question is labeled as yes/no/not sure by two different annotators and finally checked by a third person if disagreement is found[4]. Pilot study over 2000 example annotation shows that the pipeline significantly reduced disagreement rate. For the quality of annotated data, we randomly sample 5% annotation for internal auditing and the accuracy can reach more than 90%. We then build a classification model using this data to score all the knowledge candidates after coarse-grained filtering. We fine-tuned both `DeBERTa-large` [6] and our in-house language model to populate the human judgements to the whole knowledge candidates whose *plausibility* score is above 0.5 are left. After the process of knowledge refinement, we obtain high-quality e-commerce knowledge with relatively low cost and the statistics are also shown in Table 3.

3.4 Instruction-tuned COSMO Language Model

After collecting human judgments on 30k diverse knowledge samples, we can create large-scale instruction data based on annotated data. The annotation results are shown in Table 4. We can observe that more than one-third *search-buy* generations are typical and can directly serve as instruction data. But the typical ratio for *co-buy* is notably low since LLMs mostly generate intention knowledge for one of the co-purchased products rather than considering their common reasons, making generations implausible. We expect fine-tuned language models to have desired model behaviors. Apart from generating typical knowledge, we enable LMs to have abilities

[4]The instructions of each question are detailed in Appendix B and the screenshot of annotation interfaces is shown in Figure 11.

Table 4: The plausibility and typicality ratios of annotated data for two user behaviors.

	Plausibility	Typicality
SEARCH-BUY	44.3%	35.0%
CO-BUY	14.5%	9.0%

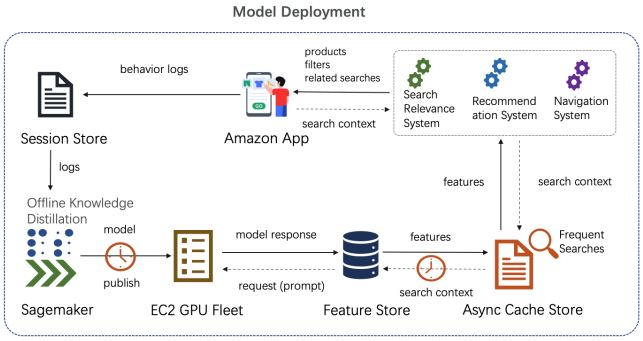


Figure 5: Illustration of COSMO-LM deployment, featuring the Asynchronous Cache Store and Feature Store as central components. It depicts the efficient processing of user queries and dynamic daily updates, crucial for meeting Amazon's search latency requirements.

of plausibility and typicality prediction, in which all the annotations are converted to instruction data for the tasks.

Considering non-negligible noises of user-behavior data, our fine-grained annotations in §3.3.2 have identified irrelevance *query-product* pairs or random *cobuy* pairs. We also consider adding *co-purchase prediction* and *search-relevance prediction* into the fine-tuned tasks. So far, we collect instruction data covering 18 product domains, 15 relation types, and 5 different types of tasks. To make the model robust to different formats, we design different templates to verbalize the instructions and input-output pairs. For example, we add prefixes of "search query", "user input" or "user searched:" etc. We finetune the LLaMA 7b and 13b models [33, 34], the widely-used open foundation models with our collected instruction data.

3.5 Online Deployment

The deployment centers around an efficient feature store and asynchronous cache store, ensuring streamlined processing and cost-effective management of customer queries and model responses.

3.5.1 Deployment Strategy. **Deployment Management**: SageMaker [11][5] is used to refresh COSMO-LM model, facilitating dynamic ingestion of customer behavior session logs and efficient model updates through robust automation. **Feature Store Integration**: This store is essential for transferring model responses to structured features, making them actionable for downstream applications. It handles features like product key-value pairs, semantic subcategory representations, and strong intent detection. **Asynchronous Cache Store**: Employed to manage frequent searches

[5]Amazon machine learning model services https://aws.amazon.com/pm/sagemaker/

and adapt to daily traffic patterns, this store efficiently captures user queries through a two-layered caching strategy, combining pre-loaded yearly frequent searches and batch-processed daily requests.

3.5.2 Operational Flow. We list key processes showed in Figure 5:

- **Model Deployment**: COSMO-LM is deployed on SageMaker for processing user behavior session logs and dynamic model updates.
- **Request Handling**: Initial query checks against the Asynchronous Cache Store quickly retrieve responses for frequent queries or forward others for batch processing.
- **Batch Processing and Cache Update**: The Feature Store formats language model responses into structured insights, updating the cache for future queries.
- **Communication with Downstream Applications**: Structured data from the cache enhances various downstream applications, providing enriched features for improved user interaction.
- **Feedback Loop**: Continuous model refinement is achieved by feeding back user interactions into COSMO-LM, ensuring up-to-date responsiveness to evolving user behaviors.

3.5.3 Impact and Limitations. The deployment of COSMO-LM, utilizing the Asynchronous Cache Store and Feature Store strategy, effectively meets Amazon's restricted search latency requirements while maintaining storage costs comparable to real-time serving for the majority of traffic. This approach significantly enhances our ability to manage online requests swiftly and economically. To acknowledge, even though we refresh our model daily, we are limited in processing real-time information, such as flash sales. These time-sensitive events, often fluctuating within a short span, pose a challenge to our current system's ability to rapidly assimilate and reflect such immediate changes. This limitation underscores the need for further development to enhance our system's agility in responding to the fast-paced dynamics of e-commerce activities.

4 EVALUATIONS AND APPLICATIONS

In this section, we adopt instruction-tuned COSMO language models to generate e-commerce commonsense knowledge for downstream applications, i.e., search relevance, session-based recommendation and search navigation. We conduct extensive offline and online evaluation experiments to demonstrate the effectiveness of our proposed framework and deployed system.

4.1 Search Relevance

Determining relevance scores between the search query and documents lies the core of information retrieval, which serves as crucial components for search engines [29]. A major challenge in e-commerce product search is the semantic gap between queries and product catalogs [10, 17]. Some of them require abundant commonsense knowledge to bridge them together. For example, the query "winter clothes" often implicates the users want clothes to keep warm. Hence we augment search relevance prediction with COSMO knowledge explaining *search-buy* behaviors.

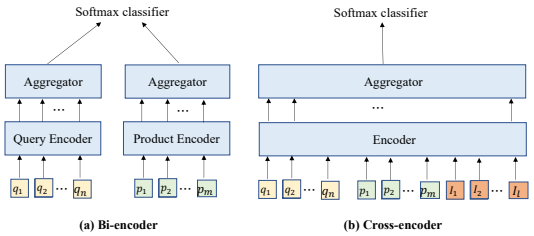


Figure 6: Illustration of Search Relevance Models.

Table 5: Statistics of ESCI evaluation datasets of different locales (markets).

	KDD Cup	US	CA	UK	IN
# Training Pairs	1,393,063	1,148,528	220,114	462,560	1,480,116
# Test Pairs	425,762	383,695	72,500	155,138	495,078
# Exact Pairs	1,247,558	1,104,417	245,796	455,947	1,352,128
# Unique Queries	97,345	57,971	9,537	32,162	42,884
# Unique Products	1,215,851	803,363	136,398	427,572	456,407

Formally, given a query $Q = \{q_1, q_2, ..., q_n\}$ and a list of retrieved products D where $P \in D$, either ranking or classification tasks require the relevance score of each query-product pair $\{Q, P\}$ [21]. In real e-commerce systems, each product is accompanied by side information, e.g., product title, descriptions and attributes. To be simple, we concatenate them into one single text span $P = \{p_1, p_2, ..., p_m\}$. As aforementioned that there remains semantic gaps between user intentions in the query Q and product information P, we leverage COSMO-LM to generate commonsense knowledge $G = \{g_1, g_2, ..., g_l\}$ behind the query-product pairs and explicitly enhance their connections.

4.1.1 Experiment Setup. We adopt open-released Amazon shopping query datasets[6] from KDD Cup 2022. Following the settings of Task 2, the problem of measuring search relevance is formulated as a four-class classification problem: to distinguish a given product as an *Exact*, *Substitute*, *Complement*, or *Irrelevant* match for a user's query. In order to verify the generalization of our approach, we also collect similar datasets from our online system to accommodate product varieties and languages habits across different markets, i.e., United States (US), Canada (CA), United Kingdom (UK), and India (IN). Dataset statistics are reported in Table 5. Considering the class imbalance distribution, we report Macro F1 and Micro F1 but focus more on the former one.

4.1.2 Baselines. We consider two representative architectures as baselines shown in Figure 6:

- **Bi-encoder** [22, 28], also known as two-tower models, takes the concatenation of the query representation and product title representation as the input of a multi-layer perceptron to predict the relevance label.
- **Cross-encoder** [42] feeds all relevant features (e.g., query, product title, description, etc) into the unified encoder and leverage joint representations to make predictions.

[6] https://github.com/amazon-science/esci-data

Table 6: Experimental Result of Public ESCI English subset.

Method	Fixed Encoder		Trainable Encoder	
	Macro F1	Micro F1	Macro F1	Micro F1
Bi-encoder	25.52	65.49	47.96	70.23
Cross-encoder [42]	28.44	66.84	57.49	74.23
Cross-encoder w/ Intent	**45.52**	**86.40**	**73.48**	**90.78**
Δ	*60.06%*	*29.26%*	*27.81%*	*22.30%*

Cross-encoder models generally outperform bi-encoder counterparts due to extra attention interactions. Hence we augment cross-encoder models with our generated knowledge features, i.e., concatenate $[Q, P, G]$ as inputs. We follow [42] to use strong *deberta-v3-large*[7] as the base model and consider both fixed and tuned settings for encoders.

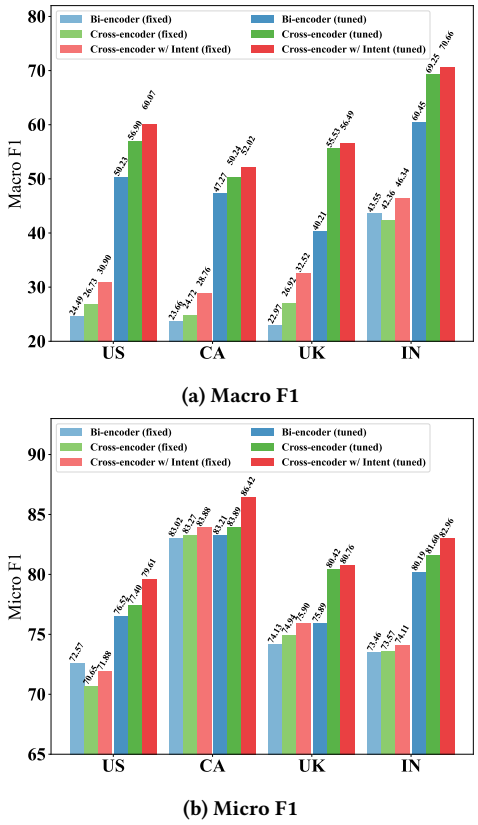
4.1.3 Public Dataset Results. Table 6 shows that knowledge generated from COSMO-LM, which captures implicit e-commerce commonsense, can significantly boost the performance of query-product semantic relevance. When the encoder is fixed, there is no huge difference between two architectures. But augmented intention knowledge boosts the performance around 60% on Marco F1 and 30% on Micro F1. We can still observe the performance enhancement around 25% when the parameters of encoders are updated. Finally, generated knowledge helps Cross-encoder achieve 73.48% Macro F1 and 90.78% Micro F1, which even surpasses the top-1 ensemble model of KDD Cup leaderboard [42].

4.1.4 Private Dataset Results. To further validate the effectiveness of our approach on multi-locale scenarios, we conduct similar experiments on a large-scale private dataset. The product distribution and query language habits might have significant differences across different locales (markets). We expect our generated knowledge can provide high-quality features or signals for search relevance systems, and generalize to more complex scenarios. From Figure 7a and Figure 7b, we can conclude the following observations: 1). Our COSMO-LM can always help strengthen cross-encoder performance even with limited annotations, which is in line with results of the public dataset in §4.1.3. 2). Intention-enhanced cross-encoder models can significantly outperform baseline methods for all locales whenever the encoder is fixed or tuned. In the online deployment environment, generated knowledge as well as other features stored in the feature store are integrated to make final predictions shown in Figure 5. In order to improve serving efficiency, we pre-cache features for frequent search queries.

4.2 Session-based Recommendation

Recommendation systems have become one of most crucial components in the e-commerce platform for customers to choose from massive and rapidly increasing products. Sessions associated with multiple *user-item* interactions in a period of time can better capture user preferences and intents besides user profiles [36]. Session-based recommendations typically predict next click or purchased

[7] https://huggingface.co/microsoft/deberta-v3-large



(a) Macro F1

(b) Micro F1

Figure 7: Comparison results on private ESCI datasets of four different locales.

item from the product item set $V = \{v_1, v_2, ..., v_m\}$ given an anonymous behavior sequence $S = \{v_1^s, v_2^s, ..., v_l^s\}$ in the chronological order where l is the length of session S. Sequential neural networks, such as RNN [7], transformers [31], are employed to capture user dynamic preferences within sessions. Further item sequences can be organized as session graphs $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s)$ that model complex pair-wise interactions of adjacent items using graph neural network. The relation of edge (v_i, v_j) can be defined by the interaction direction, i.e., *in-edge* or *out-edge* [39]. Both sequential or graph-based methods only learn item embeddings for v_i but ignore side information of products, like product titles, product attributes, and interaction patterns. Among them, search queries associated with clicked/purchased behaviors are helpful to better capture user intentions and evolving preference changes. Hence we improve session-based recommendation by auxiliary user search keyword sequences $K = \{k_1^s, k_2^s, ..., k_l^s\}$ and our generated knowledge for each *search-product* pair (v_i^s, k_i^s).

4.2.1 Experimental Setup. We collect and filter one-week session data from our log system that falls into the categoires of *clothing* and *electronics*. Each session is limited within 20 minutes, which contains highly-frequent items in the same domain and ends with successful purchases. For training/test splitting, sessions in the first five days are used as training, while the sixth and the last

Table 7: Statistics of Session-based recommendation datasets of two categories. "Avg. Sess. L." stands for average session length. "Avg. Q. L." is the average query length. "Avg Uniq. Q. L." stands for average unique query length.

	clothing			electronics		
	Train	Dev	Test	Train	Dev	Test
# Sessions	1.32M	0.24M	0.23M	3.13M	0.59M	0.58M
Avg. Sess. L.	8.79	8.78	8.70	12.27	12.17	12.22
Avg. Q. L.	8.32	8.29	8.22	11.68	11.61	11.61
Avg Uniq. Q. L.	1.36	1.37	1.36	2.47	2.48	2.48

day are employed as validation and test data. Dataset statistics are detailed in Table 7. Sessions of the *electronics* domain have longer unique query sequences than *clothing*. It indicates that users might revise their original search keywords and modeling dynamics of user query can help precisely predict user behaviors. We formulate session-based recommendations as a ranking problem as previous work [36] and employ the commonly-used metrics in our experiments, i.e., Hits@10, NDCG@10, and MRR@10,

4.2.2 Baselines. We compare with competitive sequential models and graph-based models as baselines:

- **FPMC** [23] formulates the representation of session via Markov-chain based methods.
- **GRU4Rec** [7] leverages Gated Recurrent Unit (GRU) to simulate the Markov Decision Process but has a better generalization.
- **STAMP** [12] applies attention on the last item and previous histories to represent users' short-term interests.
- **CSRM [35]** combines an inner memory encoder and external memory to capture session correlations.
- **SR-GNN** [43] is the first to apply graph neural network (GNN) to the SBR task, which transforms the session sequence into a direct unweighted graph to learn item and transition representations.
- **GC-SAN** [44] extends SR-GNN by self-attention over the whole graph after graph convolution to obtain the global representation.
- **GCE-GNN** [39] aggregates two levels of item embeddings from session graphs and global graphs with soft attention.

4.2.3 COSMO-GNN. Preliminary experiments demonstrate that `GCE-GNN` can achieve strong performance on various session-based recommendation datasets and learn better item embeddings with two-level GNNs. Therefore, we extend GCE-GNN with search query related knowledge generated from COSMO-LM, and jointly optimize GNN for search intention-aware recommendation. We name our propose approach as COSMO-GNN. Formally for the time step t in the session S, the user searches the query k_t^s and have interaction with the item v_t^s. The item embedding obtained from GCE-GNN is denoted as $\mathbf{h}_t^s$. Then COSMO-LM is used to generate intention knowledge explaining the behavior with query-product pair (v_t^s, k_t^s). We leverage the same LM to vectorize generated knowledge and obtain session knowledge embedding $\mathbf{g}_t^s$. To align the knowledge space with GNN feature space, a two-layer perceptron is used to transform knowledge representation $\mathbf{g}_t^s$ to $\hat{\mathbf{g}}_t^s$. The final representation for each step is the concatenation of GNN item embedding and knowledge embedding, i.e., $[\mathbf{h}_t^s, \hat{\mathbf{g}}_t^s]$. Following [39], the session representation can be obtained via average polling over all steps' representations.

Table 8: Experimental Results of Session-based Recommendations.

Method	clothing			electronics		
	Hits@10	NDCG@10	MRR@10	Hits@10	NDCG@10	MRR@10
FPMC	62.16	45.07	39.60	21.79	16.01	14.18
GRU4Rec	83.20	63.37	56.94	49.53	33.99	29.06
STAMP	81.34	61.32	54.86	56.96	38.74	32.92
CSRM	82.31	65.59	60.25	61.66	46.63	41.83
SRGNN	85.82	69.68	64.45	67.83	55.23	51.22
GC-SAN	84.43	68.96	63.93	66.88	55.87	**52.34**
GCE-GNN	86.67	69.35	63.79	70.13	55.17	50.37
COSMO-GNN	**90.18**	**72.30**	**67.08**	**74.21**	**56.26**	50.67
Δ	*4.05%*	*3.76%*	*4.08%*	*5.82%*	*0.70%*	*-3.19%*

4.2.4 Results. Experimental results are shown in Table 8. We can observe: 1). Our proposed COSMO-GNN significantly outperforms all the competitive baselines on Hits@10 and NDCG@10 for two domains, and compete almost all baselines with MRR@10. 2). COSMO-GNN achieves slightly more improvement (5.82% v.s. 4.05% Hits@10) on the session data that has more complex and diverse search sequences. As shown in Table 7, more unique search queries are involved in the session of *electronics* than *clothing* (2.47 v.s. 1.36). The reason might be that user intentions for *clothing* are much easier to describe, but it requires more background knowledge for revisions to reach what users really need. More investigations like how COSMO reduces query rewrites are left for future work.

4.3 Search Navigation

Besides aforementioned traditional e-commerce scenarios, COSMO can also revolutionizes search navigation, moving away from traditional product-centric taxonomies towards a customer-focused approach. This shift enhances the shopping experience, aligning it more closely with customer intents and behaviors, and bridging the gap between product classification and customer language by dynamically providing taxonomy with customer query concepts. Specifically COSMO intention knowledge can be further organized into hierarchies shown in Figure 8 that expand coarse-grained ones (*camping*) to fine-grained ones (*winter camping*), and intention concepts are further linked to product concepts such as *winter boots*.

4.3.1 Multi-Turn Navigation. COSMO distinguishes itself with a multi-layered and dynamic navigation system (Figure 9):

(1) **Broad Conception Interpretation:** It begins by tackling broad queries using advanced analytics and customer behavior insights, covering a wide range of user intents without explicit domain knowledge.
(2) **Product Type and Subtype Discovery:** Subsequently, COSMO assists users in identifying specific product types and subtypes, adept at handling both direct and abstract product queries.

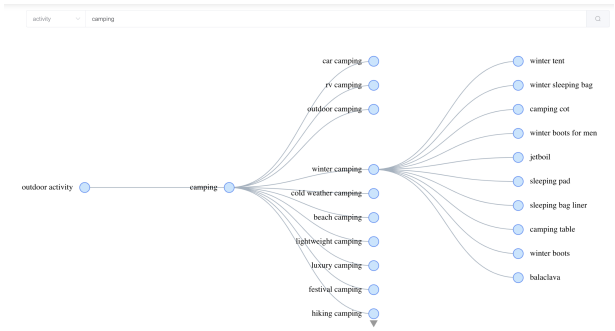


Figure 8: An illustration of hierarchical organization of COSMO tail knowledge.

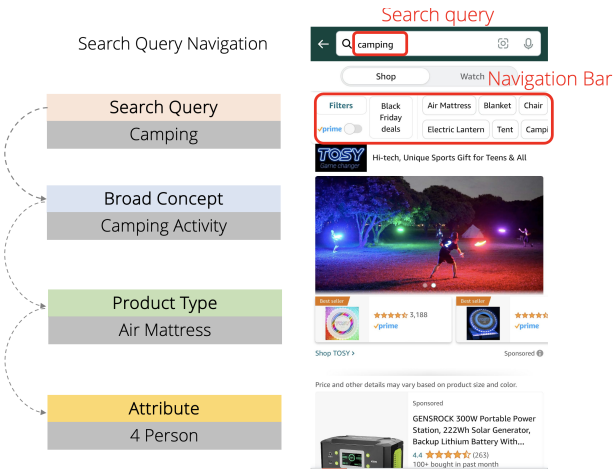


Figure 9: Search Navigation Experience using COSMO

(3) **Attribute-Based Refinement:** The final layer aids in fine-tuning search results, allowing users to filter based on specific attributes and aligning results with individual preferences.

Central to COSMO's functionality is the **Multi-Turn Navigation**. Here, COSMO excels in providing multiple rounds of search refinements through continuous recommendations. For example, a search for 'camping' might lead to a selection like 'air mattress', which then refines to 'camping air mattress'. COSMO would then offer various types of camping air mattresses tailored to different needs such as *lakeside camping*, *mountain camping*, or *4-person camping*. This multi-turn navigation allows for deeper and more precise refinements, mirroring a natural discovery process and significantly enhancing the user's search experience.

4.3.2 Online Experiments. The integration of COSMO into our online search navigation system has led to significant business improvements, underscoring the power and potential of COSMO-LM based applications. This conclusion is drawn from meticulously conducted Amazon online A/B tests carried out over several months in total, targeting approximately 10% of Amazon's U.S. traffic. These well-structured tests revealed a notable 0.7% relative increase in

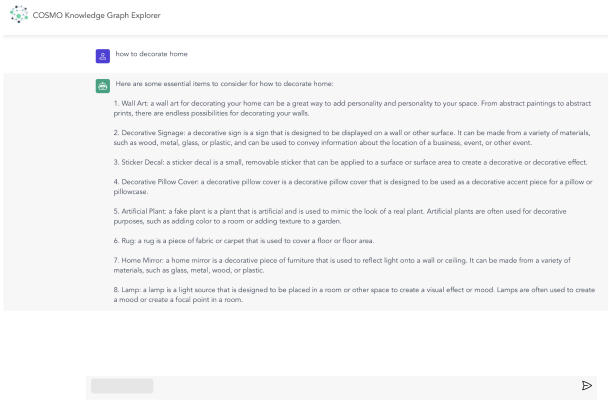


Figure 10: An example of generation from COSMO-LM

product sales within this segment, translating to hundreds of million dollars in annual revenue surge. Additionally, an 8% increase in navigation engagement rate was observed within the same traffic segment, highlighting improved customer interaction and satisfaction. These outcomes are especially significant considering they were derived from the implementation of a single, relatively minor feature on the search page with limited showroom visibility, as illustrated in Figure 9. The success of this initial implementation indicates a tremendous opportunity: by extending the adaptation of COSMO-LM to encompass all traffic for navigation, we anticipate the potential to generate a revenue increase in the billions. Moreover, this promising outcome also underscores the vast potential of leveraging the COSMO-LM across a variety of other features and applications, opening new avenues for enhanced user experience and business growth.

5 CONCLUSION AND DISCUSSION

In this paper, we propose finetuning language models on a collection of e-commerce annotated data, phrased as instructions, to generate high-quality commonsense knowledge that aligns with human preferences. To gather large-scale and diverse instruction data, we design an automatic instruction generation pipeline based on massive user behaviors. Scaling up product domains, relation types, and finetuned tasks achieves scalable knowledge extraction. Furthermore, downstream applications, such as semantic relevance and session-based recommendation, demonstrate the effectiveness of knowledge generated from instruction-finetuned language models. Compared to directly distilling knowledge from large language models, the instruction-finetuned models, with fewer parameters, offer significant advantages in terms of model inference efficiency. Our work represents the first step in aligning language models with domain-specific human preferences, and we hope that the automatic instruction data pipeline can be applied to other fields.

ACKNOWLEDGMENTS

Yangqiu Song was supported by the RIF (R6020-19 and R6021-20) and the GRF (16211520 and 16205322) from RGC of Hong Kong. Yangqiu Song thanks the support from the UGC Research

Matching Grants (RMGS20EG01-D, RMGS20CR11, RMGS20CR12, RMGS20EG19, RMGS20EG21, RMGS23CR05, RMGS23EG08).

REFERENCES

[1] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional AI: Harmlessness from AI Feedback. *arXiv preprint arXiv:2212.08073* (2022).

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *NeurIPS* (2020), 1877–1901.

[3] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality. https://lmsys.org/blog/2023-03-30-vicuna/

[4] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416* (2022).

[5] Shumin Deng, Chengming Wang, Zhoubo Li, Ningyu Zhang, Zelin Dai, Hehong Chen, Feiyu Xiong, Ming Yan, Qiang Chen, Mosha Chen, Jiaoyan Chen, Jeff Z. Pan, Bryan Hooi, and Huajun Chen. 2022. Construction and Applications of Billion-Scale Pre-trained Multimodal Business Knowledge Graph. *ArXiv* abs/2209.15214 (2022).

[6] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. In *ICLR*.

[7] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR*.

[8] Feng-Lin Li, Hehong Chen, Guohai Xu, Tian Qiu, Feng Ji, Ji Zhang, and Haiqing Chen. 2020. AliMeKG: domain knowledge graph construction and application in e-commerce. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2581–2588.

[9] Lei Li, Yongfeng Zhang, and Li Chen. 2020. Generate Neural Template Explanations for Recommendation. In *CIKM*. 755–764.

[10] Sen Li, Fuyu Lv, Taiwei Jin, Guiyang Li, Yukun Zheng, Tao Zhuang, Qingwen Liu, Xiaoyi Zeng, James Kwok, and Qianli Ma. 2022. Query Rewriting in TaoBao Search. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 3262–3271.

[11] Edo Liberty, Zohar Karnin, Bing Xiang, Laurence Rouesnel, Baris Coskun, Ramesh Nallapati, Julio Delgado, Amir Sadoughi, Yury Astashonok, Piali Das, et al. 2020. Elastic machine learning algorithms in amazon sagemaker. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 731–737.

[12] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: Short-Term Attention/Memory Priority Model for Session-based Recommendation. In *SIGKDD*. 1831–1839.

[13] Xusheng Luo, Le Bo, Jinhang Wu, Lin Li, Zhiy Luo, Yonghua Yang, and Keping Yang. 2021. AliCoCo2: Commonsense Knowledge Extraction, Representation and Application in E-commerce. In *SIGKDD*. 3385–3393.

[14] Xusheng Luo, Luxin Liu, Yonghua Yang, Le Bo, Yuanpeng Cao, Jinghang Wu, Qiang Li, Keping Yang, and Kenny Q Zhu. 2020. AliCoCo: Alibaba e-commerce cognitive concept net. In *SIGMOD*. 313–327.

[15] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332* (2021).

[16] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *EMNLP*. 188–197.

[17] Priyanka Nigam, Yiwei Song, Vijai Mohan, Vihan Lakshman, Weitian Ding, Ankit Shingavi, Choon Hui Teo, Hao Gu, and Bing Yin. 2019. Semantic product search. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2876–2885.

[18] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155* (2022).

[19] Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277* (2023).

[20] Yincen Qu, Ningyu Zhang, Hui Chen, Zelin Dai, Zezhong Xu, Chengming Wang, Xiaoyu Wang, Qiang Chen, and Huajun Chen. 2022. Commonsense Knowledge Salience Evaluation with a Benchmark Dataset in E-commerce. *arXiv:2205.10843* (2022).

[21] 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:2206.06588

[22] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *EMNLP-IJCNLP*. 3982–3992.

[23] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *WWW*. 811–820.

[24] Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Scao, Arun Raja, et al. 2022. Multitask Prompted Training Enables Zero-Shot Task Generalization. In *International Conference on Learning Representations*.

[25] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *the AAAI*. 3027–3035.

[26] William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. 2022. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802* (2022).

[27] Tobias Schröder, Terrence C Stewart, and Paul Thagard. 2014. Intention, emotion, and action: A neural theory based on semantic pointers. *Cognitive science* 38, 5 (2014), 851–880.

[28] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd international conference on world wide web*. 373–374.

[29] Amit Singhal et al. 2001. Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.* 24, 4 (2001), 35–43.

[30] Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*. 4444–4451.

[31] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1441–1450.

[32] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.

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

[34] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).

[35] Meirui Wang, Pengjie Ren, Lei Mei, Zhumin Chen, Jun Ma, and Maarten de Rijke. 2019. A Collaborative Session-based Recommendation Approach with Parallel Memory Modules. In *SIGIR*. 345–354.

[36] Shoujin Wang, Longbing Cao, Yan Wang, Quan Z Sheng, Mehmet A Orgun, and Defu Lian. 2021. A survey on session-based recommender systems. *ACM Computing Surveys (CSUR)* 54, 7 (2021), 1–38.

[37] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-Instruct: Aligning Language Model with Self Generated Instructions. *arXiv preprint arXiv:2212.10560* (2022).

[38] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022. Super-NaturalInstructions: Generalization via Declarative Instructions on 1600+ NLP Tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 5085–5109. https://aclanthology.org/2022.emnlp-main.340

[39] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xianling Mao, and Minghui Qiu. 2020. Global Context Enhanced Graph Neural Networks for Session-based Recommendation. In *SIGIR*. 169–178.

[40] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned Language Models are Zero-Shot Learners. In *ICLR*.

[41] Peter West, Chandra Bhagavatula, Jack Hessel, Jena Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2022. Symbolic Knowledge Distillation: from General Language Models to Commonsense Models. In *NAACL*. 4602–4625.

[42] Fanyou Wu, Yang Liu, Rado Gazo, Benes Bedrich, and Xiaobo Qu. 2022. Some Practice for Improving the Search Results of E-commerce. *arXiv preprint arXiv:2208.00108* (2022).

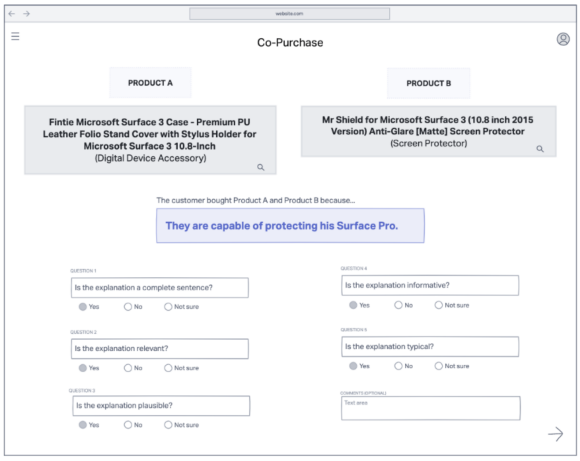


Figure 11: Screenshot of data annotation interface.

[43] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-Based Recommendation with Graph Neural Networks. In *AAAI*. 346–353.

[44] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph Contextualized Self-Attention Network for Session-based Recommendation. In *IJCAI*. 3940–3946.

[45] Changlong Yu, Weiqi Wang, Xin Liu, Jiaxin Bai, Yangqiu Song, Zheng Li, Yifan Gao, Tianyu Cao, and Bing Yin. 2023. FolkScope: Intention Knowledge Graph Construction for Discovering E-commerce Commonsense. In *Findings of the Association for Computational Linguistics: ACL 2023*.

[46] Nasser Zalmout, Chenwei Zhang, Xian Li, Yan Liang, and Xin Luna Dong. 2021. All You Need to Know to Build a Product Knowledge Graph. In *SIGKDD*. 4090–4091.

[47] Ningyu Zhang, Qianghuai Jia, Shumin Deng, Xiang Chen, Hongbin Ye, Hui Chen, Huaixiao Tou, Gang Huang, Zhao Wang, Nengwei Hua, et al. 2021. Alicg: Fine-grained and evolvable conceptual graph construction for semantic search at alibaba. In *SIGKDD*. 3895–3905.

[48] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv:2205.01068* (2022).

A KNOWLEDGE GENERATION

We present generation examples for each category in Table 9.

Table 9: Examples of Generations for Different Categories.

Category	Example
Clothing, Shoes & Jewelry	used for biking
Sports & Outdoors	capable of providing arch support
Home & Kitchen	used for peeling potatoes
Patio, Lawn & Garden	capable of hanging out in the backyard
Tools & Home Improvement	used for sharpening scissors
Musical Instruments	used for wedding party
Industrial & Scientific	capable of holding a lot of weight
Automotive	capable of digging a hole.
Electronics	used to prevent blisters
Baby Products	capable of keeping the baby's feet dry
Arts, Crafts & Sewing	used for stamping on fabric
Health & Household	capable of hydrating the skin
Toys & Games	capable of flying in the air
Video Games	used to protect the headset
Grocery & Gourmet Food	used to make potato chips
Office Products	used for writing down important information
Pet Supplies	used for walking the dog
Others	capable of tracking calories burned

B KNOWLEDGE ANNOTATION

The instructions of data annotation is summarized into five aspects:

- **Completeness**: the explanation must be a complete, meaningful sentence.
- **Relevance**: the explanation should be relevant — i.e., very closely connected in meaning — to the products it refers to.
- **Informativeness**: remember that each explanation describes the shopping behavior of a customer, and in so doing, it should also specify what the user may be looking for in terms of a product's functional requirements.
- **Plausibility**: the explanation should describe the user's shopping behavior in a way that is accurate, reasonable and appropriate in the particular context determined by the query.
- **Typicality**: although we may have equally valid inferences about a customer's shopping intention, those statements can be ranked differently with regard to how representative they are of typical user shopping behavior given what is known about the queried product.