Improving Search for New Product Categories via Synthetic Query Generation Strategies

Akshay Jagatap
Amazon

Srujana Merugu
Amazon

Prakash Mandayam Comar
Amazon

ABSTRACT

Efficient retrieval and ranking of relevant products in e-commerce product search relies on accurate mapping of queries to product categories. This query classification typically utilizes a combination of textual and customer behavioral signals. However, new product categories often lack customer interaction data leading to poor performance. In this paper, we present a novel approach to mitigate this cold start problem in product ranking via synthetic generation of queries as well as simulation of customer interactions. Specifically we study two strategies for synthetic data generation: (i) fine-tuning a generative language model (LLM) on historical product-query interactions and using it to generate synthetic queries from the product catalog, (ii) Bayesian prompt optimization with an instruct-tuned LLM to directly generate queries from catalog. Empirical evaluation of the proposed approaches on public datasets and real-world customer queries demonstrates significant benefits (+2.96% and +2.34% in PR-AUC on e-commerce queries)[1] relative to the baseline approach without synthetic data augmentation. Furthermore, evaluation of the augmented model on live search page results in a substantial increase in highly relevant product results (+3.35%) and reduction (-3.07%) in irrelevant results.

CCS CONCEPTS

• **Computing methodologies** → *Neural networks*; **Natural language generation**; • **Information systems** → **Information retrieval query processing**.

KEYWORDS

Data Augmentation, Generative Language Models, Prompting, Query Classification

ACM Reference Format:
Akshay Jagatap, Srujana Merugu, and Prakash Mandayam Comar. 2024. Improving Search for New Product Categories via Synthetic Query Generation Strategies. In *Companion Proceedings of the ACM Web Conference 2024 (WWW '24 Companion), May 13–17, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3589335.3648299

1 INTRODUCTION

Finding relevant products from a vast ever-changing catalog of millions of products is one of the core utilities of an online e-commerce store. However, this task is often challenging due to two key reasons. First, the notion of product relevance for a query often varies across product categories [10]. For instance, in fashion-related categories, brand name match between query and products is of high importance but in the case of pharmacy category, it is less critical compared to a match on the active ingredients and the medicine strength. Second, ranking models, by virtue of their training, often exhibit a favourable bias towards established products and established product categories compared to new ones [34]. To tackle these challenges, queries are first classified along product categories. The output labels are then used to boost ranking of the products from the associated categories and offer differential customer experience that can improve the customer search experience such as showing a form to support advanced search. Our primary motivation for query classification was to enable restriction of search results to products from relevant categories.

Typically, query classification models are trained in a supervised manner and rely on label information obtained from observing customer interactions such as clicks, cart-adds and purchases [16]. However, for new product categories and new products with little or no customer interaction data, this approach suffers from the cold start problem [12], leading to poor performance and a negative impact on customer search experience, e.g., pharmacy-related queries such as "Paracetamol" (pain relief drug) retrieve irrelevant products such as "Parachute Coconut Hair Oil" in the top positions. This problem is further aggravated for categories such as pharmacy where customers search with a highly targeted intent and abandon the search early in the absence of relevant results.

Despite the availability of the product catalog, it cannot be directly leveraged for improving classification [18] since the product names (or n-grams) are not sufficiently representative of the customer query distribution. Specifically, designing a query classifier requires addressing three key challenges: (a) *Linguistic variations:* Customer queries, especially in markets such as India, tend to be short, non-specific, abbreviated, ridden with spelling mistakes and interspersed with words from native language, etc. For instance, the product titled "Dolo 650 Strip of 15 Tablets" is a top match for multiple queries ["doloo 15", "fever medicine hai" [2], "paracetamol"], none of which can be directly derived from the product name, (b) *Mapping ambiguity of non-specific queries:* Real query logs indicate a high prevalence of non-specific queries such as "dandruff shampoo" that map to multiple categories such as pharmacy, health & personal care, and beauty, and (c) *Highly limited supervision on query labels and frequency:* Human expertise for labeling abandoned queries (i.e., queries without clicked or purchased products) from past search logs and potential new queries is costly and highly limited. Estimating their likely prevalence is also highly non-trivial.

[1]Note that all reported performance improvements are absolute values, e.g., +2.34% corresponds to change in PR-AUC from 0.782 to 0.805.

WWW '24 Companion, May 13–17, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-0172-6/24/05.
https://doi.org/10.1145/3589335.3648299

[2]Transliteration of Hinglish [25] phrase meaning "Is fever medicine available?".

Inspired by recent findings [22, 23] on the benefits of data augmentation with large generative language models (LLMs), in this work, we propose a novel approach to solve the query classification problem, especially for new product categories with low search volume and specialised search vocabulary based on synthetic generation of queries. Below we summarize our contributions:

1. [**Data augmentation with synthetic query-product pairs.**] We pose the query classification problem as a binary in-category vs. out-of-category classification with query labels induced from the labels of the positively associated (e.g., clicked) products. We propose a generic framework for augmenting the observed query-product pairs with synthetic pairs generated from the full product catalog that mimic observed customer behavior in terms of the linguistic variations (dialects, spell errors, hybrid languages), category label ambiguity, and the prevalence frequency. This augmentation approach yields models that can provide superior performance on underrepresented and cold-start queries, and can be used with any choice of generative LLM. Furthermore, since our approach yields synthetic query-product pairs with interaction volumes and not just labeled queries, it can also be utilized to improve ranking.[3]
2. [**Bayesian prompt optimization.**] Most powerful LLMs cannot be directly fine-tuned due to computational costs or lack of access to model parameters and only provide API access that permits prompting. Hence, we propose a Bayesian approach for prompt optimization that propagates the loss from the query classifier to LLM prompting module to progressively improve the prompt generation policy along various factors (e.g., spelling error and query length distribution) and generate better synthetic queries that can yield higher performance on the underlying classification task.
3. [**Superior empirical performance on Pharmacy data.**] We explore multiple query generation strategies based on fine-tuning and prompting with LLMs such as Distil-GPT2 [15], Flan-T5-XXL and Flan-UL2 [7]. Using data from an e-commerce Pharmacy store, we investigate the benefits of synthetic query augmentation (RQ1), the relative performance of different generation methods (RQ2), and the benefits of interaction volume estimation (RQ3). Empirical results point to significant improvement on query classification performance from data augmentation (+2.43% PR-AUC) especially for cold-start queries (+6.47% PR-AUC). The deployed augmentation-based model resulted in a substantial increase in highly relevant product results (+3.35%) and reduction (-3.07%) in irrelevant results.

2 QUERY CLASSIFICATION PROBLEM

Let $\mathcal{A} = \mathcal{A}_1 \bigcup \mathcal{A}_0$ denote the product catalog with $\mathcal{A}_1$ and $\mathcal{A}_0$ corresponding to the in-category and out-of-category products respectively. Let $\mathcal{Q}$ be the space of all customer text queries. Denoting the true relevance of a product $a \in \mathcal{A}$ for a query q by $p^{rel}(a|q)$, we can induce a soft classification on the query space with respect to category membership $y : \mathcal{Q} \mapsto \{0, 1\}$,

$$p^{true}(y = 1|q) = \frac{\sum_{a \in \mathcal{A}} \mathbb{1}(a \in \mathcal{A}_1) p^{rel}(a|q)}{\sum_{a \in \mathcal{A}} p^{rel}(a|q)} \quad (1)$$

In practice, the true relevance of a product for a query is unknown and instead, we only have observations that are shaped by the existing ranking system. Let $p^{seen}(a|q)$ denote the probability of a product a being "seen" [4] by a customer for a query q (accounting for positional effects) and $v(a, q)$ denote the observed "interaction volume" for the pair (q, a). One can then assume $v(a, q) \propto p^{seen}(a|q)p^{rel}(a|q)$. This interaction volume can be computed based on the frequency and intensity of customer interactions such as clicks, cart-adds, and purchases.

Given the observed query-product interactions $v_{train} : \mathcal{Q} \times \mathcal{A} \mapsto \mathcal{R}^+$, existing ranking, i.e., $p^{seen}(a|q)$, $\forall a \in \mathcal{A}, q \in \mathcal{Q}$, and the entire product catalog $\mathcal{A}$ along with product features, the objective is to accurately learn to classify any query $q \in \mathcal{Q}$ with respect to its category-membership, i.e., predict $\hat{p}(y|q)$ such that it approximates the true query classification probability $p^{true}(y|q)$.

Since the true relevance is unknown, we can instead evaluate with respect to the estimated probability $p^{test}(y|q)$ for an unseen test query with product relevance to query being proportional to $v^{test}(a, q)/p^{seen}(a|q)$. Even though query-product relevance in train and test settings might follow the same underlying distribution, the prediction problem is challenging since the observed interactions in train data are biased due to the ranking system and might not include new products and queries. Note that while offline evaluation with respect to an unseen future test period (with existing ranking) provides a directional sense of the goodness, since the query classification affects the ranking itself, the true benefits of improved classification can be assessed only based on increase in interaction volume in an online experiment.

3 RELATED WORK

In recent years, large language models based on the Transformer architecture [28] have made significant advancements in natural language processing. Encoder models like BERT have led to stellar results on tasks such as text classification, summarization, named entity tagging. Models equipped with decoder such as BART [14], T5 [21], Alexa-TM [26], UL2 [27], GPT-2 [20] and GPT-3 [4] have shown versatility in tasks like question-answering and dialog generation. These models can be customized for specific tasks through fine-tuning on task-specific datasets or prompt engineering. FLAN [30] and InstructGPT [19] introduced the concept of instruction tuning, where a language model is fine-tuned on instruction prompts from various tasks to generalize to new tasks.

Prompt engineering and tuning [13, 17] has emerged as a new paradigm for customizing frozen pretrained language models, where the model parameters cannot be changed either due to computational reasons or lack of access to parameters. Multiple works [11, 31] demonstrate the potential for improving downstream task performance either through customized soft prompts [13] or customized NL prompts [9] especially for instruction-tuned models such as InstructGPT and FLAN. Of these techniques, NL-based prompts use reinforcement learning based methods that can be computationally expensive. We propose a lightweight approach where we represent the prompt generation policy in terms of human-understandable linguistic factors (e.g., spell error rate, query length

[3]We report results on both query classification and ranking on public datasets. On our internal dataset, we only report results on query classification to focus on the query generation aspects and avoid details of the proprietary ranking system.

[4]Though we have the impression count for each product-query pair, it is difficult to ascertain if an impressed product is truly seen especially by customers with small devices. Hence, we rely on click/add-to-cart/purchase counts as more reliable indicators.

distribution) and optimize it using Sequence Model-Based Optimization (SMBO) [1]. This also provides explainability that can serve as a guardrail to prevent unintended synthetic generation.

With the advent of generative LLMs that encode substantial world knowledge, there has also been an increased interest in utilising LLMs for synthetic query generation. Most of the prior work in this area is focused on question-answering and binary relevance prediction where the models generate a query relevant to a given document. A recent work label-conditioned QGen [5] focuses on query generation for e-commerce products with multi-level relevance ranking using simple prompting and fine-tuning. However, the evaluation is primarily performed in a transfer learning setting and does not take into account the interaction volume of query-result pairs which is critical in real applications. We attempt to address this gap and also evaluate our approach on some of the datasets studied in [5]. Note that while our work uses specific LLMs, namely tiny-BERT [3], Flan-T5-XXL (encoder) for query classification, Distil-GPT2, Flan-T5-XXL, and Flan-UL2 for query generation using fine-tuning and prompt engineering, our proposed approach is agnostic of the choice of the models.

4 PROPOSED APPROACH

We now present our approach for query classification that addresses the challenge of product search logs not being adequately representative of new product categories such as Pharmacy. Figure 1A provides an overview of our approach, which relies on augmenting the observed query-product interactions in the search logs with synthetic data generated by simulating the interactions of customers with intent of buying products in the target category. For each query, the affinity towards the target category is computed based on its affinity with respect to the products in and outside the target category. The classifier is learned on the entire query pool using a standard encoder-based only model. Our primary focus is on the simulation, which is a two-step process: 1) synthetic query generation and 2) interaction volume estimation for the synthetic queries. The query generation step (see Section 4.1) ensures a diverse representation across the query pool. This step ensures inclusion of clear positive and negatives examples in terms of relevance to the target category and also ambiguous queries that can could pertain to products from both target category and overlapping categories (e.g., the query "tretinoin" could refer to acne medication in the Pharmacy category as well as a skin care product in Beauty category). For the generated queries, we then compute likely query-product interaction volume (see Section 4.2) using distributional statistics of the product demand as well as the likely query generation probability.

4.1 Query generation strategy

We present two approaches namely fine-tuning and prompting, to leverage large pretrained generative language models (LLM) [33] to generate high-quality and contextually relevant synthetic queries.

4.1.1 Fine-tuning Generative LLMs. In the first approach (shown in Figure 1C), we build a dataset of relevant (product, query) pairs using positive interactions (e.g., clicks, purchases) from historical customer search logs. We utilize the product catalog to construct a training dataset (Figure 2C) of input-output pairs where the input comprises the product name and attributes and the output is a search query associated with that product. Fine-tuning the LLM on this dataset enables it to learn the associations between product features and search queries as well as customers' stylistic variations. Once fine-tuned, the LLM can generate synthetic queries for any new product that mimic real customer searches, capturing the nuances of customer behavior. While this approach enables creation of a much larger and diverse dataset of synthetic queries along with representation for new products, a key drawback is that the generated query distribution is determined by the training data which could be noisy, non-representative and biased towards popular queries. One cannot explicitly control or audit the query generation policy, e.g., length and spelling error rate, since it depends entirely on latent associations learned by the model between product and query. Additionally, in this case, the query generation model is completely decoupled from the main downstream task of interest, i.e., query classification, as a result of which relevant feedback from the classification task cannot be directly passed to improve the fine-tuning.

4.1.2 Prompting. Most powerful instruction-tuned LLMs are available only through API in a frozen blackbox-form that does not permit direct fine-tuning of parameters. However, these models readily allow generation of high quality text sequences through "prompting" without any need for explicit task-specific fine-tuning. To be effective, prompts need to be concise, explicit, relevant to the task at hand, and provide all the necessary context to generate relevant outputs aligned with the intended objective. Hence, for the query generation setting, we consider three key aspects of interest: (a) query length variations (b) stylistic variations, e.g., spelling errors, and (c) inclusion of information associated with product features (e.g., usage of a medicine). We employ two different prompting methods to generate queries: static prompting and dynamic (or policy-based) prompting. To describe the prompt generation, we use the following notation. For each product a in catalog $\mathcal{A}$, let $x_a = [x_a^{cat}, x_a^{name}, x_a^{f_1}, x_a^{f_2}, \cdots]$ denote the associated attributes. Here, x_a^{cat} and x_a^{name} correspond to the category and name while $[x_a^{f_1}, x_a^{f_2} \cdots]$ refer to the rest of product fields (e.g., active ingredient) specific to the category. Let $s(f_i)$ denote a string descriptor for each field, e.g., s("active") = "containing active ingredient".

Static Prompting: Static prompts utilize a fill-in-the-blank template that is parameterized by specific characteristics of products in a given category. For any product category, one can construct a default prompt of the form:

"`Generate queries for` x_a^{cat} x_a^{name} $s(f_1)$ $x_a^{f_1}$ $s(f_2)$ $x_a^{f_2} \cdots$".

For pharmacy products, we used the following template to guide the LLM in generating queries:

"`Generate queries for the prescription drug` x_a^{name} `containing active ingredients` x_a^{active} `used to treat` x_a^{usage}"

By plugging in the relevant values for parameters, we create specific prompts for each product in the catalog and generate multiple queries for each item, leveraging the comprehensive knowledge of the underlying LLM (Figure 2A). Generating queries from static prompting is easy to implement and audit. The queries generated provide a good coverage across products unrepresented in the search click logs while including appropriate context from product attributes based on the patterns learned by the LLM from its own world knowledge. However, a key drawback is that the prompts are

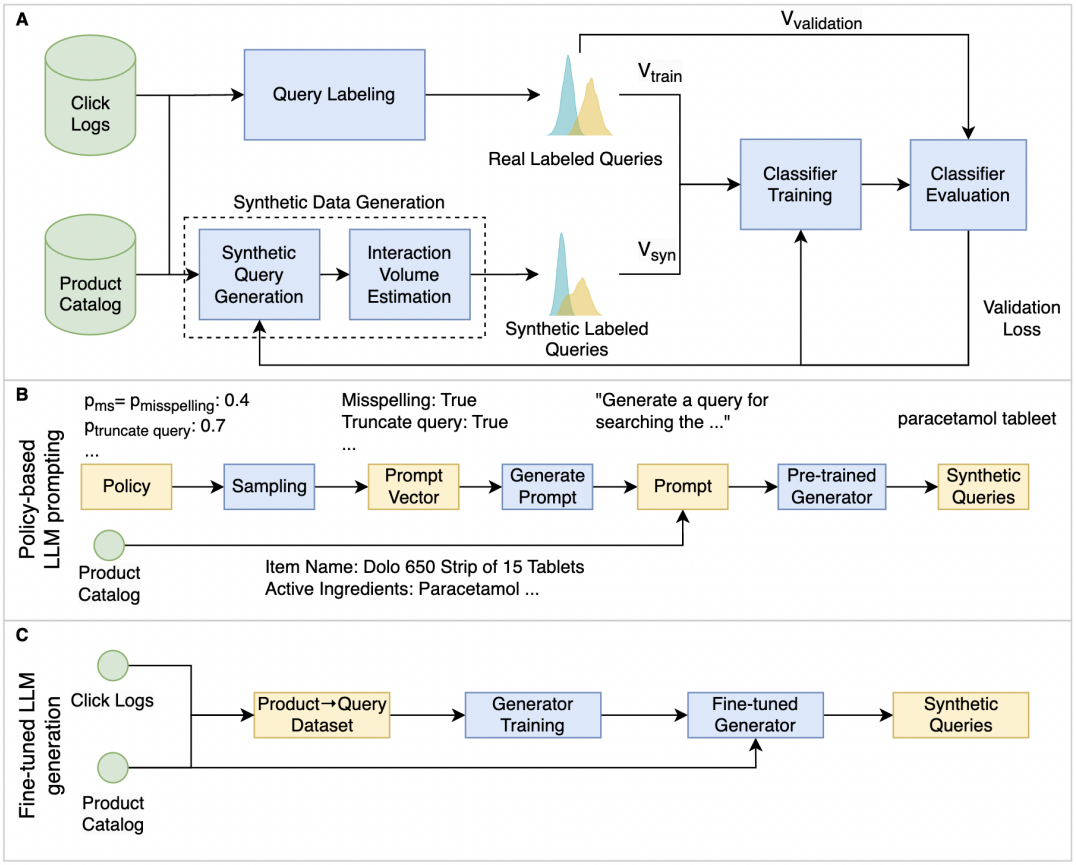

Figure 1: A. Schematic of complete training pipeline of classification model. B. Synthetic query generation via Dynamic policy-based prompting of a pretrained LLM. C. Synthetic query generation using a fine-tuned LLM.

less flexible in adapting to a wide range of scenarios as the fixed template and parameters may not cover all possible variations or context-specific nuances, potentially leading to less diverse and less contextually relevant prompts. Static prompts are prone to bias due to the template structure and parameter choices. For example, not all queries contain active ingredients since customers may search for drugs only by usage.

Dynamic Policy-based Prompting: Dynamic policy-based prompting (shown in Figure 1B) follows a template similar to static prompt except that the prompt components are generated in probabilistic fashion that allows selective retention or dropping of parts of the prompt string to control over the query generation. Figure 2B shows variations of queries that can be constructed for the same product. Table 1 summarises the prompt components and the policy parameters that determine the inclusion probabilities used for Pharmacy category.

The key idea is to consider a prompt generation policy **p** comprising both stylistic and product field inclusion parameters as in Table 1 for Pharmacy domain. Intuitively, the best choice of parameters is the one that optimises the downstream query classification task. Since the modeling pipeline comprising synthetic query generation and classifier model training does not have a closed analytic form, we employ a Bayesian approach [24] that is suitable for optimizing the hyperparameters of a blackbox module that can be evaluated on a validation set. This approach involves iteratively learning of a probabilistic model of the mapping from hyperparameter values to the desired objective function using an explore-exploit approach for evaluating promising hyperparameter configurations taking into account both uncertainty as well as likely utility. Specifically, in the current work, we use Hyperopt [2] to fine-tune the prompting policy parameters to maximize the validation accuracy of the query classifier trained with synthetic queries produced by the underlying language model as shown in Figure 1A. The optimization finds the best policy parameters that strike a balance between query diversity and relevance, resulting in queries that closely resemble authentic customer queries and also improve classification performance. Dynamic prompting allows continual learning of the prompt generation policy which is useful when the query distribution changes over time.

4.2 Interaction volume modeling

For each of queries generated as per Section 4.1, we also estimate the likely interaction volume $v_{syn}(q, a)$ associated with the product-query pair (q, a), which can be used to appropriately weight the

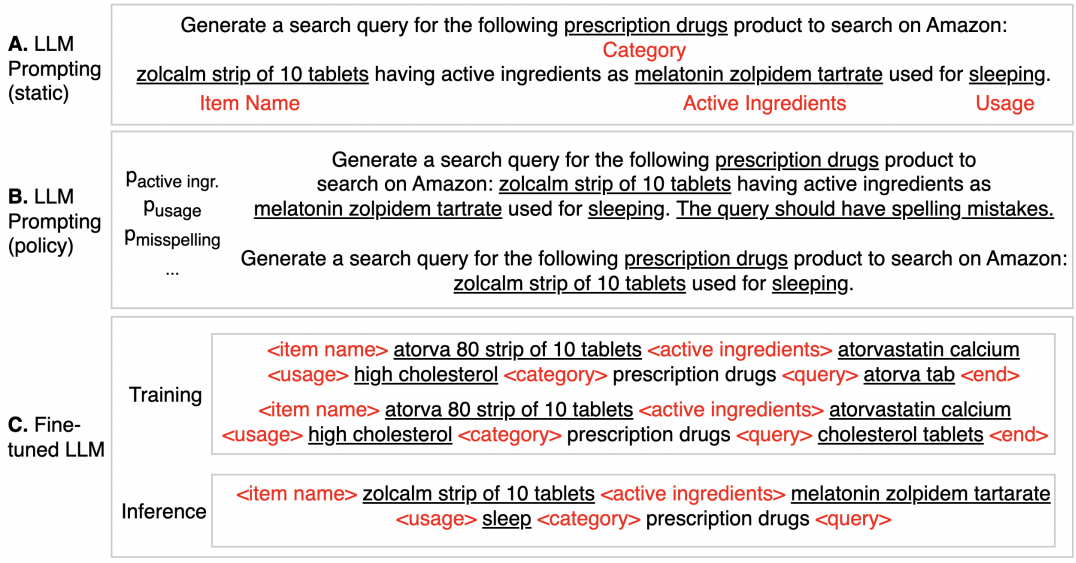

Figure 2: A. Static prompt for the product "zolcalm strip of 10 tablets". B. Dynamic policy-based prompting strategy to generate queries for the same product. C. Fine-tuned LLM-based strategy to generate queries for the same product.

Table 1: Components of a dynamic prompt generation template along with parameters that determine the respective inclusion probabilities and their optimal estimates obtained via Bayesian hyperparameter tuning [24].

Prompt String	Incl. prob.	Distribution	Optimal
Generate queries	1		
of length K	p_t	Bernoulli	0.43
	K ~NB(r, p_k)	Negative Binomial	0.61
for x_a^{name}	1		
with active ingredient x_a^{act}	p_{active}	Bernoulli	0.08
used for treating x_a^{usage}.	p_{usage}	Bernoulli	0.39
Include misspelled queries.	p_{ms}	Bernoulli	0.95

queries during the training of the query classification (or ranking) model. Since the synthetically generated query logs might not occur in past search logs, we model the interaction volume for a product-query pair (q, a) by considering two signals: (i) the customer demand for the product a based on observed search logs with adjustment for new products, and (ii) the probability of generating the query q from the LLM for the product a. Note that the interaction volume considers a weighted combination of clicks, cart-adds, and purchases to ensure high intensity interactions are appropriately accounted for. First, we estimate for each product in the catalog the likely customer demand by considering the sum of customer interactions for that product across all the queries in the training search logs, i.e., $(\sum_{q' \in Q_{train}} v_{train}(q', a))$. To accommodate new products that lack historical interactions, we employ a smoothing adjustment with an interaction volume of 1. For a given query q, we estimate the volume associated with the query-product pair (q, a) as

$$v_{syn}(q, a) = \left(1 + \sum_{q' \in Q_{train}} v_{train}(q', a)\right) \cdot P^{gen}(q|a) \quad (2)$$

where Q_{train} denotes the observed queries and $P^{gen}(q|a)$ is the probability of generating query q given product a by the LLM.

The sum of interaction volumes of query q for products in the class $c \in C = \{0, 1\}$ is given by $\sum_{a \in \mathcal{A}_c} v_{syn}(q, a)$, which can be used to determine the relative weight of query q when combining it with the true observations prior to training the classification model. Since the LLM is trained on the search logs, the probability of generating a query given a product reflects the search log distribution. Hence the above estimation approach is accurate for organic search queries and ensures consistency between synthetic and organic queries. The proposed method thus generates high quality synthetic labeled queries for products in $\mathcal{A}$ along with the associated interaction volumes.

5 EXPERIMENTS

In this section, we demonstrate the efficacy of the proposed synthetic query generation on multiple datasets by presenting our results on the research questions mentioned in Section 1.

5.1 Datasets

We benchmark our approach on an internal proprietary dataset and two external datasets described below:

Ecom-Pharmacy: This data is sampled from real customer interaction data obtained from an e-commerce Pharmacy store. We partition the dataset into three temporally separated sets: `train`, `val`, and `test`. The `train` set consists of data from Dec 2022- Jan 2023 while the `val` and `test` sets consists of data from Feb 2023 and Mar 2023 respectively. In the case of synthetic query generation

via dynamic policy prompting, we first learn the classifier on the `train` split and use the `val` split to determine the ratio for mixing synthetic and real datasets and other prompting policy parameters. Even for synthetic generation via finetuning LLM, we adopt the same approach to determine the mixing ratio with the LLM also fine-tuned on the `train` split. Once the hyperparameters are learned, we retrain the classification model on the combination of `train` and `val` as well as the synthetic data. Finally we evaluate the performance on the unseen `test` set. The temporally disjoint train, validation, and test sets enable us to accurately assess the generalisation capabilities of our classifier and generative models.

To construct the dataset, we start with the pharmacy catalog, which is the ground truth list of products from the target category and create a list of "weak pharmacy intent queries" that have led to at least 5% of clicks from Pharmacy products. For each of these queries, we retrieve all the clicked products ($\mathcal{A}$) and classify them into two categories: Pharmacy and Non-Pharmacy products. Next, we retrieve all the queries that are associated with the products in $\mathcal{A}$. This set of queries goes beyond the initial "weak pharmacy intent queries". For each of these queries, we map it to the binary categories (pharmacy or non-pharmacy) based on the interaction volume with associated products and use this mapping to train the query classifier. For fine-tuning the generator, we directly leverage the product-query pairs associated with the set of products $\mathcal{A}$ weighted by the interaction volume of the product-query pair.

WANDS [6]: This is a product-search relevance dataset released by WayFair primarily focusses on home improvement. It consists of 233,448 product-query relevance judgements with 480 unique queries and 42,994 unique products. The relevance judgements span three levels - namely `ExactMatch`, `PartialMatch` and `Irrelevant`. This dataset also includes product metadata (e.g., product name, class) for all the products. There is no notion of interaction volume, but the availability of product metadata permits evaluation of the benefits of different approaches for the synthetic query generation on the downstream query classification as well as ranking tasks.

Home Depot [8]: Similar to the WANDS dataset, the Home Depot dataset is also comprised of product-query relevance annotations (74K train examples and 166k test examples), the difference being that the annotations are in the form of a real-valued score. We split the data into three buckets -`Irrelevant`, `PartialMatch` `ExactMatch`, by considering score thresholds corresponding to the 33rd and the 66th percentile. As in the case of WANDS, product metadata is also available but there is no interaction volume.

Table 2: Performance of various strategies on correctly identifying the queries with pharmacy intent (PR-AUC, PR-AUC-LF) and synthetic query quality metrics (BERT-score) along with computational costs.

Strategy	PR-AUC	PR-AUC-LF	BERT-score	Compute Cost
None	78.16%	72.16%	-	1x
SP	78.47%	72.13%	83.72%	1.9x
PP	80.50%.	78.63%	82.85%	4.2x
FTG	80.01%	77.03%	90.27%	2.3x
FTF	81.12%	79.05%	91.62%	7x

5.2 Algorithms

We implement our proposed approach using the following LLMs:

BERT-tiny: (4.4M params) is used for query classification for `Ecom-Pharmacy` dataset.
DistilGPT2: (82M params) is a decoder-only model derived from GPT2, which we use for fine-tuning on the `Ecom-Pharmacy` dataset for product to query generation. Due to its small size and lack of instruct-tuning, it is not amenable for prompt-tuning.
FLAN-UL2: (20B params) is an encoder-decoder model based on the T5 architecture with additional instruction tuning. We use it for prompt tuning (static and dynamic) on the `Ecom-Pharmacy` dataset. The large model size makes it expensive to fine-tune and hence, we do not use it for the fine-tuning based query generation.
FLAN-T5-XXL (encoder): (5B params) is the encoder component of FLAN-T5-XXL, fine-tuned for classification task. The model is used for classification task of the two external datasets.
FLAN-T5-XXL: (11B params) is an encoder-decoder model based on T5 architecture with instruction tuning. We use it for fine-tuning and prompt tuning (static and dynamic) for the datasets `WANDS` and `HomeDepot` and also for fine-tuning on `Ecom-Pharmacy`.
Our evaluation considers three query generation strategies: a) **SP**: Static Prompting, b) **PP**:Policy-based Prompting, c) **FT**:Fine-Tuning on user queries. In the case of external datasets, we use Flan-T5-XXL for all the approaches. However, for `Ecom-Pharmacy`, we use Flan-UL2 for the first two prompting based approaches and one smaller model DistilGPT-2 (**FTG**) and one comparable model Flan-T5-XXL (**FTF**) for finetuning. For all the datasets, we train the classifier with real customer queries to construct the baseline and merge the real data with synthetic data to learn the augmented models. For the latter, we use the real customer data as is, while we scale up/down the proportion of synthetic data based on a scaling factor $(0.6 - 1)$, which is determined using hyperparameter tuning.

5.3 Metrics

Classification Metrics. To assess the performance of our classifier model, we measure PR-AUC (Precision-Recall Area Under the Curve) on the out-of-time test set. We report the PR-AUC scores for the entire test set and also specific subsets of interest such as low frequency (bottom 30 percentile) queries (PR-AUC-LF).
Generation Metrics. We also evaluate the quality of the generated queries by employing secondary Natural Language Generation (NLG) metrics. To assess the similarity of the generated queries to human-written queries, we compute the BERT-score [32], which measures the semantic similarity between the generated queries and the target queries.
Ranking Metrics. On the external datasets, since the class labels are ordered, we also evaluate the ranking performance using the approach in [5]. For each query-product pair, we compute the score as follows in case of WANDS datasets.

$$E_i = \sum_{j=\{E,P,I\}} p(y_i^j | x_i) * w_j$$

$$w_j = \{E = 2.0, P = 1.0, I = 0.0\},$$

Table 3: Sample queries through different generation strategies on `Ecom-Pharmacy` dataset.

Customer Queries	FTG	SP	PP	FTF
zolcalm	zolcalm	zolcalm	zolcalm strip of 10 tablets	zolcalm
	zolcalm tablet	zolpidem tartrate	zolcalm	alprax
	melatonin zolpidem tartrate	zolcalm strips	zollicalm tartrate strips	sleeping aid tablet
	alprax	zolcalm tablet	zolcalm strip of ten tableets	melatonin tablet

Table 4: Performance of various strategies on classification task (PR-AUC), ranking (NDCG@10) and synthetic query quality metrics (BERT-score).

Strategy	PR-AUC (micro)	NDCG@10	BERT-score
WANDS			
None	81.10%	94.11%	-
SP	81.46%	94.31%	82.68%
PP	82.00%	94.76%	83.99%
FT	82.45%	95.22%	85.57%
Home Depot			
None	63.80%	96.45%	-
SP	64.31%	97.72%	82.07%
PP	64.82%	97.78%	83.01%
FT	65.33%	97.98%	86.38%

where E, P, I denote `ExactMatch`, `PartialMatch` and `Irrelevant` respectively. We then compute Normalized Discounted Cumulative Gain (NDCG@10)[29] for queries by ranking products based on E_i.

5.4 Results & Discussion

In this section, we discuss the relative efficacy of our proposed strategies for synthetic generation and interaction volume estimation in the context of our research objectives.

RQ1. *Does augmenting training with synthetic data improve classification performance?*

Table 2 presents a comparison of performance of the classifiers trained with synthetic data along with that of the baseline ("None") where no query augmentation is employed. We observe that the classifier trained with synthetic data generated using the FTF strategy exhibits the highest improvement, achieving a significant increase of +2.96% compared to the baseline. Next, the classifier trained with synthetic data generated using PP demonstrates a performance increment of +2.34%. Following closely, the FTG strategy improves the performance by +1.85%, since DistilGPT-2 is considerably smaller than the FLAN-T5-XXL model. On the other hand, the SP strategy yields the lowest improvement of only +0.31% over the baseline. Classification performance on low frequency queries (PR-AUC-LF) follows the same trend with FTF yielding the highest improvement followed by PP, then FTG and SP, respectively but the gains from synthetic generation using PP, FTG and FTF are substantially more. We observe similar behaviour on the external datasets as shown in Table 4 where the proposed approach consistently performed better than baseline of not augmenting the train set with synthetic samples. Since, the classification task on external datasets had three labels each, the PR-AUC reported is micro-averaged across the labels. Even in case of the ranking task, we observe that the ranking performance (NDCG@10) consistently improves when the training data is augmented with synthetic queries with the relative gains following the same trend (i.e., FT outperforms PP which in turn outperforms SP) as in the case of classification.

RQ2. *How do the synthetic generation strategies differ?*

To answer this question, we evaluate the models based on natural language generation metrics, specifically focusing on the BERT-score. For the test data, we consider each customer search query and the associated product. We generate eight synthetic queries for each product and compute the BERT-score by comparing the customer search query with the synthetic queries. The results show that the synthetic queries generated using FTF achieve the highest BERT-score (see BERT-score in Table 2 of 91.62%. This outcome is expected since Flan-T5-XXL is fine-tuned on historical product-to-query data from customers, enabling it to generate synthetic queries that closely resemble customer queries. The FTG strategy achieves the next highest BERT-score of 90.27%, since it is also fine-tuned on customer data but does not have the same modeling capacity due to smaller model size. While the policy-based prompt strategy (PP) yields the lowest BERT score of 82.85% since it is not trained on product-to-query data, it does more diversity in the training (see Table 3), which helps improve the classifier performance. We also observe that all strategies, except PP, result in similar average query lengths of about 3.2. The PP strategy yields an average query length of 3.48, reinforcing the claim that the diversity introduced by this strategy contributes the most to improving classification accuracy. Table 1 lists the optimal choice of parameters determined using Bayesian optimization approach on the internal Pharmacy query dataset, which provides a sense of customer query distribution. For example, the high value of $p_{ms} = 0.95$ indicates high rate of spelling errors while $p_{active} = 0.08$ points to active ingredients rarely being mentioned in the queries. These findings highlight the effectiveness of different synthetic generation strategies in improving classification performance and provide insights into the strengths and weaknesses of each strategy. Comparing the compute cost for each of the strategies (see Table 2), we observe that the relative compute cost for FTF is the highest and PP is the next highest, while that of SP and FTG strategies are the lowest among the augmentation strategies. In general, the results point to the efficacy of synthetic generation by fine-tuning (FT) a moderate-sized LLM or policy-prompting (PP) a high performant LLM. The appropriate choice for a context would depend on considerations such as labeled data availability, computational costs, as well as the relative importance of downstream objective and adherence to observed distribution.

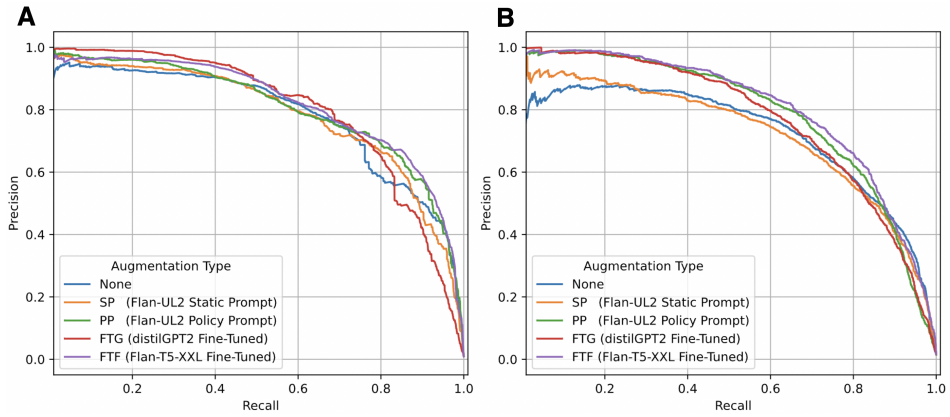

Figure 3: Precision vs Recall plot on `Ecom-Pharmacy` dataset computed across A. the complete test set and B. computed across low frequency keywords (with less than 30 percentile interaction volume) in test set.

Table 5: Ablation experiment with and without interaction volume estimation for FTF model on `Ecom-Pharmacy` dataset.

Model	PR-AUC	PR-AUC-LF
With Interaction Vol.	81.12%	79.05%
Without Interaction Vol.	80.20%	78.80%

RQ3. *What is the impact of interaction volume estimation?* To evaluate the benefits of the interaction volume estimation, we performed an ablation experiment without uniform weighting (no volume estimation) on the synthetically generated queries for the `Ecom-Pharmacy` dataset (refer Table 5). We observe a difference in overall PR-AUC of 0.92% and PR-AUC-LF of 0.25% which can be attributed to the weighting based on interaction volume. In the absence of interaction volume estimation, all the generated keywords are weighted equally, which results in high-volume queries being under-represented and the low-volume ones being over-represented. Hence, the drop in performance for low-frequency keywords is not as much as the overall drop.

6 DEPLOYMENT

We performed a live A/B test comparing the search results of the proposed approach against the production model under following setup. For the queries identified as pharmacy intent by the proposed model we restrict the retrieved results to products from the pharmacy store in the treatment and in the control the retrieved results are ranked by the production model. For evaluation, the query-product result pairs were split into four groups: `ExactMatch`, `Substitute`, `Complement` and `Irrelevant` [19], with the primary metric being the fraction of `Irrelevant` results for a query (less is better) and safety guardrails on the fractions of other groups. We observed that the irrelevance rate in the treatment decreased significantly by 3.07% compared to the control, pointing to the superiority of the proposed approach. For the head/torso/tail queries, the irrelevance rate decreased by 1.30%, 2.32% and 4.14% respectively in the treatment. The treatment also outperformed control by retrieving +3.35% more `ExactMatch` products and +0.23% more `Substitute` products compared to control. However, the treatment fetched -0.51% lower complementary products compared to the control. Since complementary (e.g., fetching "steamer" for "cough syrup" related queries) products had low relevance, the end user experience was consistently improved with increase in `ExactMatch` and `Substitute` results and reduction of `Complement` and `Irrelevant` results leading to production deployment.

7 CONCLUSION

Our work presents a novel methodology to improve search query classification for new categories under-represented in historical customer search activity using synthetic augmentation through generative LLMs and simulation of likely interactions. We evaluate two generation approaches based on fine-tuning an LLM with historical data and Bayesian prompt optimization of a LLM accounting for downstream task performance. Our key learnings can be summarised as follows: (i) Incorporating synthetic queries into the training of the classification and ranking models in a systematic way leads to superior model performance with benefits amplified for low-frequency and cold-start scenarios, where customer interaction data is scarce. (ii) Choice of the best augmentation strategy depends on multiple consideration such as (a) availability of data to fine-tune the model, (b) access to computing resources, (c) access to model parameters. Dynamic Policy-based Prompt optimization (PP) offers a feasible high performing solution even when fine-tuning is not possible. (iii) Query characteristics vary with the generation approach. For instance, the prompt optimization (PP) results in more diverse queries that have low BERT-scores relative to the fine-tuning approach (FTG, FTF) with respect to historical data but yield high test classification performance. Our findings on the benefits and considerations of training with synthetic data, such as the trade-off between query realism and diversity, have broad applicability to other classification and ranking scenarios that utilize a combination of text and customer behavioral signals.

REFERENCES

[1] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for Hyper-Parameter Optimization. In *Advances in Neural Information Processing*

Systems.

[2] James Bergstra, Brent Komer, Chris Eliasmith, Dan Yamins, and David D Cox. 2015. Hyperopt: a Python library for model selection and hyperparameter optimization. (2015).

[3] Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. Generalization in NLI: Ways (Not) To Go Beyond Simple Heuristics. arXiv:2110.01518 [cs.CL]

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165 [cs.CL]

[5] Aditi Chaudhary, Karthik Raman, Krishna Srinivasan, Kazuma Hashimoto, Mike Bendersky, and Marc Najork. 2023. Exploring the Viability of Synthetic Query Generation for Relevance Prediction. (2023).

[6] Yan Chen, Shujian Liu, Zheng Liu, Weiyi Sun, Linas Baltrunas, and Benjamin Schroeder. 2022. WANDS: Dataset for Product Search Relevance Assessment. In *Proceedings of the 44th European Conference on Information Retrieval*. 12 pages.

[7] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling Instruction-Finetuned Language Models. arXiv:2210.11416 [cs.LG]

[8] Home Depot Product Search Relevance Dataset. 2016. https://www.kaggle.com/c/home-depot-product-search-relevance

[9] Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P. Xing, and Zhiting Hu. 2022. RLPrompt: Optimizing Discrete Text Prompts with Reinforcement Learning. arXiv:2205.12548 [cs.CL]

[10] Bernard J. Jansen and Danielle Booth. 2010. Classifying Web Queries by Topic and User Intent. In *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (Atlanta, Georgia, USA) *(CHI EA '10)*. Association for Computing Machinery, New York, NY, USA, 4285–4290. https://doi.org/10.1145/1753846.1754140

[11] Tomohito Kasahara, Daisuke Kawahara, Nguyen Tung, Shengzhe Li, Kenta Shinzato, and Toshinori Sato. 2022. Building a Personalized Dialogue System with Prompt-Tuning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Student Research Workshop*. Association for Computational Linguistics, Hybrid: Seattle, Washington + Online, 96–105. https://doi.org/10.18653/v1/2022.naacl-srw.13

[12] Vijay Kotu and Bala Deshpande. 2019. Chapter 11 - Recommendation Engines. In *Data Science (Second Edition)* (second edition ed.), Vijay Kotu and Bala Deshpande (Eds.). Morgan Kaufmann, 343–394. https://doi.org/10.1016/B978-0-12-814761-0.00011-3

[13] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. arXiv:2104.08691 [cs.CL]

[14] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. arXiv:1910.13461 [cs.CL]

[15] Tianda Li, Yassir El Mesbahi, Ivan Kobyzev, Ahmad Rashid, Atif Mahmud, Nithin Anchuri, Habib Hajimolahoseini, Yang Liu, and Mehdi Rezagholizadeh. 2021. A Short Study on Compressing Decoder-Based Language Models. arXiv:2110.08460 [cs.CL]

[16] Yiu-Chang Lin, Ankur Datta, and Giuseppe Di Fabbrizio. 2018. E-commerce Product Query Classification Using Implicit User's Feedback from Clicks. In *2018 IEEE International Conference on Big Data (Big Data)*. 1955–1959. https://doi.org/10.1109/BigData.2018.8622008

[17] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. arXiv:2107.13586 [cs.CL]

[18] Joanna Misztal-Radecka, Bipin Indurkhya, and Aleksander Smywiński-Pohl. 2021. Meta-User2Vec Model for Addressing the User and Item Cold-Start Problem in Recommender Systems. *User Modeling and User-Adapted Interaction* 31, 2 (apr 2021), 261–286. https://doi.org/10.1007/s11257-020-09282-4

[19] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. arXiv:2203.02155 [cs.CL]

[20] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

[21] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv:1910.10683 [cs.LG]

[22] Andy Rosenbaum, Saleh Soltan, Wael Hamza, Amir Saffari, Marco Damonte, and Isabel Groves. 2022. CLASP: Few-shot cross-lingual data augmentation for semantic parsing. In *AACL-IJCNLP 2022*. https://www.amazon.science/publications/clasp-few-shot-cross-lingual-data-augmentation-for-semantic-parsing

[23] Andy Rosenbaum, Saleh Soltan, Wael Hamza, Yannick Versley, and Markus Boese. 2022. LINGUIST: Language model instruction tuning to generate annotated utterances for intent classification and slot tagging. In *COLING 2022*. https://www.amazon.science/publications/linguist-language-model-instruction-tuning-to-generate-annotated-utterances-for-intent-classification-and-slot-tagging

[24] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. 2016. Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proc. IEEE* 104, 1 (2016), 148–175. https://doi.org/10.1109/JPROC.2015.2494218

[25] R. Mahesh K. Sinha and Anil Thakur. [n. d.]. Machine Translation of Bi-lingual Hindi-English (Hinglish) Text. In *Proceedings of Machine Translation Summit X: Papers*.

[26] Saleh Soltan, Shankar Ananthakrishnan, Jack FitzGerald, Rahul Gupta, Wael Hamza, Haidar Khan, Charith Peris, Stephen Rawls, Andy Rosenbaum, Anna Rumshisky, Chandana Satya Prakash, Mukund Sridhar, Fabian Triefenbach, Apurv Verma, Gokhan Tur, and Prem Natarajan. 2022. AlexaTM 20B: Few-Shot Learning Using a Large-Scale Multilingual Seq2Seq Model. arXiv:2208.01448 [cs.CL]

[27] Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. 2023. UL2: Unifying Language Learning Paradigms. arXiv:2205.05131 [cs.CL]

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. arXiv:1706.03762 [cs.CL]

[29] Yining Wang, Liwei Wang, Yuanzhi Li, Di He, Tie-Yan Liu, and Wei Chen. [n. d.]. A Theoretical Analysis of NDCG Type Ranking Measures.

[30] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned Language Models Are Zero-Shot Learners. arXiv:2109.01652 [cs.CL]

[31] Yiqing Wu, Ruobing Xie, Yongchun Zhu, Fuzhen Zhuang, Xu Zhang, Leyu Lin, and Qing He. 2023. Personalized Prompt for Sequential Recommendation. arXiv:2205.09666 [cs.IR]

[32] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. arXiv:1904.09675 [cs.CL]

[33] 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, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A Survey of Large Language Models. arXiv:2303.18223 [cs.CL]

[34] Ziwei Zhu, Yun He, Xing Zhao, and James Caverlee. 2021. Popularity Bias in Dynamic Recommendation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery amp; Data Mining* (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 2439–2449. https://doi.org/10.1145/3447548.3467376