Extreme Multi-label Query Classification for E-commerce

Giuseppe Di Fabbrizio[†], Evgeny Stepanov[†] and Filippo Tessaro[†]

Abstract

This paper addresses the challenge of extreme multi-label query classification (XMQC) in e-commerce, where short, ambiguous queries must be categorized into a vast label space to improve search relevance. We propose a supervised attention-based neural network framework that leverages clickstream data for automatic multi-label query annotation. Our approach employs a DistilBERT language model fine-tuned with a sparsemax loss function to effectively model the sparse category distribution for each query. Experiments on a real-world e-commerce dataset demonstrate that our model outperforms baseline approaches, achieving 78.96 precision@1, 73.75 recall@1, and 78.21 nDCG@1 overall. The sparsemax loss enables the model to handle label sparsity and ambiguity, with strong performance on head, torso, and tail queries. Qualitative analysis shows the model's robustness to challenges like misspellings and the ability to identify relevant categories for ambiguous queries.

Keywords

Query classification, Query understanding, Extreme multi-label text classification

1. Introduction

E-commerce platforms offer customers access to catalogs with millions of products categorized into taxonomies and thousands of hierarchically organized categories. Products are continuously updated through inventory feeds from partners and suppliers, including images, attributes, and other metadata utilized to populate the product detail pages. In such a dynamic environment, product search engines must continuously reindex products and optimize relevance through users' behavioral signals when available [1].

Both in conventional and semantic product search [2, 3], categorizing product search queries into single or multiple predefined target categories [4] can aid search engines in boosting relevance by passing the category to which the query belongs as a ranking signal and mitigating the cold start scenario when new products are added. Since an e-commerce catalog encompasses taxonomy trees with several thousand leaves, a query classification (QC) model must classify typically short and ambiguous text into a large label space [5].

This presents several challenges. Firstly, due to the large, unbalanced label set and ambiguity, a QC model must be framed as an extreme multi-label, multi-class classification problem in which a query can simultaneously belong to more than one non-mutually exclusive class. The model loss function must consider the multi-labels' sparseness and allow the model to return a sparse

eCom'24: ACM SIGIR Workshop on eCommerce, July 18, 2024, Washington, DC, USA

[†]Work done when at VUI, Inc.

difabbrizio@gmail.com (G. Di Fabbrizio); stepanov.evgeny.a@gmail.com (E. Stepanov); filippotessaro96@gmail.com (F. Tessaro)

https://difabbrizio.com/ (G. Di Fabbrizio)

posterior probability where some classes can have zero probability while others can represent a probability distribution among the sparse labels that sum to one.

Secondly, as we frame the task as a supervised learning problem, a substantial amount of high-quality annotated data is required. Since manual data annotation is expensive, time-consuming, and non-scalable for e-commerce traffic volumes, click-stream data that captures the user's behavioral signals is the only viable option for annotating data for a supervised model. However, labels need to be carefully denoised to represent query categories accurately.

Finally, data partitioning must respect the natural distribution across classes and labels. Conventional stratified sampling used in single-label classification tasks must be revised to consider stratified sampling in a multi-label scenario, where the sparse label distribution is preserved across training, validation, and test data.

To overcome these challenges, we propose a supervised attention-based neural network framework that leverages clickstream data to automatically annotate query search data using the user's behavioral signals in a multi-label fashion. To preserve the correct label distribution, we adopted a multi-label stratified sampling technique [6] that avoids the issues introduced using the traditional random distribution where the derived data subset may miss samples for rare labels, causing evaluation metrics problems. To model the multi-label sparseness, we adopt a *sparsemax* loss function [7] that optimizes the model to assign probability zero to most of the output predicted labels consistently with the users' clickstream data.

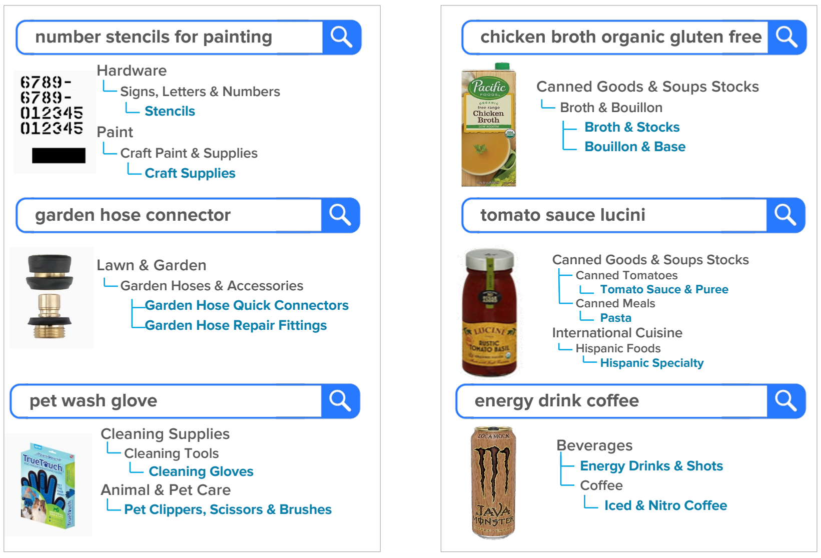
We demonstrate that the above process is significantly better than multiclass models, reporting robust generalization capabilities for torso and tail data. The resulting sparse category distribution can also capture ambiguous queries that apply to a broader range of categories and generic queries that may be better represented by intermediate nodes in the taxonomy tree.

The rest of the paper is organized as follows. Section 2 provides an overview of query classification in e-commerce. Section 3 formulates the extreme multi-label query classification (XMQC) problem. Section 4 discusses multi-label sampling and probability estimation techniques. Section 5 details the experimental setup, including dataset preprocessing, evaluation metrics, and model configuration. Section 6 presents the main results, analysis, and discussion. Section 7 reviews related work, and Section 8 concludes the paper and suggests future research directions.

2. Query classification

When searching for a particular item on an online retail platform, a user might input a search query briefly describing the product. Depending on how closely the search results align with the user's initial intent, the user may click on a relevant product, modify their query to refine the results, or leave the site if the displayed products are not accurately related to what was expected.

Manually classifying user queries into product categories is challenging and time-consuming. This difficulty arises from the complex interpretation of user intentions based on brief query texts and the large number of categories found in an e-commerce catalog, which can easily reach several thousand classes [8]. However, if a user selects a product immediately after receiving a list of products in response to their search query, the selected product category can be considered an accurate but sometimes noisy indication of the category associated with the query [9]. In practical terms, if multiple users use the same search query within a reasonable time frame (e.g.,



(a) Home improvement domain (b) Grocery domain

Figure 1: Examples of multi-label query classification for commercial e-commerce grocery and home improvement domains

30-90 days) and this generates at least several clicks on products with the same category label, then that selected category can be viewed as a valid label for that particular query.

Using behavioral signals like clicks, adding items to the cart, and completing purchases offers a practical method for automatically creating category labels. Annotating query classification datasets with behavioral signals suggests that a given query might interact with multiple taxonomy labels (e.g., product categories) to some extent. This interaction with various product categories could be viewed as a distribution of probabilities over the labels corresponding to a specific query. Approaching the issue as a standard multi-class problem allows each label to be treated independently from others, where they cannot coexist and have probabilities of zero or one in the training dataset. However, this approach fails to accurately represent real-world data by overlooking crucial insights into the ambiguous nature of search queries that can simultaneously pertain to multiple taxonomy category labels.

Figure 1 shows three product search query examples for home improvement and grocery domains from two large e-commerce organizations. For instance, in the home improvement domain, the query *"number stencils for painting"* would have relevant products in the category *Stencils* at the third level of the taxonomy tree, under *Signs, Letters & Numbers* and *Hardware*. A more realistic view of the problem should also consider the interaction with other labels in the taxonomy tree. Figure 1 also shows the categories that are selected less frequently but are still a legitimate category since number stencils can also be categorized as *Craft Supplies* under the broader *Paint* category.

Similarly, in the grocery domain, the query *"chicken broth organic gluten free"* may refer to products that are sharing the top two categories *Canned Goods & Soup Stocks / Broth & Bullions* but with relevant products in both leaf nodes *Broth & Stocks* and *Bouillon & Base*.

Yet, simply considering the presence of multiple labels is not sufficient to correctly represent a query classification prediction model. A given query q_i that has an interaction of 1% with label c_j

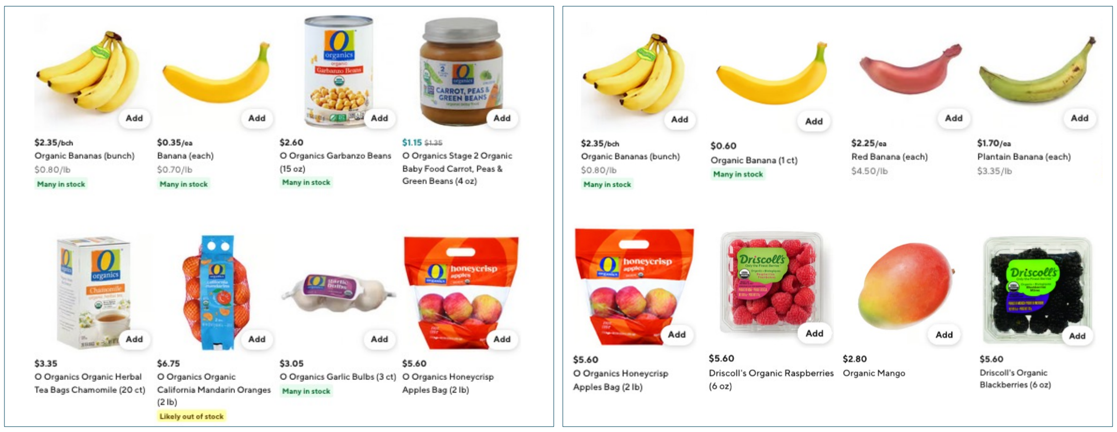

$2.35/bch
Organic Bananas (bunch)
$0.80/lb
Many in stock
$0.35/ea
Banana (each)
$0.70/lb
Many in stock
$2.60
O Organics Garbanzo Beans (15 oz)
Many in stock
$1.15 $1.35
O Organics Stage 2 Organic Baby Food Carrot, Peas & Green Beans (4 oz)
$3.35
O Organics Organic Herbal Tea Bags Chamomile (20 ct)
$6.75
O Organics Organic California Mandarin Oranges (2 lb)
Likely out of stock
$3.05
O Organics Garlic Bulbs (3 ct)
Many in stock
$5.60
O Organics Honeycrisp Apples Bag (2 lb)
$2.35/bch
Organic Bananas (bunch)
$0.80/lb
Many in stock
$0.60
Organic Banana (1 ct)
Many in stock
$2.25/ea
Red Banana (each)
$4.50/lb
$1.70/ea
Plantain Banana (each)
$3.35/lb
$5.60
O Organics Honeycrisp Apples Bag (2 lb)
$5.60
Driscoll's Organic Raspberries (6 oz)
$2.80
Organic Mango
$5.60
Driscoll's Organic Blackberries (6 oz)
Add

Figure 2: Search result examples for the query *"organic bananas"* for a commercial e-commerce search engine in the grocery domain with (right) and without (left) query understanding.

and 99% with label c_k would be considered in the same way a query $q_{i'}$ that has 99% interaction with label c_j and 1% interaction with label c_k, producing a skewed prediction where the minority label c_k could take precedence on the more popular usage of the query. This is particularly important when the predicted query labels are used as input features to optimize (or re-rank) a search result returning matching products from a catalog. Considering the first example in Figure 1, a search engine could return a majority of products from the *Craft Supplies* minority class rather than boosting results from the *Stencils* category, compromising the actual result relevance and potentially missing product conversion opportunities.

The image in Figure 2 illustrates the potential benefits of integrating query understanding capabilities, such as query classification, into an e-commerce search engine. By accurately classifying the intent behind the query *"organic bananas"*, the search engine can significantly improve the relevance of the results.

Without query understanding (left side), the search engine relies primarily on lexical matching, focusing on the individual terms *"organic"* and *"bananas"* independently. This leads to the retrieval of less relevant items such as organic baby food, tea, and onions, which happen to contain the word "organic" but are not related to the core intent of the query. The lack of understanding of the query's true meaning results in a suboptimal user experience.

In contrast, when query understanding is integrated (right side), the search engine can classify the query *"organic bananas"* as belonging to the category of fruits. This deeper understanding allows the engine to prioritize and boost categories and products directly related to organic bananas, such as fresh bananas, plantains, mangoes, and other fruits. By leveraging the query classification, the search engine can assign higher relevance scores to these categories and products, pushing them to the top of the search results.

Besides query classification in the e-commerce domain, other domains have similar challenges. For example, movies can have more than one genre label, and each label can also contribute a different weight to the overall movie genre. Negative online behaviors classification, which has been recently getting attention to improve social media and online content quality [10], is also considered a multi-label problem since toxic comments can have different labels at the same time. The main difference with the e-commerce domain is that e-commerce is also considered an extreme classification task due to the number of labels that often reach several thousand.

3. Extreme multi-label query classification

Extreme multi-label query classification (XMQC) tries to find the most relevant subset of class labels associated with a short query text from an extremely large number of categories.

In the XMQC problem for e-commerce, we are given a set of n queries $\mathcal{Q} = \{q_1, q_2, \ldots, q_n\}$, where each query q_i is represented by a D-dimensional feature vector $\mathbf{x}_i \in \mathbb{R}^D$. The feature vector $\mathbf{x}_i$ can encode various attributes of the query, such as the textual content, user context, etc. Additionally, we have an extremely large set of L product categories $\mathcal{C} = \{c_1, c_2, \ldots, c_L\}$, where typically $L \gg D$ and can be in the order of hundreds of thousands or millions in e-commerce applications [11]. Each query q_i is associated with a binary label vector $\mathbf{y}_i \in \{0,1\}^L$, where $y_{ij} = 1$ if query q_i belongs to category c_j, and $y_{ij} = 0$ otherwise. A query can simultaneously belong to multiple categories, i.e., $\sum_{j=1}^{L} y_{ij} \geq 1$.

The objective is to learn a classifier $f : \mathbb{R}^D \to \{0,1\}^L$ that accurately maps each query feature vector $\mathbf{x}_i$ to its corresponding label vector $\hat{\mathbf{y}}_i$, i.e., $f(\mathbf{x}_i) = \hat{\mathbf{y}}_i \approx \mathbf{y}_i$.

However, instead of predicting the binary label vector directly, we aim to predict a sparse label distribution $\mathbf{p}_i \in \Delta^{L-1}$, where $\Delta^{L-1} := \{\mathbf{p} \in \mathbb{R}^L | \mathbf{1}^T \mathbf{p} = 1, \mathbf{p} \geq \mathbf{0}\}$ is the $L-1$ dimensional simplex. Each entry p_{ij} represents the probability of query q_i belonging to category c_j.

The $L-1$ dimensional simplex is a geometric object that represents the set of all probability distributions over L discrete categories. Using the simplex is necessary for representing sparse label distributions, capturing label uncertainty, enabling the use of appropriate loss functions, and facilitating the application of label embedding techniques to handle the extremely large number of categories in e-commerce applications.

The objective is to learn a classifier $f : \mathbb{R}^D \to \Delta^{L-1}$ that maps each query feature vector $\mathbf{x}_i$ to its corresponding sparse label distribution $\hat{\mathbf{p}}_i$, i.e., $f(\mathbf{x}_i) = \hat{\mathbf{p}}_i \approx \mathbf{p}_i$. This can be achieved by minimizing a loss function that measures the discrepancy between the predicted label distribution $\hat{\mathbf{p}}_i$ and the ground truth distribution $\mathbf{p}_i$, e.g., using KL divergence or sparsemax loss [7].

The learned classifier f will predict the relevant categories for a given query q_i, taking into account the label distribution's sparsity. The label embedding approach helps reduce the computational complexity and mitigate the data sparsity issue associated with the extremely large label space.

3.1. Extreme multi-label learning

To address the challenges of extreme multi-label query classification, we propose a distillation-based approach using a pre-trained DistilBERT model [12] as the student model and the *sparsemax loss* function [7] for fine-tuning.

DistilBERT is a distilled version of BERT [13] that retains 97% of BERT's performance while being 40% smaller and 60% faster at inference time. The key idea is to leverage knowledge distillation during the pre-training phase to learn a compact model that can be fine-tuned for downstream tasks. By using the pre-trained DistilBERT model, we can take advantage of the knowledge already distilled into the model during its pre-training phase.

In our approach, we first fine-tune the pre-trained DistilBERT model on the query-category pairs using the sparsemax loss function. The sparsemax loss is a sparse alternative to the *softmax* loss that encourages the model to predict sparse probability distributions. The sparsemax loss is defined as:

$$L_{\text{sparsemax}}(z;k) = -z_k + \frac{1}{2}\sum_{j\in S(z)} (z_j^2 - \tau^2(z)) + \frac{1}{2}$$

where z is the input vector, k is the target label, $S(z)$ is the support set of sparsemax(z), and $\tau(z)$ is the threshold function given by:

$$\tau(z) = \frac{\sum_{j\leq k(z)} z_{(j)} - 1}{k(z)}$$

with $z_{(1)} \geq z_{(2)} \geq ... \geq z_{(K)}$ being the sorted coordinates of z, and $k(z) = \max\{k \in [K] \mid 1 + kz_{(k)} > \sum_{j\leq k} z_{(j)}\}$.

The sparsemax loss has several desirable properties, including convexity, differentiability everywhere, and a connection to the Huber classification loss in the binary case [14]. It can be used as a loss function for training multi-label linear classifiers and in attention-based neural networks.

During fine-tuning, we minimize the sparsemax loss between the predicted label distribution and the ground truth sparse label distribution. This allows the model to learn to predict sparse label distributions that align with the true label sparsity in the XMQC task.

During inference, we apply the sparsemax transformation to the output logits of the fine-tuned DistilBERT model to obtain a sparse label distribution over the categories. We then select the categories with non-zero probabilities as the predicted labels for the query.

By leveraging the pre-trained DistilBERT model and fine-tuning it with the sparsemax loss, our approach can effectively handle the extreme multi-label classification problem while being computationally efficient. The use of a pre-trained model allows us to benefit from the knowledge already distilled into the model, while the sparsemax loss encourages the prediction of sparse label distributions, which is crucial for XMQC.

4. Multi-label sampling and probability estimation

Sechidis et al. [6] discusses two approaches for stratified sampling in multi-label data: a) stratified sampling based on distinct labelsets, and b) iterative stratification. The latter is particularly relevant XMQC where the number of distinct label sets is very large compared to the number of examples.

Iterative stratification aims to ensure that the ratio of positive to negative examples for each label is approximately maintained in each subset of the data. It does this by greedily assigning examples to subsets based on the desired number of positive examples for each label in each subset.

More formally, let D be the full dataset, $L = \{c_1, ..., c_q\}$ the set of labels, $S_1, ..., S_k$ the desired subsets, and $r_1, ..., r_k$ the desired proportion of examples in each subset. The algorithm calculates the desired number of examples in each subset S_j as $n_j = |D|r_j$, and the desired number of positive examples for each label c_i in subset S_j as $n_j^i = |D^i| r_j$, where D^i is the subset of D containing positive examples of label c_i.

The algorithm then iteratively assigns examples to subsets. In each iteration, it considers the label c_i with the fewest remaining positive examples, and for each example of this label, assigns it to the subset S_j that maximizes the current desired number of positive examples for this label n_j^i.

The Labels Distribution (LD) measure is used to evaluate how well the ratio of positive to negative examples for each label is maintained in each subset compared to the full data. It is defined as:

$$LD = \frac{1}{q}\sum_{i=1}^{q}\left(\frac{1}{k}\sum_{j=1}^{k}\left|\frac{|S_j^i|}{|S_j|-|S_j^i|}-\frac{|D^i|}{|D|-|D^i|}\right|\right)$$

where S_j^i is the subset of S_j containing positive examples of label c_i.

The paper empirically shows that iterative stratification achieves lower LD (i.e., better label distribution) compared to stratification based on distinct label sets, especially when the ratio of distinct label sets to examples is large. This makes it particularly suitable for XMQC problems.

4.1. Estimating Label Probabilities from Click Data

In the XMQC problem, each query q_i is associated with a sparse label distribution $\mathbf{p_i} \in \Delta^{L-1}$, where p_{ij} represents the probability of query q_i belonging to category c_j. To estimate these probabilities, we leverage the click data associated with each query-category pair.

Let w_{ij} be the number of clicks received by category c_j starting from query q_i, and let $w_i = \sum_{j=1}^{L} w_{ij}$ be the total number of clicks starting from query q_i across all categories. We estimate the probability p_{ij} as: $p_{ij} = \frac{w_{ij}}{w_i}$.

For example, consider the query q_i = *"wood for crafts"* in Table 1. The total number of clicks for this query is $w_i = 197$. The category *Outdoors/Outdoor Games & Toys/Kids Tools & Building Kits/Toy Miniatures* received $w_{ij} = 99$ clicks, resulting in an estimated probability of $p_{ij} = \frac{99}{197} = 0.502538$. Similarly, the category *Paint/Craft Paint & Supplies/Craft Supplies* received $w_{ij} = 49$ clicks, leading to an estimated probability of $p_{ij} = \frac{49}{197} = 0.248731$.

This approach allows us to obtain a distribution over labels for each query, rather than relying on one-hot label vectors. The estimated probabilities p_{ij} capture the strength of association between query q_i and category c_j based on user click behavior.

To reduce label noise and ensure consistency, we apply a heuristic-based filtering approach. We remove labels with $p_{ij} < 0.1$, as the clicks for these labels are too sparse and may not reliably represent the query's intent. Similarly, we remove queries with $\max_j p_{ij} < 0.4$, as these queries lack a consistent category association and may introduce noise into the training process.

By estimating the label probabilities based on click data and applying noise reduction techniques, we obtain a sparse label distribution $\mathbf{p}_i$ for each query q_i. This distribution provides a more informative representation of the query's category associations compared to binary label vectors, capturing the uncertainty and multiplicity inherent in the XMQC task.

The sparse label distributions $\mathbf{p}_i$ serve as the target probabilities for training the DistilBERT model using the sparsemax loss, as described in the previous section. By learning to predict these sparse distributions, the model can effectively handle the extreme multi-label nature of the problem and provide more accurate category recommendations for queries.

w_{ij}	p_{ij}	Category c_j
99	0.502538	Outdoors/Outdoor Games & Toys/Kids Tools & Building Kits/Toy Miniatures
49	0.248731	Paint/Craft Paint & Supplies/Craft Supplies
16	0.081218	Building Supplies/Lumber & Composites/Appearance Boards
10	0.050761	Building Supplies/Decking/Deck Board Samples
8	0.040609	Flooring/Tile & Tile Accessories/Tile
6	0.030457	Home Decor/Furniture/Furniture Parts/Table Tops
3	0.015228	Home Decor/Wall Art & Decor/Wall Art
2	0.010152	Storage & Organization/Shelves & Shelving/Wall Mounted Shelving
2	0.010152	Building Supplies/Lumber & Composites/Plywood & Sheathing/Plywood
1	0.005076	Moulding & Millwork/Moulding/Crown Moulding
1	0.005076	Moulding & Millwork/Moulding/Window & Door Trim/Window & Door Moulding

Table 1
Category click counts (w_{ij}), estimated probabilities (p_{ij}), and category labels (c_j) for the query q_i = *"wood for crafts"*. The total number of clicks for this query is $w_i = 197$.

5. Experimental setup

5.1. Dataset Overview

The data used for the experiments was collected from the clickstream of an online store of a large home improvement retail corporation. The dataset consists of query-category pairs, where the labels are inferred based on the frequency of user clicks on products belonging to specific taxonomy categories. In total, there are 4,462 categories in the taxonomy tree with different levels of nesting.

Each data instance consists of three elements: 1) the search query typed by the user q_i, 2) the label referring to the product category related to the query c_j, and 3) the number of clicks w_{ij} associated with the query-category pair in a specific time interval. Since a query can lead to clicks on products from different categories, queries are duplicated per label, and the total number of clicks for a query $w_i = \sum_{j=1}^{L} w_{ij}$ is also computed, where L is the number of categories for the query q_i (see also example in Table 1).

Following the same approach as described in [15], the total number of clicks for a query is used to categorize queries into head, torso, and tail distributions. Queries with a total click count equal to or higher than 100 are considered head queries, while queries with a click count of one are considered tail queries. All queries in between are classified as torso queries.

Table 2 provides a description of the dataset in terms of query and label counts, as well as their distribution among head, torso, and tail queries. When considering the entire dataset, the majority of unique queries (86.4%) are associated with a single label. However, the distribution of single-label and multi-label queries varies significantly when examining head, torso, and tail queries separately.

In e-commerce data, tail queries often form a long tail distribution. By definition, tail queries have a click count of 1 and are, therefore, all single-label. For torso queries, the percentage of single-label queries drops to 66.2%, and for head queries, it further decreases to 8.5%.

Head queries represent 69% of total traffic, with multi-label head queries accounting for 67.9%. Combined, multi-label head and torso queries comprise 81.1% of total clicks (112,742,307), indicating their high relevance for conversion rates. Tail queries, on the other hand, are known to be

	Total	Head	Torso	Tail
Total clicks (sum of w_i)	138,965,331	96,233,946	33,705,963	9,025,422
Total clicks (%)	100.0%	69.3%	24.3%	6.5%
Total clicks for multi-label queries	112,742,307	94,425,519	18,316,788	0
Total clicks for multi-label queries (%)	81.1%	67.9%	13.2%	0.0%
Unique queries (q_i)	14,841,471	102,619	5,713,405	9,025,422
Unique query-category pairs	18,234,809	647,811	8,559,713	9,025,422
Single-label queries	12,818,844	8,748	3,784,674	9,025,422
Single-label queries (%)	86.4%	8.5%	66.2%	100%
Unique categories (out of 4,462)	3,121	3,102	3,115	3,109
Categories with 50+ queries	2,992	2,057	2,865	2,869

Table 2
Query and category counts for the whole dataset and head, torso, and tail queries.

challenging to classify due to their low frequency. Text preprocessing techniques are often applied to tail queries to reduce their variability and improve classification performance (see Section 5.2.1).

The query classification model is intended for the classification of unseen data. From Table 2, we observe that only 3,121 out of the total 4,462 categories are represented in the dataset. The number of well-represented categories, arbitrarily defined as those with more than 50 unique queries, is even lower. To increase the number of well-represented categories and mitigate data sparsity, we apply label pruning and aggregate queries on parent categories, as described in the following sections.

5.2. Data preprocessing, pruning, and partitioning

5.2.1. Preprocessing

Product search queries often contain various extraneous characters and information that may not be directly relevant to the classification task. To reduce data noise and space dimensionality, it is beneficial to apply several preprocessing and normalization steps to the query data. These steps typically include measurements normalization, punctuation normalization and removal, non-ASCII characters removal, mixed alphanumeric tokens replacement, numeric tokens replacement, and lowercasing. The specific preprocessing steps and their order have been calibrated for the home improvement domain, where measurements, quantities, and specific SKU numbers are frequently used in search queries. The preprocessing pipeline has been designed and fine-tuned based on empirical analysis and domain knowledge to ensure optimal results for this particular domain. After preprocessing, the cleaned and normalized query data is ready to be used as input to the DistilBERT model for training and inference.

5.2.2. Noise Removal

Search engine queries are not always directly related to product searches, and consequently, they may not correspond to any specific product category. To remove frequent non-product queries and improve the quality of the training data, we apply two threshold-based filtering techniques: label removal threshold and query removal threshold.

For multi-label queries, such as the example shown in Table 1, we remove all category labels with an estimated probability p_{ij} below the *label removal threshold* (t_l). This step helps to

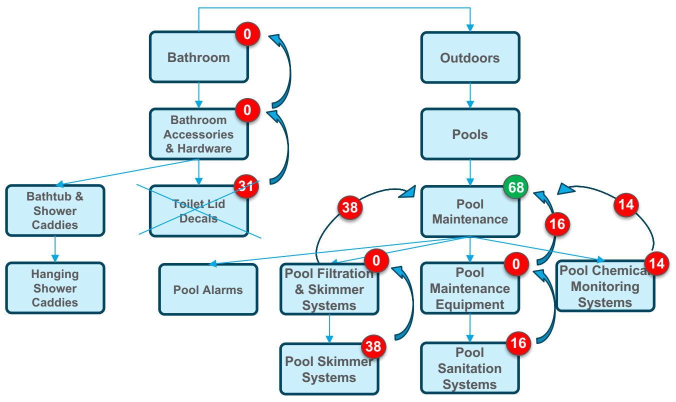


Figure 3: Reduce sparseness: Taxonomy tree pruning/merging

eliminate labels that have a weak association with the query based on user click behavior.

If none of the remaining category labels for a given query has an estimated probability p_{ij} higher than or equal to the *query removal threshold* (t_q), the entire query is removed from the dataset. This step helps to filter out queries that lack a strong association with any product category, even after removing weakly associated labels. The values of the label removal threshold and query removal threshold were determined through experimental tuning, with t_l set to 0.1 and t_q set to 0.4. These thresholds were found to effectively remove non-product queries while retaining relevant query-category pairs.

Referring back to the example in Table 1, the noise removal procedure would remove the last two rows, corresponding to the categories *Moulding & Millwork/Moulding/Crown Moulding* and *Moulding & Millwork/Moulding/Window & Door Trim/Window & Door Moulding*, as their estimated probabilities p_{ij} are below the label removal threshold of 0.1. However, the query *"wood for crafts'*' would be retained in the dataset because it has at least one category label (*Outdoors/Outdoor Games & Toys/Kids Tools & Building Kits/Toy Miniatures*) with an estimated probability above the query removal threshold of 0.4.

The noise removal procedure effectively filters out non-product queries, such as URLs, frequently asked questions, commands, and chitchat (e.g., *"we are," "sign out," "change store'*').

5.2.3. Labels pruning

To reduce data sparsity for the category labels with less frequent clicks, a large catalog taxonomy tree can be pruned to increase the density of less frequent queries. Labels with less than K-tagged examples (e.g., K=50) can be merged with the upper taxonomy node and their labels are replaced with the upper-level taxonomy label (see Figure 3). For each label in the taxonomy tree, the number of examples per node is tracked to capture the real queries distribution. After applying the pruning procedure, every leaf in the taxonomy tree will include at least K samples [4].

In the example illustrated in Figure 3, the taxonomy tree is pruned based on a threshold of K=50 samples per node. The *Pool Chemical Monitoring Systems* category, which has only 14 samples, is merged with its parent node *Pool Maintenance.* The *Pool Skimmer Systems* category, having 38 samples, is first merged with its sibling node *Pool Filtration & Skimmer Systems,* which is initially empty because it is not a leaf of the tree. The combined category now has 38 samples and is further merged

	Total	Head	Torso	Tail
Total clicks (sum of w_i)	129,130,839	87,715,635	32,395,413	9,019,791
Total clicks (%)	100.0%	67.9%	25.1%	7.0%
Total clicks for multi-label queries	38,198,328	26,511,025	11,687,303	0
Total clicks for multi-label queries (%)	29.6%	20.5%	9.1%	0.0%
Unique queries (q_i)	14,719,934	95,505	5,604,638	9,019,791
Unique query-category pairs	16,699,427	133,475	7,546,161	9,019,791
Single-label queries	13,030,572	64,234	3,946,547	9,019,791
Single-label queries (%)	88.5%	67.3%	70.4%	100%
Unique categories (out of 4,462)	2,964	2,774	2,964	2,964
Categories with 50+ queries	2,964	656	2,795	2,869

Table 3
Query and category counts for the whole dataset and head, torso, and tail queries after the application of the pre-processing and pruning.

with the parent node *Pool Maintenance.* The *Pool Sanitation Systems* category, which has only 16 samples, is also merged with *Pool Maintenance.* After these merging steps, the *Pool Maintenance* category accumulates a total of 68 samples, surpassing the required threshold of 50 samples per node. On the other hand, the *Toilet Lid Decals* category, which has only 31 samples, is merged with its parent node *Bathroom Accessories & Hardware.* However, even after merging, the combined category still does not reach the threshold of 50 samples. As a result, the *Toilet Lid Decals* category is eliminated from the pruned taxonomy tree. After applying the pruning procedure, every leaf in the taxonomy tree will include at least K samples, ensuring a minimum density of examples per category label. Categories that fail to meet the threshold after merging with their parent nodes are removed from the tree. This process helps to mitigate the issue of data sparsity for less frequent queries and enables more effective training and prediction in the extreme multi-label classification setting.

5.2.4. Data split procedure

After preprocessing and pruning, the data is split into training, development, and test folds using a K-fold stratified partitioning procedure for multi-label data as described in [6] where the number of folds could be, for instance, three with a large training set (95%) and two smaller testing (2.5%) and development sets (2.5%). The iterative stratified splitting procedure described in [6] has been adapted to accommodate frequency-weighted samples. Query weights can be derived from the click data associated with each query-category pair as described in Section 4.1.

The adapted procedure allows weights to be used instead of raw query counts to compute the fold label requirements. Since a query can have multiple labels, each of these is multiplied by the query weight and added to the total label count. During the data splitting, the query weights are deducted from the fold label requirement values. The procedure ensures that the queries with higher weights are distributed first, thus maintaining the distribution of head/torso/tail queries across the folds. As a result, the data is split such that it keeps the folds disjoint in terms of samples while maintaining the same label distribution. This approach helps to mitigate the issue of missing labels or underrepresented classes that can occur when using the more traditional random sampling process to split data folds by taking into account the importance of each query-category pair based on user click behavior.

The data description after the application of the preparation steps is presented in Table 3.

Notable changes include the overall decrease in the number of head queries and a significant increase in single-label head queries (from 8.6% to 67.3%). There is also a reduction in the number of multi-label queries and a decrease in the number of categories with 50+ unique queries among head queries, indicating shifts in category popularity. For the experimentation, the data was split into training (95%), development (2.5%), and testing (2.5%) folds.

5.3. Evaluation Metrics

We employ several evaluation metrics to assess the performance of our proposed XMQC approach, considering various aspects such as precision, recall, and ranking quality[1]. Let $\mathbf{y}_i \in \{0,1\}^L$ be the ground-truth label vector and $\hat{\mathbf{p}}_i \in \mathbb{R}^L$ be the predicted score vector for query q_i.

Precision at K (P@K) measures the proportion of relevant categories among the top K predicted categories, while Recall at K (R@K) measures the proportion of relevant categories found in the top K predictions. They are calculated as follows:

$$\text{P@K} = \frac{1}{K} \sum_{l \in \text{topK}(\hat{\mathbf{p}}_i)} y_{il} \qquad \text{R@K} = \frac{1}{|Y_i|} \sum_{l \in \text{top-K}(\hat{\mathbf{p}}_i)} y_{il}$$

where top-K$(\hat{\mathbf{p}}_i)$ represents the set of top K predicted categories, y_{il} is the ground truth label, and Y_i is the set of true categories for query q_i.

Normalized Discounted Cumulative Gain (nDCG@K) is a ranking-based metric that considers the position of relevant categories in the top K recommendations. It is calculated as:

$$\text{nDCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}} \qquad \text{DCG@K} = \sum_{i=1}^{K} \frac{2^{y_{il}} - 1}{\log_2(i+1)}$$

where DCG@K is the Discounted Cumulative Gain at K, and IDCG@K is the Ideal Discounted Cumulative Gain at K, representing the maximum possible DCG@K score. These metrics provide a comprehensive evaluation of the proposed XMQC approach, assessing its ability to recommend relevant categories, considering precision, recall, and ranking quality.

5.4. Model Configuration

We trained our proposed model using the DistilBERT pre-trained language model [12]. We added a multi-layer perceptron (MLP) layer with 512 units, ELU activation function, and L2 regularization, which acts as a pre-classifier. A dropout layer with a dropout rate of 0.4 was also included to prevent overfitting.

The model was trained for 18 epochs using an Adam optimizer with a learning rate of 1e-5 and an epsilon value of 1e-8. We employed the sparsemax loss function, which is a variation of the softmax function that encourages sparse output distributions [7]. This is particularly suitable for the XMQC task, where the model needs to predict a small subset of relevant categories for each query.

We also trained two baseline models for comparison: a DistilBERT model with the cross-entropy loss function and a fastText model. FastText is a lightweight text classification model that represents documents as an average of their word embeddings and uses a linear classifier [16]. It serves as a simple and efficient baseline for the XMQC task.

[1]https://en.wikipedia.org/wiki/Evaluation_measures_(information_retrieval)

6. Main Results and Examples

Table 4 presents the evaluation results comparing our proposed DistilBERT model with sparsemax loss function against the DistilBERT and fastText models using the cross-entropy loss function. Our model consistently outperforms the baselines across all evaluation metrics, achieving 78.96 precision@1, 73.75 recall@1, and 78.21 nDCG@1 overall; demonstrating the sparsemax loss's effectiveness in capturing the XMQC task's sparse nature.

Model	Metric@K	1	2	3	4	5
SparseMax	Precision	78.96	46.74	32.74	25.07	20.27
	Recall	73.75	84.82	88.32	89.83	90.64
	nDCG	78.21	82.88	84.5	85.14	85.45
DistilBERT	Precision	66.50	34.75	23.19	17.4	13.92
	Recall	66.71	69.74	69.83	69.85	69.85
	nDCG	66.5	67.29	66.91	66.7	66.85
fastText	Precision	50.98	25.49	16.99	12.75	10.20
	Recall	47.59	47.59	47.59	47.59	47.59
	nDCG	50.13	48.65	48.60	48.60	48.61

Table 4
Evaluation results comparing the DistilBERT model with SparseMax loss function and DistilBERT and fastText models using cross-entropy loss function

To further analyze our model's performance, we evaluated it on different parts of the query distribution: head, torso, and tail. Table 5 shows the comparison of metrics for these query subsets. The model achieves higher precision, recall, and nDCG scores for head queries compared to torso and tail queries. This is expected, as head queries have better query-category relevance.

However, our model still performs well on torso and tail queries, despite their lower query-category relevance due to less customer behavior information for identifying relevant categories. This can be attributed to the sparsemax loss function, which helps the model to focus on the most relevant categories for each query, even when the number of relevant categories is small. By encouraging sparsity in the output distribution, the sparsemax loss enables the model to handle the ambiguity and uncertainty present in torso and tail queries more effectively than the cross-entropy loss.

These results support our initial hypothesis that a sparse label model better addresses the query classification ambiguity in the head and torso distribution. By inducing sparsity in the label space, the sparsemax loss allows the model to make more confident predictions for queries with multiple relevant categories. Furthermore, the model's ability to focus on the most relevant category for tail queries, where there is typically only one relevant category per query, highlights the benefits of the sparse labeling approach in handling the long-tail distribution of queries in e-commerce search.

6.1. Analysis and Discussion

The multi-label model could also help to identify specific use cases where the predicted posterior sparse probabilities provide further insights about the user's intent. For instance, table 6 shows the query *"leona silver"*, which is a misspelling of the brand *"Leonia Silver"*. Despite the

Query distribution	Metric@K	1	2	3	4	5
Head	Precision	97.89	62.63	43.86	33.42	26.87
	Recall	80.97	94.09	96.55	97.39	97.62
	nDCG	95.09	96.24	96.69	96.94	97.01
Torso	Precision	88.52	54.40	38.50	29.58	23.94
	Recall	75.10	86.55	90.00	91.42	92.13
	nDCG	86.59	87.85	89.24	89.83	90.11
Tail	Precision	72.84	41.83	29.06	22.19	17.93
	Recall	72.84	83.66	87.19	88.77	89.64
	nDCG	72.84	79.67	81.43	82.11	82.45

Table 5
DistilBERT model performance with SparseMax loss function for different query distribution parts

Query	Label	Prob
leona silver	3/Flooring/Tile & Tile Accessories/Tile	0.94
	3/Flooring/Tile & Tile Accessories/Tile Samples	0.03
	3/Flooring/Tile & Tile Accessories/Accent & Trim Tile	0.03
brrom	3/Cleaning Supplies/Cleaning Tools/Brooms	0.74
	3/Automotive/Automotive Accessories/Ice Scrapers	0.10
	4/Cleaning Supplies/Cleaning Tools/Cleaning Brushes/Tile & Grout Brushes	0.06
616295	3/Home Decor/Wall Art & Decor/Wall Art	0.02
	3/Flooring/Tile & Tile Accessories/Tile	0.02
	3/Lighting & Ceiling Fans/Ceiling Fans & Accessories/Ceiling Fans	0.01
lighting for ceiling	3/Lighting & Ceiling Fans/Ceiling Lights/Flush Mount Lighting	0.6
	3/Lighting & Ceiling Fans/Ceiling Lights/Pendant Lighting	0.35
	3/Lighting & Ceiling Fans/Ceiling Lights/Chandeliers	0.02
	3/Lighting & Ceiling Fans/Lighting Parts & Accessories/Ceiling Light Mounts	0.02
	4/Lighting & Ceiling Fans/Ceiling Lights/Recessed Lighting/Recessed Downlights	0.01

Table 6
Examples of multi-label query classification.

spelling error, the model correctly identifies the most relevant categories, such as *Flooring/Tile & Tile Accessories/Tile* and *Flooring/Tile & Tile Accessories/Tile Samples*, with high probabilities. Similarly, for the misspelled query *"brrom"*. This demonstrates the robustness of the sparse label model in handling misspellings and variations in brand names.

For the user query specified as product SKU number, *"616295"*, the model assigns relatively uniform probabilities to multiple categories across the taxonomy, indicating that the SKU number does not provide enough semantic information to confidently predict a specific category.

For the query *"lighting for ceiling"*, the model assigns the highest probabilities to categories related to ceiling lighting, such as*Lighting & Ceiling Fans/Ceiling Lights/Flush Mount Lighting* and *Lighting & Ceiling Fans/Ceiling Lights/Pendant Lighting*.

These examples showcase the strengths of our proposed approach in handling diverse query types, including misspellings, SKU numbers, and queries that map to generic taxonomy nodes. However, there are still opportunities for improvement. For queries lacking clear semantic information, such as SKU numbers, incorporating additional context from user sessions or product metadata could potentially enhance the classification performance. Additionally, the model's ability to handle extremely rare or unseen queries can be further investigated and improved through techniques like few-shot learning or data augmentation.

7. Related work

Recent work on query classification in e-commerce has focused on leveraging hierarchical category structures to improve representation learning and address challenges such as class imbalance and query ambiguity. HCL4QC [17] introduces hierarchical loss functions to adjust category representations and ensure semantic consistency, while [18] presents a framework utilizing enhanced representation learning and neighborhood-aware sampling to improve classification accuracy.

Ahmadvand et al. [15] propose DeepCAT, a deep learning model that learns joint word-category representations to enhance query understanding, particularly for minority classes and tail queries in e-commerce search. Their approach incorporates category-category co-occurrences through a novel loss function.

In contrast, our work emphasizes the multi-label aspect of query classification, recognizing that a product may belong to multiple categories simultaneously. We employ a sparse label approach using the sparsemax loss to effectively handle query ambiguity and data sparsity in XMQC.

Although large language models (LLMs) are seen as the future of e-commerce search [19], traditional deep learning methods remain crucial for scalability and low latency. While LLMs excel at leveraging pre-training knowledge, encoding structured data like product catalogs and taxonomies as natural language text may lead to efficiency and latency challenges at large scales. Until LLM optimization challenges are fully resolved, traditional deep learning architectures tailored for query classification and understanding tasks are likely to remain core components for scalable, low-latency query processing in e-commerce search.

8. Conclusions

In this paper, we presented a novel approach for extreme multi-label query classification in e-commerce using an attention-based neural network optimized with a sparsemax loss function. Our experiments on a large-scale e-commerce dataset validated the effectiveness of modeling queries as a sparse distribution over product categories. The proposed DistilBERT model with sparsemax loss outperformed baseline classifiers, demonstrating significant improvements in precision, recall, and nDCG across different query types. The sparsemax loss proved crucial for handling the sparsity and ambiguity inherent in XMQC, enabling the model to focus on the most relevant categories for each query. Notably, our approach showed strong performance on both frequent head queries and less common torso/tail queries, underlining its ability to capture meaningful category associations even with limited training data. Qualitative analysis highlighted the model's robustness to real-world challenges such as misspellings and its capability to identify pertinent categories for ambiguous or broad queries. These findings underscore the practical value of our XMQC framework in enhancing query understanding and search relevance in e-commerce. Future research could explore incorporating additional context, addressing data sparsity through techniques like few-shot learning, and optimizing inference efficiency for real-time query processing.

Acknowledgments

We want to thank Amirhossein Tebbifakhr for the extensive data analysis and contribution to modeling ideas, Danilo Giulianelli real-time optimizations in the production system, and Federico Marinelli for an early proof of concept.

References

[1] S. Jiang, Y. Hu, C. Kang, T. Daly, D. Yin, Y. Chang, C. Zhai, Learning query and document relevance from a web-scale click graph, in: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 185–194. URL: https://doi.org/10.1145/2911451.2911531. doi:10.1145/2911451.2911531.

[2] J. K. Bergum, Redefining Hybrid Search Possibilities with Vespa, 2024. URL: https://blog.vespa.ai/redefining-hybrid-search-possibilities-with-vespa/.

[3] P. Nigam, Y. Song, V. Mohan, V. Lakshman, W. A. Ding, A. Shingavi, C. H. Teo, H. Gu, B. Yin, Semantic product search, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 2876–2885. URL: https://doi.org/10.1145/3292500.3330759. doi:10.1145/3292500.3330759.

[4] Y.-C. Lin, A. Datta, G. Di Fabbrizio, E-commerce product query classification using implicit user's feedback from clicks, in: 2018 IEEE International Conference on Big Data (Big Data), 2018, pp. 1955–1959. doi:10.1109/BigData.2018.8622008.

[5] J.-W. Ha, H. Pyo, J. Kim, Large-scale item categorization in e-commerce using multiple recurrent neural networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 107–115. URL: https://doi.org/10.1145/2939672.2939678. doi:10.1145/2939672.2939678.

[6] K. Sechidis, G. Tsoumakas, I. Vlahavas, On the stratification of multi-label data, in: D. Gunopulos, T. Hofmann, D. Malerba, M. Vazirgiannis (Eds.), Machine Learning and Knowledge Discovery in Databases, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 145–158.

[7] A. F. T. Martins, R. F. Astudillo, From softmax to sparsemax: a sparse model of attention and multi-label classification, in: Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, ICML'16, JMLR.org, 2016, p. 1614–1623.

[8] D. Shen, Y. Li, X. Li, D. Zhou, Product query classification, in: Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM '09, Association for Computing Machinery, New York, NY, USA, 2009, p. 741–750. URL: https://doi.org/10.1145/1645953.1646047. doi:10.1145/1645953.1646047.

[9] K. Bi, C. H. Teo, Y. Dattatreya, V. Mohan, W. B. Croft, Leverage implicit feedback for context-aware product search, in: J. Degenhardt, S. Kallumadi, U. Porwal, A. Trotman (Eds.), Proceedings of the SIGIR 2019 Workshop on eCommerce, co-located with the 42st International ACM SIGIR Conference on Research and Development in Information

Retrieval, eCom@SIGIR 2019, Paris, France, July 25, 2019, volume 2410 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2019, p. n. pag. URL: https://ceur-ws.org/Vol-2410/paper15.pdf.

[10] B. D. Dirting, G. A. Chukwudebe, E. C. Nwokorie, I. I. Ayogu, Multi-label classification of hate speech severity on social media using bert model, in: 2022 IEEE Nigeria 4th International Conference on Disruptive Technologies for Sustainable Development (NIGERCON), 2022, pp. 1–5. doi:10.1109/NIGERCON54645.2022.9803164.

[11] K. Bhatia, K. Dahiya, H. Jain, P. Kar, A. Mittal, Y. Prabhu, M. Varma, The extreme classification repository: Multi-label datasets and code, 2016. URL: http://manikvarma.org/downloads/XC/XMLRepository.html.

[12] V. Sanh, L. Debut, J. Chaumond, T. Wolf, Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, ArXiv abs/1910.01108 (2019). URL: https://api.semanticscholar.org/CorpusID:203626972.

[13] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: J. Burstein, C. Doran, T. Solorio (Eds.), Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. URL: https://aclanthology.org/N19-1423. doi:10.18653/v1/N19-1423.

[14] P. J. Huber, Robust estimation of a location parameter, Annals of Mathematical Statistics 35 (1964) 492–518. URL: https://api.semanticscholar.org/CorpusID:121252793.

[15] A. Ahmadvand, S. Kallumadi, F. Javed, E. Agichtein, DeepCAT: Deep Category Representation for Query Understanding in E-commerce Search, in: S. Kallumadi, T. H. King, S. Malmasi, M. de Rijke (Eds.), Proceedings of the SIGIR 2021 Workshop on eCommerce, co-located with the 42st International ACM SIGIR Conference on Research and Development in Information Retrieval, eCom@SIGIR 2021, Online, 2021, p. n. pag.

[16] A. Joulin, E. Grave, P. Bojanowski, T. Mikolov, Bag of tricks for efficient text classification, in: M. Lapata, P. Blunsom, A. Koller (Eds.), Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, Association for Computational Linguistics, Valencia, Spain, 2017, pp. 427–431. URL: https://aclanthology.org/E17-2068.

[17] L. Zhu, K. Zhang, H. Chen, C. Wei, W. Zhang, H. Tang, X. Li, HCL4QC: Incorporating Hierarchical Category Structures Into Contrastive Learning for E-commerce Query Classification, in: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 3647–3656. URL: https://doi.org/10.1145/3583780.3614907. doi:10.1145/3583780.3614907.

[18] B. He, S. Nag, L. Cui, S. Wang, Z. Li, R. Goutam, Z. Li, H. Zhang, Hierarchical query classification in e-commerce search, 2024. arXiv:2403.06021.

[19] H. Wang, T. Na, Rethink e-commerce search, in: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, WSDM '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 1653. URL: https://doi.org/10.1145/3488560.3510005. doi:10.1145/3488560.3510005.